Modeling A Multi-segment Wargame Leveraging Machine Intelligence and Event-Verb-Event (EVE) Structures

Ying Zhao, Bruce Nagy, Tony Kendall, Riqui Schwamm *

Abstract

The paper depicts a generic representation of a multi-segment wargame leveraging machine intelligence with two opposing asymmetrical players. We show an innovative Event-Verb-Event (EVE) structure that is used to represent small pieces of knowledge, actions, and tactics. We show the wargame paradigm and related machine intelligence techniques, including data mining, machine learning, and reasoning AI which have a natural linkage to causal learning applied to this game. We also show specifically a rule-based reinforcement learning algorithm, i.e., Soar-RL, which can modify, link, and combine a large collection EVEs rules, which represent existing and new knowledge, to optimize the likelihood to win or lose a game in the end. We show a simulation and a real-time test for the methodology.

Introduction

In recent years, machine learning (ML) has successfully used back-propagation and large data sets to reach human level performance. The techniques have been used to solve difficult pattern recognition problems as perceptive artificial intelligence or perceptive AI for many types of use cases. These algorithms implement learning as a process of gradual adjustment of underlying models' parameters (Lake et al. 2016). However, although there is great potential for these techniques to be applied to real-life, they have been criticized for being black boxes and lacking understanding of causality which can be very important for decision makers. Reasoning AI such as reinforcement learning (Sutton and Barto 2014) and game theory (Brown and Sandholm 2017) have been successful as well in terms of producing human level performance benchmarks. Convolutional neural networks and reinforcement learning combined can achieve the best perceptive AI for input data of imagery and acoustics for superior human level of performance (Silver, Schrittwieser, and Simonyan 2017).

These technologies have the great potential to address the unique challenges of modeling complex functions of defense applications including mission planning, decision making, and causal reasoning. Leveraging machine intelligence, in the sense of leveraging big databases, existing/new knowledge, and tactics repositories, is critical for the future success of defense applications. For example, when warfighters make decisions, they need to take into considerations of all possible states of different types of opponents and adversaries' intentions, strategies, decisions, and actions, which can be overwhelming for humans. Machine intelligence tools are needed for assisting humans to reduce their cognitive load. The paper presents a use case and a real-life test with a need to elevate machine intelligence to assist mission planning, decision making, and causal learning for warfighters.

EVEs Structures and Multi-segment Wargame

We first define a generic representation of a multi-segment wargame with two opposing asymmetrical players as shown in Figure 1. Such a wargame is divided into multi-segments with events and verbs or actions alternating with a self-player and opponent. For each player, actions characterized by the verbs are grouped into a few categories, e.g., V_A, V_B, V_C, V_D, and V_E. These categories can represent actions typically used in various warfare areas. Events generated by the actions or verbs happen sequentially or in parallel in each segment. Probabilistic rules $E \rightarrow V$ and $V \rightarrow E$ present the valid moves and probability of states. An EVE example would be: "If an opponent is found (event), then track (verb) the opponent using tool A" and "if the opponent has been successfully tracked (event), then target (verb) the opponent using tool B." Figure 2 shows a list of action options for Segment 1 in a very high level for a test game named "Battle Readiness Engagement Management (BREM)" as shown in Figure 3. Such EVEs can be different or asymmetrical for each player, i.e., two opposing asym-

 Ying Zhao, Tony Kendall, and Riqui Schwamm are with the Naval Postgraduate School, Monterey, CA; Bruce Nagy is with the NAVAIR, China Lake, CA.

metrical players have their own sets of EVEs rules guiding corresponding valid moves. The verbs or actions consume time and other costs. An event represents a single measurable outcome or state after an action. Events are discrete and do not consume time but have value (e.g., contribution to win or lose a game in the end). They are evaluated by a set of unifying equations to determine expected winning, losing, or drawing status for each of the opposing asymmetrical players.

Where Do EVEs Structures Come From?

