Visualizing the Evolution of Multi-agent Game-playing Behaviors

Shivam Agarwal[†1], Shahid Latif[1], Aristide Rothweiler[2], and Fabian Beck[1]

[1]University of Bamberg, Germany [2]University of Duisburg-Essen, Germany

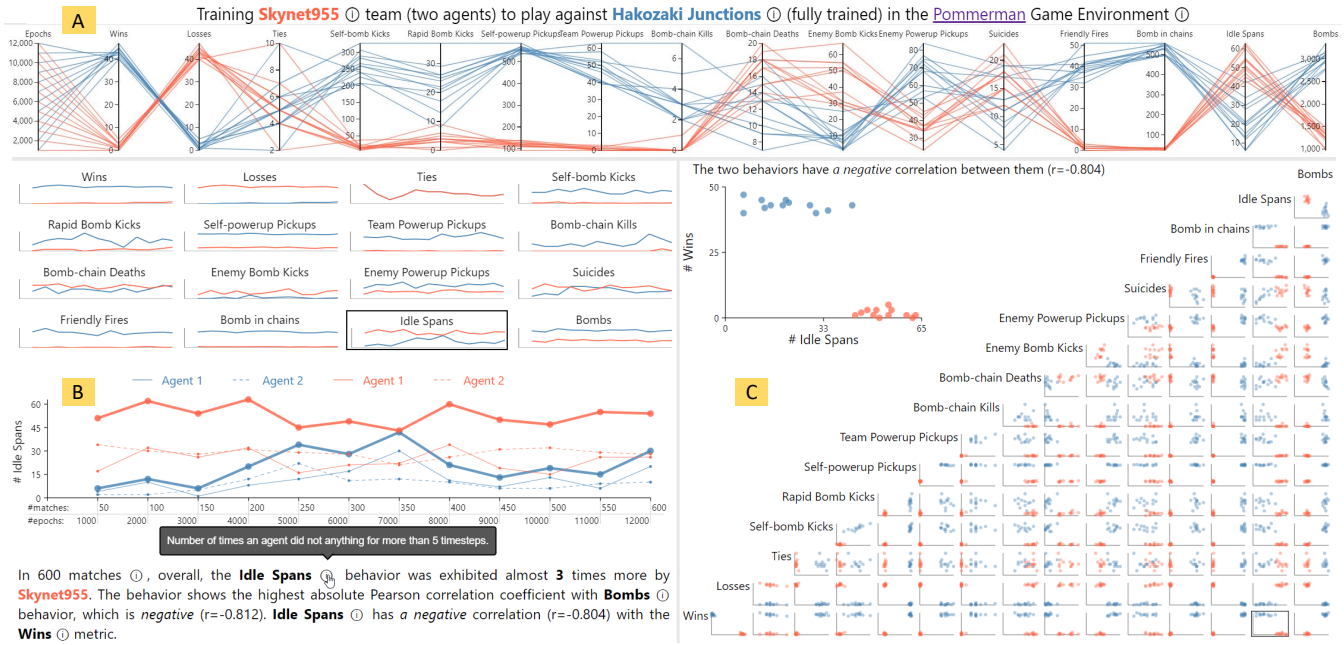

Figure 1: *Visualizing behavior and game statistic metrics of a training: (A) An overview of all metrics, (B) evolution of invidiual metrics, and (C) correlation among metrics. Demo at* https://s-agarwl.github.io/evolvingai.

Abstract

Analyzing the training evolution of AI agents in a multi-agent environment helps to understand changes in learned behaviors, as well as the sequence in which they are learned. We train an existing Pommerman team from scratch and, at regular intervals, let it battle against another top-performing team. We define thirteen game-specific behaviors and compute their occurrences in 600 matches. To investigate the evolution of these behaviors, we propose a visualization approach and showcase its usefulness in an application example.

Keywords: Multi-agent system, multivariate data, evolving gameplay behaviors, AI training, visualization.

