MAPS - A Metacognitive Architecture for Improved Social Learning

Juan David Vargas[1,2,3], Natalie Kastel[1,3,6], Antoine Pasquali[4], Axel Cleeremans[5], Zahra Sheikhbahaee[1,3,*], Guillaume Dumas[1,3,6,* 1]

[1]Precision Psychiatry and Social Physiology (PPSP) laboratory CHU Sainte-Justine, Montréal, Québec, Canada
[2]DIRO, Université de Montréal, Montréal, Québec, Canada
[3]MILA - Quebec AI Institute, Montréal, Québec, Canada
[4]Université libre de Bruxelles, Bruxelles, Belgium
[5]CrossLabs, Tokyo, Japan
[6]Département de Psychiatrie, Université de Montréal, Montréal, Québec, Canada
[*]Co-Senior Authors

Abstract

Theory of Mind (ToM) and metacognition are essential for social intelligence but remain underexplored in AI beyond basic pattern recognition tasks. This paper introduces MAPS (Metacognitive Architecture for Perceptual and Social learning), which integrates a second-order network (2nd-Net) with cascaded activation to support reflective processing across domains.

We evaluate MAPS in pattern recognition (Blindsight, AGL), single-agent reinforcement learning (SARL; MinAtar) and multi-agent reinforcement learning (MARL; MeltingPot 2.0). MAPS consistently improves performance in Pattern Recognition (PR) and SARL, particularly in complex environments, and shows promising results in high-variability MARL tasks. These findings demonstrate the potential of metacognitive architectures to improve learning and social adaptability in AI systems (AIS).

1 Introduction

In cognitive science, Theory of Mind (ToM) refers to the ability to attribute beliefs, desires, and intentions to others in order to predict their behavior. In AI, ToM represent transformative shift—enabling systems that go beyond mechanistic responses and interact with humans in socially intelligent ways[18, 17]. Closely related is is metacognition—the capacity to monitor and regulate one's cognitive processes. While both involve meta-representations, ToM centers on understanding others' minds, whereas metacognition involves higher-order reasoning about one's mental states.

Neurocognitive research shows that metacognition and Theory of Mind (ToM) share neural and cognitive foundations, with metacognition enhancing ToM and supporting better social outcomes[16, 10, 3]. Theories on social cognition [8], suggest this connection may arise from the brain's capacity to model its own internal states, which could form the

basis for understanding the minds of others.

Building on this connection, AI increasingly incorporates metacognition to enhance artificial social cognition. By combining self-monitoring with social reasoning, metacognitive architectures support flexible learning and strengthen ToM capacities—enabling AIS to engage in more humanlike interactions[6, 21, 4].

One approach to embedding metacognition in AIS is through a second-order network (2nd-Net), which pairs a primary task network with a secondary system that monitors and evaluates its performance. This layer assesses confidence, identifies knowledge gaps, and initiates adjustments to optimize decision-making [14].

Although metacognition is theorized to support ToM in AIS, current methods focus on low-level tasks like pattern recognition (PR), missing its potential in modeling complex interactions [9]. Reinforcement Learning (RL) offers a promising alternative, engaging agents in dynamic, prosocial settings [12].

To bridge this gap, we test whether a 2nd-Net improves AI performance beyond PR tasks, extending to single- and multi-agent reinforcement learning (SARL and MARL). We introduce MAPS (Metacognitive Architecture for Perceptual and Social learning), a streamlined adaptation of Pasquali & Cleeremans' 2nd-Net [1], designed to integrate metacognition across both PR and social learning domains.

PR is tested with 'Blindsight' and 'AGL' [1], SARL with MinAtar games[19], and MARL with 4 MeltingPot 2.0 settings—3 with the lowest and 1 with the highest coefficient of variation (CV) from Agapiou[2]:1). These experiments assess whether MAPS can enhance both PR learning and socially intelligent behavior in AIS.