EVEs rules are generated top-down from experts, or learned bottom-up using unsupervised learning from historical data and knowledge repositories as shown in Figure 4:

- The bottom up approach applies data mining performed on the historical unstructured text data (e.g., wargame logs) using entity/event extraction tools such as spaCy (sciSpaCy 2020) or Bert (Beltagy, Lo, and Cohan 2019) tools and sequential pattern/link analysis tools such as lexical link analysis. The data mining process output initial EVEs structures that may include the ones are not causal.
- The top approach ingest existing databases such as databases about red/blue capabilities combined with human experts on-the-loop process to design the initial EVEs. Since it is a human-on-the-loop process, the resulted EVEs are causal. The human experts can also validate, filter, and combine the EVEs from the data mining process and make sure they are causal.
- The initial EVEs are the input to the reinforcement learning or other machine learning algorithms to reason the best course of actions for blue/red. The integration of machine and causal learning techniques has the potential for tactical decision edge for the players.

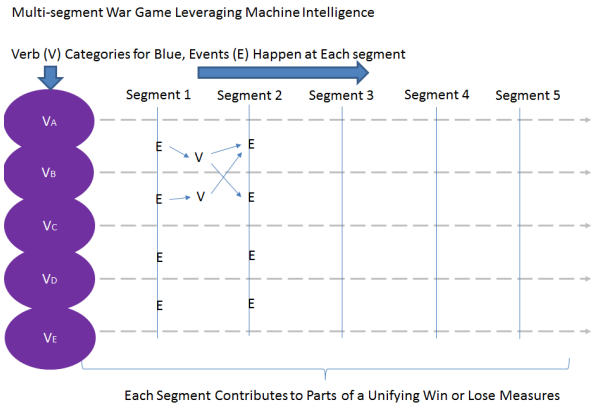

Figure 1: A wargame divided into multi-segments with events and verbs alternating with opposing asymmetrical players

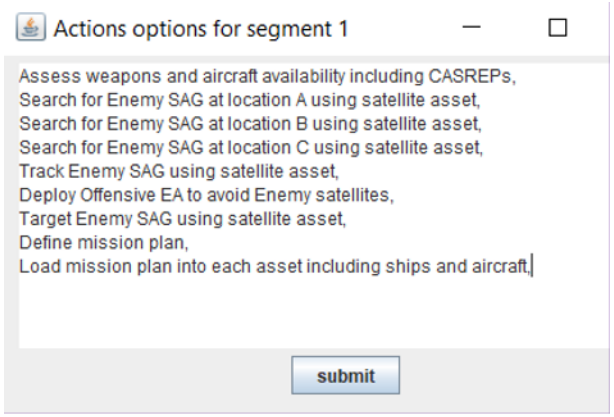

Figure 2: Action options for Segment 1

The novelty/significance of the EVEs structures is that they are used to describe small pieces of knowledge, actions, and tactics which can be systematically linked and combined to optimize a global measure of effectiveness (MOE), e.g.,likelihood to win or lose a game in the end. Such EVEs repositories can be extremely large; some EVEs rules may be outdated or inconsistent – as they can be accumulated over a long period time. New rules and tactics from big data, new sensors are necessary to be incorporated into current and future warfighting planning and executions. As shown in Figure 3 of the BREM game, searching, optimizing, learning, and gaming with novel course of actions using a large collection of knowledge, patterns, and rules from databases can only be made possible for warfighers with the help of computing power and machine intelligence algorithms. One novelty/significance of this paper is to show how to apply a specific form of machine learning of reinforcement learning (RL), i.e., Soar-RL, to select, modify, link, and combine the EVEs rules. Finally, our wargame and machine intelligence paradigms possess natural linkages to causal learning that is especially important to warfighting activities such as mission planning, cognitive behavior, and intent pattern recognition.

Causal Learning Factors

Our paper first offers a paradigm and supported evidence that our wargame definition which have a natural linkage

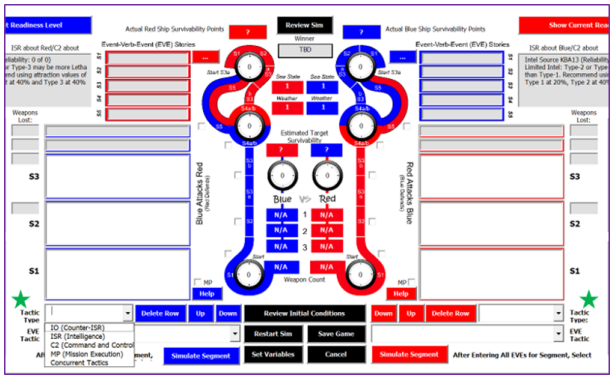

Figure 3: Battle Readiness Engagement Management (BREM) Game

to causal learning in the following aspects. It relates to the three layers of a causal hierarchy (Mackenzie and Pearl 2018) (Pearl 2018) - association, intervention, and counterfactuals, as well as a few other key elements of causal learning as detailed in the following sections.