1. Introduction

Developing AI agents in game environments is becoming increasingly popular as a testbed for machine learning. Analyzing evolving behaviors in AI training helps discover crucial stages in the training. For instance, AlphaZero [MKT*21] exhibited drastic changes in game-playing behavior (e.g., selecting the first move in the game of chess) during 25k–60k epochs during training. We focus on multi-agent environments (e.g., [REH*18, MNL*20]), which allow studying teams of AI agents who learn to cooperate, with the aim to use such skills in real-world scenarios or understand the underlying training approach better. In such environments, the agents are trained via an optimization function by focusing on a performance metric (e.g., reward value), which is tailored toward winning the

† email: shivam.agarwal@uni-bamberg.de

DOI: 10.2312/evp.20221111

maximum number of matches. However, this quantitative approach neglects an analysis of (a) which behavior changes have led to improvements in performance, (b) desirable behaviors (e.g., cooperation) that do not immediately reflect in the performance metric, and (c) the sequence in which behaviors are learned. Few visualization approaches (e.g., [AWB20, AWWB22]) enable a qualitative exploration of team behaviors, but they only focus on analyzing the fully trained agents. Addressing the challenge, we propose a visual analysis approach for evolving behaviors of AI agents in the bomb-laying game environment Pommerman (Figure 1).

2. Game Environment and Data Collection

Pommerman [REH*18] is a bomb-laying game environment, where two competing teams—each comprising two agents—try to eliminate enemies in an 11×11 grid map. An agent can move, drop a bomb (which explodes after 10 time steps), and uncover hidden power-ups by blasting wooden walls. Available power-ups allow agents to increase the blast radius, kick bombs, and drop bombs without waiting for the previous one to explode. Bomb chains can be formed if the blast flame engulfs other bombs, making them explode instantly.

To analyze the evolution of behaviors, we used two top-performing teams in the NeurIPS 2018 Pommerman competition. *Hakozaki Junctions* [OT19] (HJ) is a tree search technique and won the competition. On the other hand, *Skynet955* [GHLKT19] (S955) is a neural network that learns via self-play using reinforcement learning, placed second in the 'learning agents' category of the competition. To collect the data, we trained the S955 team again from scratch. After every 1000 training epochs, we recorded 50 matches against the HJ team, through 12 intervals (until 12000 epochs). We defined and quantified 13 behaviors from the recordings, e.g., *Enemy Powerup Pickups*, which is defined as the number of times an agent grabbed a power-up that was uncovered by an opponent. In each interval, the values of game results per team (*Wins*, *Losses*, and *Ties*) and quantified behaviors are summed across all 50 matches. This results in a multivariate dataset (13 behaviors and three game statistics across 12 training intervals.

3. The Visualization System

Our visualization system (Figure 1) consists of three linked views. Enabling comparison between teams, we sum the behavior values of two agents in a team. To make the system easy to use, we leverage adaptable captions and consistent color linking. The information icon ⓘ in the captions can be hovered to see details in a tooltip, e.g., definition of a behavior in Figure 1B.

The parallel coordinates plot (A) provides an **overview** and compares the behaviors of the two teams, S955 and HJ. At a glance, it reveals clear differences between the teams in the frequency of some behaviors (e.g., *Self-powerup Pickups*, *Bombs*) or similar frequencies in other behaviors (e.g., *Suicides*, *Idle Spans*). It indicates that S955 is yet to learn some behaviors (e.g., *Bomb-chain Kills*). Since HJ performs far better than S955 (majority of *Wins*), we can quickly see—following the bundle of blue lines—which behaviors might have lead to success.

The small multiple line charts (B) offer a comparative view of teams' **evolving** behavior frequencies along training epochs. Each small line chart can be clicked to see an enlarged version below that discerns both agents of a team as different lines (dotted and solid). The adaptable caption of the plot summarizes the frequency, describes the behavior that was highly correlated with the selected behavior, and mentions the correlation with *Wins*. For instance, the selection of *Idle Spans* shows that S995 exhibits this behavior almost three times more than that of HJ, but with a much smaller difference in the middle of the training and at the end. Also, the behavior was negatively correlated with dropping *Bombs* ($r = -0.812$). The small multiple also indicates that the S955 team commits many suicides across all training intervals (), but shows early signs of a declining trend near the end of the training.