2 Methodology

For PR tasks, we used an auto-encoder for the main task, and a comparator matrix connected to two wagering units for the 2nd-Net, as in [1]. . We used a contrastive loss for the main task, which provided crucial information flow for wagering [5]. For wagering, we used a cross-entropy loss to handle class imbalance. Both 1st and 2nd-Net implemented a cascade model that facilitated a smooth graded accumulation of activation [11]. We empirically chose 50 cascade iterations (except for MARL, given computation constraints). For SARL, we used a DQN framework [15]. We applied convolutional layers, which allowed for reduced computational complexity, an autoencoder, and a replay buffer for learning stability. We then calculated the comparison matrix using the inputs and outputs of the value network's auto-encoder, and connected this to 2 wagering units. For the wagering objective, we calculated the rewards in batches of 128 using an EMA with a smoothing factor of $\alpha = 0.45$. At each step t, a low/high wager was assigned based on whether the last reward was greater than EMA. For MARL, $\alpha = 0.25$. Both were found empirically. For MARL, we used an MAPPO framework[20], convolutional layers, sinusoidal-based relative positional encoding to add positional information, and a Gated Recurrent Unit (GRU) for stability.

3 Results

For Blindsight, suprathreshold patterns were used during training, and 3 types were used for testing. To prevent overfitting, new patterns were generated per epoch. Table 1 compares the proposed model with variants turning the 2nd-Net and/or the cascade model on/off . We observed a performance gain using a 2nd-Net and cascade model, achieving statistical significance compared to the baseline (Z-score: 8.6, 450 seeds). We also observed that gains are mostly driven by the cascade model. For AGL, we pre-trained the model, saved the weights of the 2nd-Net, and disabled backpropagation through it during training. Random strings were used for pre-training, grammar A for training, and a mix of grammar A and grammar B for testing. Grammar strings are defined as per

[13], and we followed the data proportions in [1]. We employed a low training scheme (3 epochs). Results show a statistical significance (Z-score: 15.0 - MAPS, and 4.2 - 2nd-Net).

2nd Net	Cascade	Accuracy	Z-score (Significant)
No	No	0.95 ± 0.03	
No	1st Net	0.97 ± 0.02	**8.50 (Yes)**
Yes	No	0.96 ± 0.03	0.77 (No)
Yes	1st Net	0.97 ± 0.02	**9.01 (Yes)**
Yes	Both	0.97 ± 0.02	**8.6 (Yes)**
No	No	0.54 ± 0.08	
No	1st Net	0.61 ± 0.07	**13.3 (Yes)**
Yes	No	0.57 ± 0.07	**4.2 (Yes)**
Yes	1st Net	0.62 ± 0.07	**15.7 (Yes)**
Yes	Both	0.62 ± 0.06	**15.0 (Yes)**

Table 1: Accuracy for Blindisght (top) and AGL (bottom). Chance level: 0.01 and 0.15.

In MinAtar (table 2), we tested "Seaquest" and "Asterix" for 3 seeds (1 million steps). We show an improvement with MAPS (Z-score: 2.97 and 2.15). For Seaquest, the setting with the most obstacles, we observed that it is when both the cascade model and 2nd-Net are active, that effective learning occurs. In MARL (Table 3, 1.5 million steps), both GRU-only and a 2nd-Net variant were tested. While the 2nd-Net model is slightly superior to GRU, it still lags behind the top model (ACB) presented in [2]. Conversely, for territory inside out, we noted a tendency of MAPS to produce positive outliers (see Appendix D.2), and, over 10 seeds, we found a positive Z-score of 2.59 with respect to the baseline.

2nd Net	Cascade	Rewards	Z-score (Sig.)
No	No	1.21 ± 0.16	
No	1st Net	0.76 ± 0.19	**-2.59 (Yes)**
Yes	No	0.97 ± 0.61	-0.53 (No)
Yes	1st Net	3.06 ± 0.34	**7.03 (Yes)**
Yes	Both	6.15 ± 2.33	**2.97 (Yes)**
No	No	2.49 ± 1.94	
No	1st Net	1.59 ± 0.94	-0.59 (No)
Yes	No	5.48 ± 1.30	1.81 (No)
Yes	1st Net	4.54 ± 1.01	1.32 (No)
Yes	Both	5.77 ± 0.94	**2.15 (Yes)**

Table 2: Validation rewards: Seaquest (top) and Asterix (bottom). Chance level: 0.09 and 0.47.