Association

Association is the lowest level in the causality ladder hierarchy. The common consensus of statisticians is that data-driven machine intelligence analysis, including data mining and various flavors of ML, is appropriate in discovering statistical correlations from data. However, machine intelligence requires human analysts to validate the correlations to conclude, in a scalable and error-prone way, which correlations are causal and which ones are co-accidental. For instance, the EVEs rules can be learned and extracted from historical documents as correlations, associations, and sequential patterns. These rules are later validated by human experts in their area of domain expertise.

Intervention

Intervention ranks higher than association in the causality ladder hierarchy which involves taking actions and generating new data. A typical question at this level of causality ladder would be: What will happen if we increase the intensity of an action defined by a verb? The answers to the question are more than just mining the existing data. They need to have a new data generated in reaction to an intervention to see if the underlying action is the cause to the desired effect (e.g., winning the game) or how sensitive the effect to the cause is. The intervention can be modeled as an action or an verb. For example, instead of examining $P(X|M)$, i.e., the likelihood of observable data X given the model M, one should further make sure M is actionable or $P(X|do(M))$ can be examined. The EVEs structures contain verbs as possible actions, therefore act as interventions naturally. Soar-RL and related global MOE are designed to evaluate the effects of the interventions or action/state combinations are shown in Sec. .

Counterfactuals

The top of the causality ladder hierarchy is a typical question asked "What if I had acted differently?" Traditionally, the effect is defined as the outcome of an action for an entity and for the same entity without the action, i.e.,$P(E|C)$–$P(E|Not\ C)$. However, since this causal effect is impossible to be directly observed for the same entity. This is commonly referred to as the fundamental problem of causal inference. The predicted counterfactual outcome or counterfactual-based model of causal inference has led to key breakthroughs in applied statistics and game AI. The conceptual advances come from the idea of an entity-level action or "treatment" effect, although it is unobservable, can be predicted in various ways.

For example, the causal effect is typically measured using randomized two populations, one with the "treatment" or action (or cause C) and other one without the "treatment" or action (Not C or control group), two populations are randomized to ensure they are similar to each other (as if they are the same entity) in average and in all other dimensions except the treatment dimension. This is the randomized control treatment (RCT) theory, which is a standard practice in the social sciences, drug development, and clinic trials.

With recent data-driven approaches such as data mining and machine learning, people can robustly estimate a local average treatment effect in the region of overlap between treatment and control populations, but inferences for averages, outside this zone are sensitive to underline machine learning algorithms. For example, people have applied non-parametric models of machine learning such as nearest neighbors and random forests (Wager and Athey 2018) for better causal learning since these methods can approximate the local treatment and control populations close to a real RCT setting.

Relations of Machine Intelligence Algorithms

Traditional statistical analysis heavily depends on hypothesis tests, where the likelihood of the data $P(X|H)$ given a hypothesis H is compared to the a null hypothesis H_0. The hypothesis test directly relates to the MLE where the likelihood of the data $P(X|M)$ given a model M is estimated. The general assumption is that the model is a gener-