The scatterplot matrix (C) reveals **statistical relationships** between all behaviors and game results. The last two rows of the scatterplot matrix are particularly interesting, as they indicate potentially beneficial behaviors (positively correlated with *Wins*—e.g., *Bomb in Chains*, *Enemy Powerup Pickups*) or potentially disadvantageous ones (negatively correlated with *Wins*—*Suicides*, *Enemy Bomb Kicks*). Other pairwise correlations describe the relationship of one behavior with others and may help refine the reward function; optimizing for one behavior could affect other behaviors if they are correlated to that specific behavior. Every pairwise plot can be clicked to see its larger version for a closer inspection.

4. Application Example

From a developer's perspective of team S955, just based on the number of wins and losses, it is evident that HJ clearly outperforms their opponents, not just at the beginning but throughout the training (cf. Figure 1B). The parallel coordinates plot (Figure 1A) helps in pinpointing behaviors that might be contributing to the low performance; for instance, low frequencies of certain behaviors (e.g., *Bombs*, *Bomb-chain Kills*, *Bomb in Chains*) and high values of *Idle Spans*. To dig deeper, the developer refers to the line plot to see the evolution of *Idle Spans*; the S955 agents first learn to reduce the *Idle Spans* (at epoch 7000), but these spans increase again with further training. Reading the caption, it becomes evident that *Idle Spans* and laying *Bombs* are negatively correlated; therefore, optimizing for one in the reward function might suffice. In the scatterplot matrix, the developer discovers that none of the thirteen behavior is significantly correlated with number of *Ties* (third row from bottom in Figure 1C). Through such analysis, the system provides insights into behaviors that should be rewarded or penalized in the revised reward function for re-training and whether there are anomalies in these behaviors during the training intervals.

Acknowledgments

We thank students of the project group "CompaT AI: Comparative Training of Computer Game AI Agents" at University of Duisburg-Essen, who defined and wrote scripts for quantifying the agent behaviors in Pommerman. This work is partially funded by the German Research Foundation (DFG, Deutsche Forschungsgemeinschaft) under the project "vgiReports" (424960846).

References

[AWB20] AGARWAL S., WALLNER G., BECK F.: Bombalytics: Visualization of competition and collaboration strategies of players in a bomb laying game. *Computer Graphics Forum 39*, 3 (2020), 89–100. doi:10.1111/cgf.13965. 2

[AWWB22] AGARWAL S., WALLNER G., WATSON J., BECK F.: Spatio-temporal analysis of multi-agent scheduling behaviors on fixed-track networks. In *IEEE Pacific Visualization Symposium (PacificVis)* (2022). doi:10.1109/PacificVis53943.2022.00011. 2

[GHLKT19] GAO C., HERNANDEZ-LEAL P., KARTAL B., TAYLOR M. E.: Skynet: A top deep RL agent in the inaugural Pommerman team competition, 2019. doi:10.48550/ARXIV.1905.01360. 2

[MKT*21] MCGRATH T., KAPISHNIKOV A., TOMAŠEV N., PEARCE A., HASSABIS D., KIM B., PAQUET U., KRAMNIK V.: Acquisition of chess knowledge in AlphaZero, 2021. doi:10.48550/arXiv.2111.09259. 1

[MNL*20] MOHANTY S., NYGREN E., LAURENT F., SCHNEIDER M., SCHELLER C., BHATTACHARYA N., WATSON J., EGLI A., EICHENBERGER C., BAUMBERGER C., VIENKEN G., STURM I., SARTORETTI G., SPIGLER G.: Flatland-RL: Multi-agent reinforcement learning on trains. *arXiv* (2020). doi:10.48550/arXiv.2012.05893. 1

[OT19] OSOGAMI T., TAKAHASHI T.: Real-time tree search with pessimistic scenarios: Winning the NeurIPS 2018 Pommerman Competition. In *Proceedings of The Eleventh Asian Conference on Machine Learning* (2019), PMLR, pp. 583–598. URL: http://proceedings.mlr.press/v101/osogami19a/osogami19a.pdf. 2

[REH*18] RESNICK C., ELDRIDGE W., HA D., BRITZ D., FOERSTER J., TOGELIUS J., CHO K., BRUNA J.: Pommerman: A multi-agent playground, 2018. doi:10.48550/arXiv.1809.07124. 1, 2