Environment	GRU	GRU (2nd-Net)	ACB
Harvest C.	18.9 ± 1.4	20.6 ± 2.1	32.8 ± 10.6
Harvest P.	28.1 ± 1.9	28.7 ± 3.8	31.9 ± 11.0
Chem. 3D.	$\mathbf{1.2 \pm 0.1}$	$\mathbf{1.2 \pm 0.1}$	1.1 ± 0.8
Terr. I.O.	63.5 ± 8.7	76.5 ± 8.3	80.3 ± 48.0

Table 3: Training rewards in MARL.

4 Conclusion

This study demonstrates that the MAPS architecture enhances learning across PR, SARL, and MARL tasks. In PR and SARL, combining the cascade model and 2nd-Net consistently improved performance, particularly in complex environments. Even without backpropagation through the 2nd-Net, MAPS maintained strong results in AGL, highlighting its robustness.

In MARL, while MAPS did not outperform the top benchmark, it matched or exceeded baselines in most cases and showed strong performance in high-variability scenarios like “Territory Inside Out.” These findings show the value of metacognitive architectures for building more adaptive and socially aware AI systems.

5 Acknowledgements

This study was supported by the Institute for Advanced Consciousness Studies (IACS), the Institute for Data Valorization, Montreal (IVADO; CF00137433 & PRF3), and the Canadian Institute for Advanced Research (CIFAR). Computation was enabled by Calcul Québec (www.calculquebec.ca) and Digital Research Alliance of Canada (www.alliancecan.ca). JDV would like to thank the UNIQUE center for travel funding to attend the AAAI conference. GD was supported by the Fonds de recherche du Québec - Santé (FRQ-S; 2024-2025 - CB - 350516), Natural Sciences and Engineering Research Council of Canada (NSERC; DGECR-2023-00089), the Brain Canada Foundation (2022 Future Leaders in Brain Research).

References

[1] B. Timmermans A. Pasquali and A. Cleeremans. Know thyself: Metacognitive networks and measures of consciousness. *Cognition*, 117:182–190, 2010.

[2] J.P. Agapiou, A.S. Vezhnevets, E.A. Duéñez-Guzmán, J. Matyas, Y. Mao, P. Sunehag, R. Köster, U. Madhushani, K. Kopparapu, R. Comanescu, D.J. Strouse, M.B. Johanson, S. Singh, J. Haas, I. Mordatch, D. Mobbs, and J.Z. Leibo. Melting pot 2.0. *arXiv preprint arXiv:2211.13746*, 2023.

[3] Federica Bianco and Ilaria Castelli. The promotion of mature theory of mind skills in educational settings: a mini-review. *Frontiers in Psychology*, 14:1197328, 2023.

[4] S. Bolotta and G. Dumas. Social neuro ai: Social interaction as the 'dark matter' of ai. *Frontiers in Computer Science*, 4, 2022.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. ICML 2020.

[6] Brendan Conway-Smith and Robert L. West. Toward autonomy: Metacognitive learning for enhanced ai performance. *Proceedings of the AAAI 2024 Spring Symposium Series: Symposium on Human-Like Learning*, 2024.

[7] Zoltan Dienes, Gerry Altmann, Lisa Kwan, and Andrew Goode. Unconscious knowledge of artificial grammars is applied strategically. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, 21(5):1322–1338, 1995.

[8] Michael SA Graziano. *Consciousness and the social brain*. Oxford University Press, 2013.

[9] Ryota Kanai, Ryota Takatsuki, and Ippei Fujisawa. Meta-representations as representations of processes. *PsyArXiv*, 2024. Preprint.

[10] Emily L Long, Caroline Catmur, Stephen M Fleming, and Geoffrey Bird. Metacognition facilitates theory of mind through optimal weighting of trait inferences. *Cognition*, 256:106042, 2025.