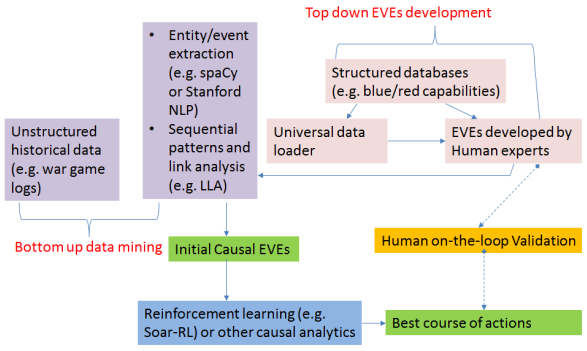

Figure 4: Causal learning integration for the war game

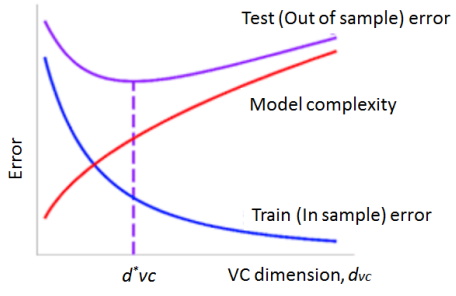

Figure 5: ML algorithms and model complexity

ative model of the data if the likelihood is maximized compared to all other models, therefore, the model is the cause of the data (effect). The generative models of the current ML/AI (Rezende, Mohamed, and Wierstra 2014) related to the hypothesis tests and MLE, also consider given classification labels, what are the likelihood of the data that are observed. Causal learning can be considered as a subclass of generative models because, at an abstract level, that resembles how the data are actually generated as shown in reinforcement learning (Sutton and Barto 2014).

The current machine learning techniques are different from the maximum likelihood estimation approach, for example, neural networks or many other machine learning classifiers which directly estimate posterior probabilities of the models (e.g., probability of a classification given the data), i.e., estimate $P(M|X)$. The posterior probability estimation related approaches have been criticized for being black boxes, lack of explainability and causality.

The two approaches have been competing historically. One of the important AI applications is the automatic speech recognition, where Hidden Markov Models (HMMs) have been the leading approach since the late 1980s (Juang and Rabiner 1990), yet this framework has been gradually replaced with deep learning components which are considered a better approach than HMM for speech recognition (Hintonb et al. 2012). In other words, although these black box ML/AI techniques are not often causal or explainable, they do work well in applications in term of producing accurate prediction and classification for new data (Wager and Athey 2018).

To explain this blackbox effectiveness, for many machine learning algorithms, researchers apply cross-validation, regularization, and transfer learning theories. The focus of supervised machine learning is to make accurate prediction or classification for out of sample data, i.e., new data that do not show in sample of train data. For example, machine learning classifiers, a typical classification error graph for train (in-sample) and test (out-of-sample) errors, with respect to the complexity of the models, e.g., measured by so called Vapnik–Chervonenkis dimension or VC dimension (Vapnik 2000), is shown in Figure 5. From machine learning perspective, out-of-sample error is always larger than the in-sample error (i.e., so-called overfitting problem), there is an optimal model complexity d*vc which gives the lowest test error for a given train data. This is usually accomplished using cross-validation or regularization to avoid the overfitting (Bishop 2007). The theories of cross-validation and regularization theories suggest to keep the models as small and smooth as possible so the difference between in-sample error and out-of-sample error is minimized. Transfer learning theories suggest to consider fundamental reasons why some learning results can be transferred from one data set to another, or from one area to another. For example, in deep learning, the initial layers of neural networks learn the basic features of machine visions, e.g., edges and corners of images, which can be transferred across domains and data sets.

When the applications require causality analysis such as in a multi-segment wargame we study in this paper, we need to integrate the reasonable causal elements with data-driven machine learning approaches together, and always keep human experts on-the-loop for interpreting the cause and effect relations.

In our setting, the EVEs structures $E \rightarrow V$ rules, especially how actions are made to use these rules, belong to machine learning techniques; while the $V \rightarrow E$ rules are generative rules.

Reinforcement Learning

One of the most successful machine learning techniques is reinforcement learning where an AI agent takes action and generates a new state. It learns from reward data from the environment by modifying its internal models. Reinforcement learning is considered a causal learning model in this context since it is designed to generate the desired data (effect) by taking the right actions (cause). The EVEs structures allow perform reinforcement learning following certain constraintsfrom large-scale existing knowledge bases.

Soar and Reinforcement Learning (Soar-RL)

Soar (Laird 2012) (Laird, Derbinsky, and Tinkerhess 2012) is a cognitive architecture that scalably integrates a rule-based AI system with many other capabilities, including reinforcement learning and long-term memory. The main decision cycle of Soar involves rules that propose new operators (e.g., internal decisions or external actions), as well as preferences for selecting amongst them; an architectural operator-selection process; and application rules that modify agent state. A preference is defined as the probability, contribution, or impact to reach the desired outcome (event) if an operator is selected. 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 warfighting processes like the ones in a kill chain (Zhao, Mooren, and Derbinsky 2017).

Machine Learning in Soar-RL

A multi-segment wargame as we defined is played by a self-player and her opponent. There are large collections of different (asymmetrical) actions (verbs) for both players based on initial collections of EVEs rules. A state is the input data to a self-player that can not be decided or controlled by herself. Part of states may refer to the state of an opponent. An opponent may be the environmental factor such as rain or no rain. A combination of the self-player's actions and states of the self-player can result in certain reward for the self-player, for example, win or lose a game in the end. In some cases, the environmental factors can be the opponent. The opponent of a self-player can be a competitor or an adversarial who can take deliberate actions to defeat the self-player. In any of these cases, the self-player needs to constantly simulate the behavior and intent of the potential opponent and take the best of course of actions. If the opponent is hidden

Table 1: Asymmetric Action/State Combinations

Self-Player Action/state combination d_1	Opponent (e.g.,adversarial) o_1
Action/state combination d_i	o_j
...	...
Action/state combination d_N	o_M

Table 2: Action/State Combination Components

Action/State Combination	f_1	f_i	...	f_K	End Reward
d_i	1	0	...	1	win
	...	...	...	...	not win

or information about the opponent is not perfect or the opponent's intent changes dynamically in the game (Brown and Sandholm 2017), the self-player's course of action (COA) needs constantly adjust, re-dynamically program,and adapt her COAs.

Table 1 shows a self-player and opponent taking asymmetric action/state combinations. Table 2 shows each action/state combination can consist of multiple components. An action is a decision of the self-player needs to decide that can maximize her reward along the game timeline in the end. An action/state combination d_i consists of a sequence of components f_k with its value v_1 or v_0 that the self-player needs to decide. An f_k with its value v_1 or v_0 can be an EVEs rule or tactics selected from a library of rules with parameters. An f_k with its value v_1 or v_0 can be also the state of herself (e.g., capability of her defense) that she needs to consider when making decisions; an f_k with its value v_1 or v_0 can be also the state of the opponent that the self-player has to estimate from observable data (e.g., sensor data).

We use an action/state combinations instead of a course of action or COA because an action/state combination represent more flexible sequential and parallel actions and states in a wargame, while a COA refers more of traditional sequential actions taken by warfighters.

Soar-RL Details

In a Soar-RL, a preference is defined as the probability of a rule to be used with respect to a total reward. To translate into the multi-segment wargame, a preference is the contribution of an EVE rule or f_k to be selected for a self-player to win. Define preferences $f_k\_v_1\_c_1$, $f_k\_v_0\_c_1$, $f_k\_v_1\_c_0$, and $f_k\_v_0\_c_0$, where $f_k\_v_1 - c_1$ means "if an action/state combination component f_k is included ($v = 1$), there is a preference (probability) $f_k\_v_1\_c_1$ for the self-player to win the game in the end ($c = 1$)."

We show how the preferences can be computed for the rules. Let m be the number of rules and N the number of data for Soar-RL to perform on-policy learning (Laird 2012) (Laird, Derbinsky, and Tinkerhess 2012).

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

Since we only consider an on-policy setting or SARSA, $Q(s_{t+1}, a) = 0$ and let

$$\delta_t = \alpha(r_{t+1} - Q(s_t, a_t)) \quad (2)$$

$\alpha, r_{t+1} = 1$ for a positive reward or -1 for a negative reward. In order to converge, $r_* = Q(s_*, a_*)$ in Eq. (2), we ask: Is there a set of preferences p_1,p_2,...p_m that makes δ_t in Eq. (2) as small as possible when $t \to \infty$.