[11] James L. McClelland, David E. Rumelhart, Jerome Feldman, and Patrick Hayes. *Explorations in Parallel Distributed Processing - Macintosh version: A Handbook of Models, Programs, and Exercises*. Bradford Books. The MIT Press, 1989. Includes 2 diskettes for the Macintosh, In Special Collection: CogNet.

[12] Kamal K Ndousse, Douglas Eck, Sergey Levine, and Natasha Jaques. Emergent social learning via multi-agent reinforcement learning. In *International conference on machine learning*, pages 7991–8004. PMLR, 2021.

[13] N. Persaud, P. McLeod, and A. Cowey. Post-decision wagering objectively measures awareness. *Nature Neuroscience*, 10:257–261, 2007.

[14] Kristian Sandberg, Bert Timmermans, Morten Overgaard, and Axel Cleeremans. Measuring consciousness: Is one measure better than the other? *Consciousness and Cognition*, 2010.

[15] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. *arXiv preprint arXiv:1509.06461*, 2015. AAAI 2016.

[16] Bamicha Victoria and Drigas Athanasios. Theory of mind in relation to metacognition and icts. a metacognitive approach to tom. *Scientific Electronic Archives*, 16(4), 2023.

[17] Jessica Williams, Stephen M Fiore, and Florian Jentsch. Supporting artificial social intelligence with theory of mind. *Frontiers in artificial intelligence*, 5:750763, 2022.

[18] Alan FT Winfield. Experiments in artificial theory of mind: From safety to story-telling. *Frontiers in Robotics and AI*, 5:357467, 2018.

[19] Kenny Young and Tian Tian. Minatar: An atari-inspired testbed for thorough and reproducible reinforcement learning experiments. *arXiv preprint arXiv:1903.03176*, 2019.

[20] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu. The surprising effectiveness of ppo in cooperative multi-agent games. In *Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[21] E. Zaroukian. Theory of mind and metareasoning for artificial intelligence: A review. 2022. Submitted for review.

A Appendix / supplemental material

Appendix A - Additional Environment details

Appendix A.1 - Blindsight task

Blindsight is a neurological phenomenon where individuals with damage to their primary visual cortex can still respond to visual stimuli without consciously perceiving them.

To study this, we use a simulated dataset that mimics the conditions of blindsight according to [1]. This dataset contains 400 patterns, equally split between two types:

- **Random noise patterns:** These consist of low activations ranging between 0.0 and 0.02.
- **Designed stimulus patterns:** Each pattern includes one unit that shows a higher activation level, varying between 0.0 and 1.0.

This dataset allows us to test hypotheses concerning how sensory processing and network responses adapt under different conditions of visual impairment.

We have three main testing scenarios, each designed to alter the signal-to-noise ratio to simulate different levels of visual impairment:

- **Suprathreshold stimulus condition:** Here, the network is tested against familiar patterns used during training to assess its response to known stimuli.
- **Subthreshold stimulus condition:** This condition slightly increases the noise level,

akin to actual blindsight conditions, testing the network's capability to discern subtle signals.

- **Low vision condition:** The intensity of stimuli is decreased to evaluate how well the network performs with significantly reduced sensory input.

Appendix A.2 - Artificial Grammar Learning Task

In the AGL experiment, Persaud et al. [13] demonstrate that participants exposed incidentally to letter strings generated by an artificial grammar perform better than chance on a subsequent, unexpected test where they distinguish between new grammatical and non-grammatical strings. However, they fail to optimize their earnings through wagering. Once participants were informed about the grammar rules, they began to place advantageous wagers (explicit condition) [1].

To simulate this, we utilize artificially generated strings ranging from 3 to 8 letters, classified into three types: randomly generated, grammar A, and grammar B, as defined by Persaud et al.

During training, the networks are exposed to two conditions: explicit and implicit, reflecting the results of implicit learning [7]. For the implicit condition (low consciousness), networks are trained for 3 epochs, while for the explicit condition (high consciousness), they are trained for 12 epochs.

Appendix A.3 - MinAtar

MinAtar provides simplified versions of classic Atari 2600 games, designed specifically for AI agent testing and development. MinAtar offers more accessible and computationally efficient environments for AI research and experimentation [19]. There are 5 Atari games implemented:

- **Space Invaders:** The player controls a cannon to shoot at aliens that move across and down the screen, with each destroyed alien providing +1 reward and causing the remaining aliens to speed up. Aliens also shoot back at the player, new waves spawn at increased speeds after clearing a wave, and termination occurs when the player is hit by an alien or bullet [19].
- **Breakout:** The player controls a paddle at the bottom of the screen to bounce a diagonally-traveling ball toward three rows of bricks at the top, earning +1 reward for each brick broken and getting new rows when all are cleared. The ball's direction changes based on which side of the paddle it hits or when it contacts walls and bricks, with game termination occurring when the ball reaches the bottom of the screen [19].
- **Seaquest:** The player controls a submarine that can fire bullets at enemy submarines and fish, earning +1 reward for each hit while also rescuing divers to fill a progress bar and maintaining oxygen that depletes over time. Oxygen replenishes when surfacing with at least one rescued diver, surfacing with six divers provides additional rewards based on remaining oxygen, and the game ends when hit by enemies, running out of oxygen, or surfacing without divers [19].
- **Asterix:** The player moves freely in four cardinal directions to collect treasure while avoiding enemies that spawn from the sides, with each treasure providing a +1 reward and enemy contact causing termination. Enemy and treasure movements are indicated by trail channels, and the game's difficulty increases periodically by enhancing the speed and spawn rate of both enemies and treasures [19].
- **Freeway:** The player moves vertically up and down at a restricted pace (once every 3 frames) to cross a road filled with

horizontally-moving cars, earning +1 reward upon reaching the top before being returned to the bottom. When hit by a car, the player returns to the bottom without penalty, car speeds randomize after each successful crossing, and the game terminates after 2500 frames have elapsed [19].

Appendix A.4 - Meltingpot

The Melting Pot Suite provides a comprehensive framework for generating test scenarios that assess an agent population's ability to generalize cooperative behavior in new situations. It offers up to 50 distinct training and testing environments. The test scenarios combine novel background populations of agents and include a variety of substrates, such as classic social dilemmas like the Prisoner's Dilemma, as well as complex mixed-motive coordination games. In our experiments, we selected four environments based on the coefficient of variation among the models tested in [2]. This value was calculated for the 37 non-zero-sum environments out of the 50 available (see Figure 1). We chose the three environments with the lowest variability and the environment with the highest positive variability.

Our tested environments are: Commons Harvest Closed, Commons Harvest Partnership, Chemistry Three Metabolic Cycles with Plentiful Distractors, and Territory Inside Out. A short description is provided below:

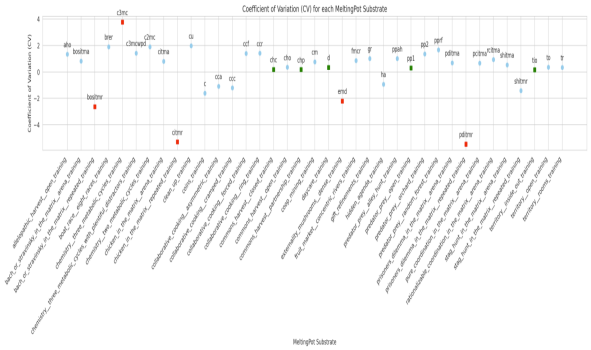

Figure 1: Variability among Melting Pot environments according to the experimentation in [2].