The total probability of winning for an action/state combination is the summation of the preferences from each of the action/state combination components (Q-value in Eq. (1)). For any action/state combination d_i which consists of K components included ($v = 1$) and K' components not included ($v = 0$). Equation (3) predicts a win in the end.

$$\sum_{k=1}^{K} f_k\_v_*\_c_1 > \sum_{k'=1}^{K'} f'_k\_v_*\_c_0, \quad (3)$$

where $*$ denotes value 1 or 0. The self-player gains a positive reward 1 if a correct action is taken at time t or a negative reward -1 if a wrong action is taken. For example, for an action/state combination, total preference added for win is 4 and lose is 1, the predicted result would be win. If the truth is indeed win for this combination for the self-player, then each of the K win rules' preferences related to the combination is modified using a positive reward $\frac{1}{K}$. If the ground truth is lose for this combination, each of the same K rules' preferences is modified using a negative reward $-\frac{1}{K}$. In other words, Soar-RL always modify the rules that involve the predicted win or lose. Note some components that are not included ($v = 0$) can also contribute positively to the win ($c = 1$) in Eq. (3), this is an example of counterfactuals considered in Soar-RL. Figure 6 (a) shows an example of the BREM game where Soar suggests an action component, i.e., "C2 (Command and Control) - Assess weapons and aircraft availability including CASREPs." Figure 6 (b) shows that Soar makes the suggestion by selecting the highest difference of the scores for class 1 (good) and class 0 (bad) for all nine possible choices of the current segment of the game. Since the score (-0.099093) is negative, the Soar's predicted class is 0 (bad). Figure 6 (c) shows Soar-RL updated the preferences of the rules reflected the predicted class and the input components.

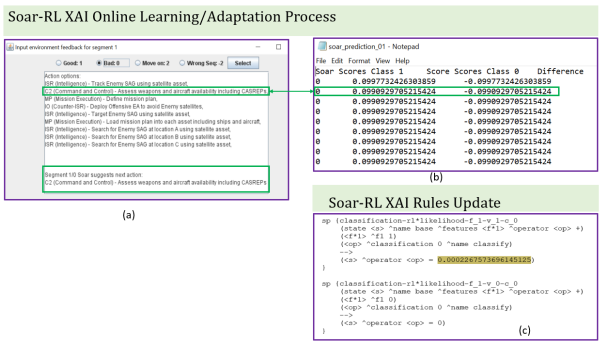

Figure 6: Soar example in BREM

Simulation

The simulation case contains about 50 components for an action/state combination. The 50 components include 30 dimensional actions as a vector S_a possibly taken by a self-player, 20 dimensional states of the opponent (o in Tab. 2), and 10 dimensional states of the self-player. In our representation as in Table 2, each component of state or action is represented as binary 1 or 0 as a Boolean lattice (Boolean 2019). The training set contains about 1 million action/state combinations that have win and lose tagged for the end game results. The test set contains about 300,000 action/state combinations. The value (good or bad) of each action/state combinations is based on if the self-player wins (good) or loses (bad) a game. There is less than 10% of the combinations are good combinations. There are possible 2^{50} combinations for the self-player's action and state components. The sample data sets are only part of all the possible combinations. The paper focused on applying Soar-RL to learn the value (win or lose a game) function from sample action/state combinations as shown in Eq. (4):

$$Win\ or\ lose = f(\mathbf{S_a}, \mathbf{S_s}, \mathbf{O_s}) \tag{4}$$

When a self-player simulates and performs what-if analysis in the future, she can use Eq. (4) and optimization algorithms to search actions S_a when varying S_s, O, or both.

We used Soar-RL with a fixed learning rate $\alpha = 0.0004$ and reward values 1 when the value is predicted correctly compared to the ground truth and -1 when the value prediction is incorrect compared to the ground truth.

Convergence of Soar-RL

Since Soar-RL is an online on-policy machine learning algorithm, it is important to show the algorithm converges in theory and in practice. Figure 7 shows the convergence of the changes of the preferences for the use case when the iteration is 20. The convergence of the preferences can be proved using the game theory and reinforcement learning theory.