- **Commons Harvest Closed:** Apples are dispersed and can be consumed by agents. Additionally, apples have a probability at every step to regrow, which depends on the number of nearby apples: 0.0025 when there are three or more apples, 0.005 for two, 0.001 if there is one, and 0 otherwise. Thus, agents need to exercise restraint in consuming all apples in a batch to ensure the long-term regrowth of apples. Even though it is not beneficial to consume the last apple, agents are incentivized to do so to prevent other agents from consuming it. In this closed variant, there are rooms full of apples, promoting agents to defend them and minimize the probability of other agents harvesting the full patch of apples [2].
- **Commons Harvest Partnership:** Similar to the Commons Harvest Closed environment, this variant still has rooms filled with apples. However, it requires two agents to protect a room, thus promoting the development of cooperative behavior and a mutually sustainable situation[2].
- **Chemistry Three Metabolic Cycles with Plentiful Distractors:** In this setting, a set of agents work to generate mutual benefits from metabolic reactions defined by a predefined graph. These reactions occur stochastically when reactants are in close proximity to one another. Agents can carry molecules and are rewarded when the molecule in their inventory is part of a reaction, either as a reactant or a product. In the three metabolic cycles variant, agents benefit from three different cycles, which continue as long as the minimum energy requirements are fulfilled. Agents must

learn to facilitate the right reactions to generate enough energy to sustain the cycles. The environment also contains distractors, which are molecules that do not provoke reactions but provide a small constant reward to encourage agents to pursue less rewarding strategies[2].

- **Territory Inside Out:** Each agent is assigned a unique color and seeks to claim territory by painting walls in that color. Wet paint does not yield rewards. After 25 steps following the application of paint, if no further paint has been added, the paint dries and turns into a brighter shade of the agent's color. Once dry, the painted wall rewards the claiming player at a consistent rate. The more walls a player claims, the higher their expected rewards per timestep. In the Inside Out variant, agents are generated in a maze and must move inward toward the center of the map to claim territory. In this scenario, agents can zap each other, immobilizing the other agent for a set number of steps. An agent that is zapped twice is eliminated[2].

Appendix B - Hyperparameter choices and Computational resources

Appendix B.1 - Blindsight task

For the blindisight task, we used an Nvidia RTX3070 GPU for training, with 8GB of RAM. The training time was maximum for MAPS (2nd order network and cascade model in both 1st and 2nd order networks). For this setting, training over the 450 seeds took roughly 12 hours.

Hyperparameter	**Value**
Input size	100
Output size	100
Hidden size	60
lr first order	0.5
lr second order	0.1
Temperature	1.0
Step size	25
Gamma	0.98
Epochs number for training	200
Optimizer	$Adamax$
Cascade iterations	50

Table 4: Hyperparameters used for the Blindsight Task.

Appendix B.2 - Artificial Grammar Learning Task

bgFor the AGL task, we used an Nvidia RTX 3070 GPU for training, with 8GB of RAM. The training time was maximum for MAPS (2nd order network and cascade model in both 1st and 2nd order networks). For this setting, training over the 450 seeds took roughly 12 hours.

Hyperparameter	**Value**
Input size	48
Output size	48
Hidden size	40
lr first order	0.4
lr second order	0.1
Temperature	1.0
Step size	1
Gamma	0.999
Epochs number for pre-training	60
Epochs number for training(high consciousness)	12
Epochs number for training(low consciousness)	3
Optimizer	$RangerVA$
Cascade iterations	50

Table 5: Hyperparameters used for the Artificial Grammar Learning Task.

Appendix B.3 - MinAtar

For the MinAtar environments, we used a GPU V100 for training. The training time was maximum for MAPS (2nd order network and cascade model in both 1st and 2nd order networks). For this setting, training took roughly 6 days per million steps per seed.

Hyperparameter	**Value**
Batch size	128
Replay buffer size	$100,000$
Target network update frequency	$1,000$
Training frequency	1
Number of frames	$500,000$
First N frames	$100,000$
Replay start size	$5,000$
End epsilon	0.1
Step size	0.0003
Step size (second order)	0.0002
Gradient momentum	0.95
Squared gradient momentum	0.95
Minimum squared gradient	0.01
Gamma	0.999
Step Size	1
Epsilon	1.0
Alpha	0.45
Cascade iterations	50
Optimizer	*Adam*

Table 6: Hyperparameters used for the MinAtar experiments.