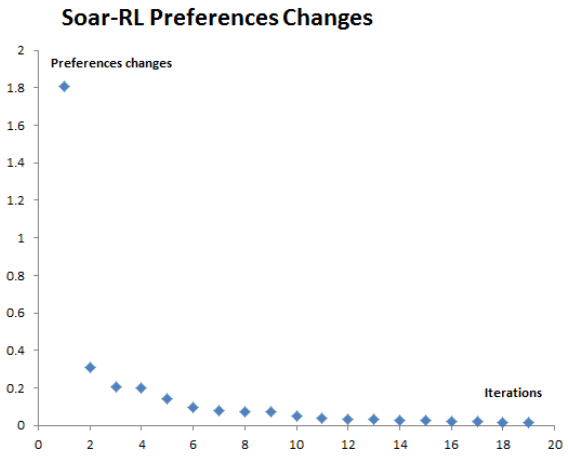

Figure 7: The convergence of learning preferences of the rules in Soar-RL

Soar-RL and Counterfactuals

When a Soar-RL learns/updates the rules, it learns/updates the preferences of a component as well as the preferences of the counterfactuals. In other words,

- $P(good\ result|component\ k\ included)$,
- $P(good\ result|component\ k\ not\ included)$,
- $P(bad\ result|component\ k\ included)$, and
- $P(bad\ result|component\ k\ not\ included)$,

where an action/state combination with K components is included and K' component not included, are estimated independently and used for causality reasoning to see if a component k is a cause for good or bad result (effect).

We also compute

$$\begin{aligned} &P(good\ result|an\ action/state\ combination) \\ &= \sum_{k=1}^{K} P(good\ result|component\ k\ included) \\ &+ \sum_{k=1}^{K'} P(good\ result|component\ k\ not\ included) \end{aligned} \tag{5}$$

and

$$\begin{aligned} &P(bad\ result|an\ action/state\ combination) \\ &= \sum_{k=1}^{K} P(bad\ result|component\ k\ included) \\ &+ \sum_{k=1}^{K'} P(bad\ result|component\ k\ not\ included). \end{aligned} \tag{6}$$

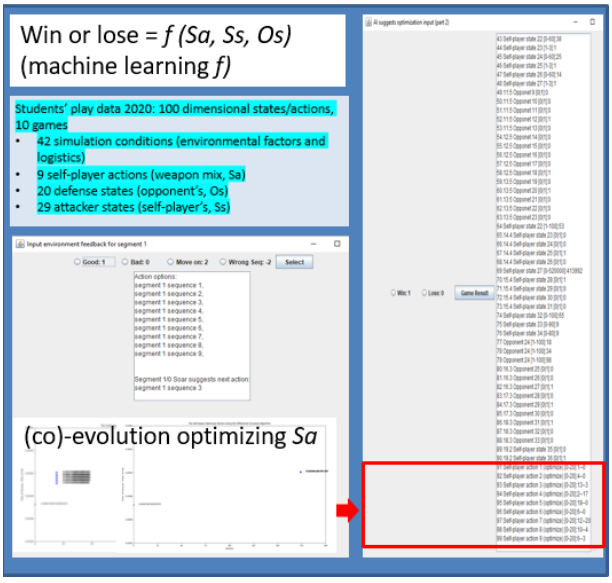

Figure 8: Students Test Setup at the Naval Postgraduate School (NPS)

The difference between
$P(good\ result|an\ action/state\ combination)$ and
$P(bad\ result|an\ action/state\ combination)$
is used to predict if an action combination is good or bad.

Soar-RL and Explainable AI (XAI)

Soar-RL is also based on understandable EVEs rules and provides the advantage of explainable AI (XAI) (xai 2018). The rules used in the prediction and updating are listed in Figure 6 (c).

The Test at the NPS

The BREM game can be played by a human, e.g. such Naval Postgraduate School (NPS) students (i.e., the blue player) against the AI assistant (i.e., the red player) as shown in Figure 3. Once we receive the NPS Institutional Review Board (IRB) approval we will organize an event and recruit students who will test the BREM game and played against the AI assistant as shown in Figure 8. There are 100 dimensional states/actions in total, 42 simulation conditions (environmental factors and logistics), 20 defenser's states (opponent's), 29 attacker's states (self-player's), and nine self-player actions, i.e., the weapon mix. The nine self-player's actions (Sa) will be compared what the AI assistant's actions powered by the algorithms. Soar-RL is used to learn the value function (win or lose) f as shown in Figure 8. From preliminary observations among researchers playing the BREM game, we collected data from ten games in total. Before the machine learning, human players won eight games and the AI assistant won one game. The nine games were used in the machine learning combining Soar-RL with DE. After the machine learning, the AI assistant won one game that a human player lost. Lesson learned is that human players tend to use default or known actions while the AI assistant was able to search a wide range of possibilities of actions when performing the machine learning.

Differential Evolution (DE) Algorithms

We focus on evolutionary algorithms for optimization for the nine self-player's actions. Genetic algorithms are evolutionary algorithms (Goldberg 1989) which keep the metaphor of genetic reproduction of selection, mutation, and crossover where the objective function's derivatives are not easy to compute, therefore gradient decent type of algorithms can not be applied for optimization. Differential evolution (DE) (Rocca, Oliveri, and Massa 2011) is in the same style, however, deals with the optimization parameters in real numbers as floats, or doubles. For continuous parameter optimization, DE is a derivative-free optimization inspired by the theory of evolution, where the fittest individuals of a population are more likely to survive in the future, the population improves generation after generation.

Conclusion

In this paper, we showed the EVEs structures and integration of machine and causal learning and optimization techniques used in modeling a multi-segment wargame. We showed how the EVEs structures and related machine intelligence techniques used to link, modify, and update a large collection of existing and new knowledge and tactics for a multi-segment wargame. We also illustrated the critical elements of causal learning for the EVEs structures and Soar-RL. We tested the methodology with a simulation data set and a real-life game test with the NPS human players. The integration of machine and causal learning techniques has the potential for a wide range of tactical decision edge applications.

Acknowledgements

Authors would like to thank NAVAIR China Lake, the Office of Naval Research (ONR), the NPS's Naval Research Program (NRP), and DARPA's Explainable Artificial Intelligence (XAI) program for supporting 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

Beltagy, I.; Lo, K.; and Cohan, A. 2019. A pretrained language model for scientific text. Retrieved from https://arxiv.org/abs/1903.10676.

Bishop, C. M. 2007. *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer.

Boolean. 2019. Boolean lattice. Retrieved from https://www.sciencedirect.com/topics/mathematics/boolean-lattice.

Brown, N., and Sandholm, T. 2017. Safe and nested endgame solving for imperfect-information games. Proceedings of the AAAI workshop on Computer Poker and Imperfect Information Games.

Goldberg, G. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley.

Juang, B. H., and Rabiner, L. R. 1990. Hidden markov models for speech recognition. *Technometric* 33(3):251–272.

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.

Laird, J. E. 2012. *The Soar Cognitive Architecture*. Cambridge, MA, USA: MIT Press.

Lake, B. M.; Ullman, T. D.; Tenenbaum, J. B.; and Gershman, S. J. 2016. Building machines that learn and think like people. Retrieved from https://arxiv.org/abs/1604.00289.

Mackenzie, D., and Pearl, J. 2018. *The Book of Why: The New Science of Cause and Effect*. New York, NY, USA: Penguin.

Pearl, J. 2018. The seven pillars of causal reasoning with reflections on machine learning. Retrieved from https://ftp.cs.ucla.edu/pub/stat_ser/r481.pdf.

Rezende, D. J.; Mohamed, S.; and Wierstra, D. 2014. Stochastic backpropagation and approximate inference in deep generative models. Proceedings of the 31st International Conference on Machine Learning (ICML).

Rocca, P.; Oliveri, G.; and Massa, A. 2011. Differential evolution as applied to electromagnetics. *In IEEE Antennas and Propagation Magazine* 53(1):38–49.

sciSpaCy. 2020. scispacy. Retrieved from https://allenai.github.io/scispacy/.

Silver, D.; Schrittwieser, J.; and Simonyan, K. 2017. Mastering the game of go without human knowledge. *Nature* 550:354–359.

Sutton, R. S., and Barto, A. G. 2014. *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press.

Vapnik, V. 2000. *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer.

Wager, S., and Athey, S. 2018. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association* 113(523):1228–1242.

xai. 2018. Darpa xai. Retrieved from https://www.darpa.mil/program/explainable-artificial-intelligence.

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