Appendix B.4 - Meltingpot

For the melting pot tasks, we used an Nvidia A100 GPU for training. The average training time was roughly 16 hours per seed(baseline, MAPS not implemented fully, only with simple 2nd order network with no cascade model due to limitations with computational resources). Every run required roughly 4-6 GB of RAM, mainly depending on the number of agents.

Hyperparameter	**Value**
Num agents (harvest closed)	6
Num agents (harvest partnership)	4
Num agents (chemistry)	8
Num agents (territory)	5
Hidden size	100
Actor lr	$7e-5$
Critic lr	100
Num env steps	$15e6$
Entropy coef	0.01
Clip param	0.2
Weight decay	$1e-5$
PPO epoch	15
Optimizer	*Adam*

Table 7: Common hyperparameters used for the Meltingpot environments.

Appendix C - Architectures

Appendix C.1 - Blindsight task and Artificial Grammar Learning Task

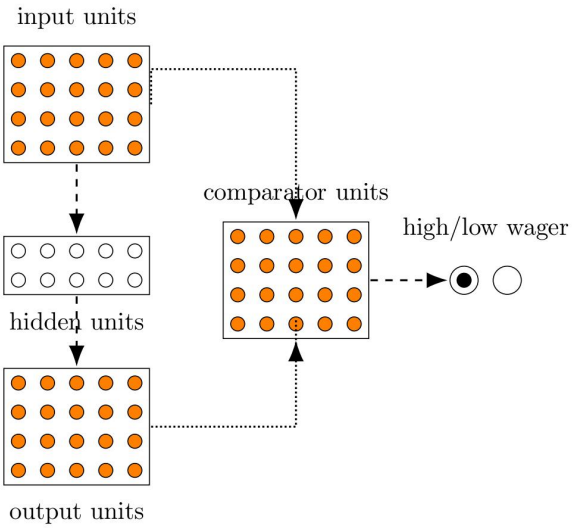

Figure 2: Illustration of the architecture used for both the Blindsight and Artificial Grammar Learning tasks.

Appendix C.2 - Meltingpot

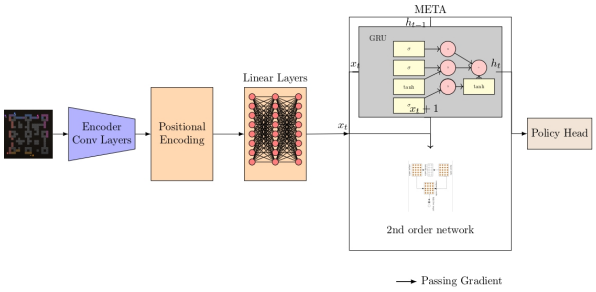

Figure 3: Illustration of the architecture used for all the Meltingpot environments

Appendix D - Additional results

Appendix D.1 - MinAtar

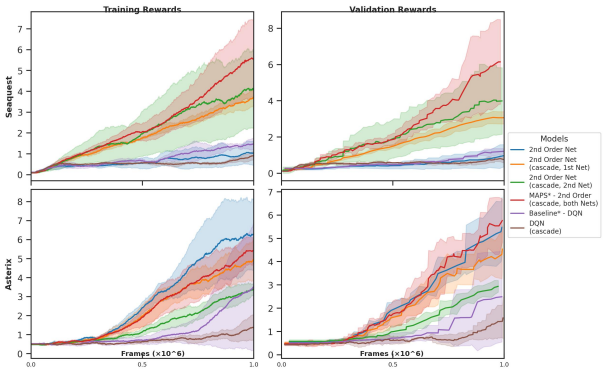

Figure 4: Training (left) and validation rewards (right) plots for SARL.

Appendix D.2 - Meltingpot

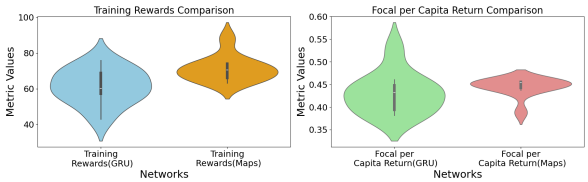

Figure 5: Territory Inside Out Results (10 seeds). Violin plot for avg. rewards (left); and Focal per Capita Return (right). Focal per capita return is a fairness measure (i.e. equal to 1.0 when all agents receive equal rewards), as defined by [2]

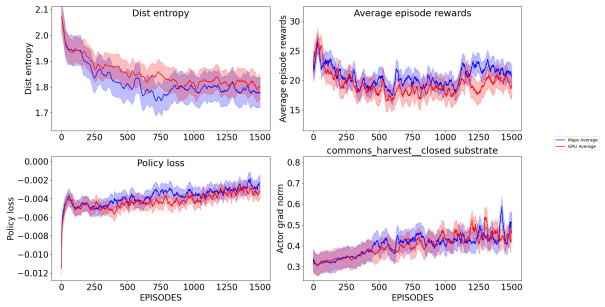

Figure 6: Results per episode over 1.5 million steps for commons harvest closed environment. To the top left the evaluation parameter is dist entropy, which represents the action distribution entropy, where a lower value points to a lower overall stochastic behaviour of the agents. The top right represents the average reward of all agents, where a higher value is desired. Bottom left is the policy loss, and bottom right is the actor gradient norm.

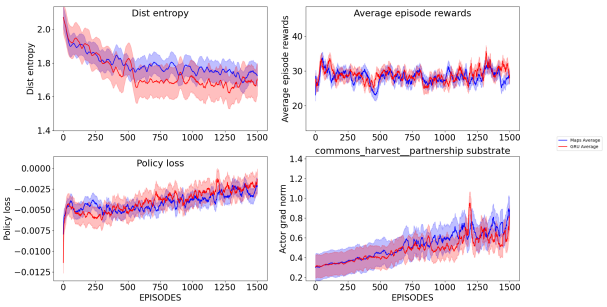

Figure 7: Results per episode over 1.5 million steps for commons harvest partnership environment. To the top left the evaluation parameter is dist entropy, which represents the action distribution entropy, where a lower value points to a lower overall stochastic behaviour of the agents. The top right represents the average reward of all agents, where a higher value is desired. Bottom left is the policy loss, and bottom right is the actor gradient norm.

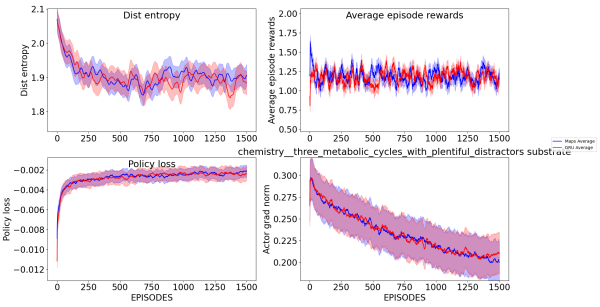

Figure 8: Results per episode over 1.5 million steps for chemistry environment. To the top left the evaluation parameter is dist entropy, which represents the action distribution entropy, where a lower value points to a lower overall stochastic behaviour of the agents. The top right represents the average reward of all agents, where a higher value is desired. Bottom left is the policy loss, and bottom right is the actor gradient norm.

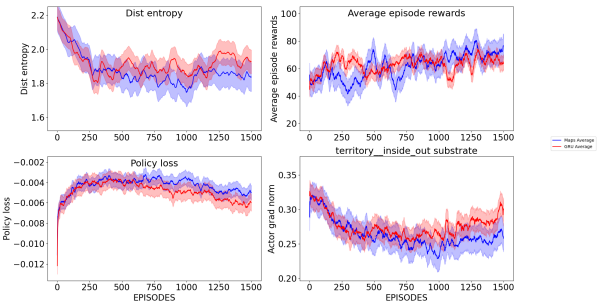

Figure 9: Results per episode over 1.5 million steps for territory inside out environment. To the top left the evaluation parameter is dist entropy, which represents the action distribution entropy, where a lower value points to a lower overall stochastic behaviour of the agents. The top right represents the average reward of all agents, where a higher value is desired. Bottom left is the policy loss, and bottom right is the actor gradient norm.