Learning Object-Oriented Dynamics for Planning from Text

Guiliang Liu, Ashutosh Adhikari, Amir Farahmand, Pascal Poupart
University of Waterloo, University of Toronto & Vector Institute

ICLR 2022 Presentation

Introduction

Texting-Based Games

Learning Object-Oriented Dynamics for Planning from Text

Introduction

Model-base RL

Learning Object-Oriented Dynamics for Planning from Text

Dynamics: transition function $p(s'|s, a)$ and reward function $\mathrm{r}(s, a)$.

- Difficult → in high-dim space, open ignored by model-free RL.
- Important → generalization ability, sample efficiency

Planning: algorithms like MCTS, Dyna-Q.

- Robust and well-perform

Introduction

Factorized states capturing only the object information

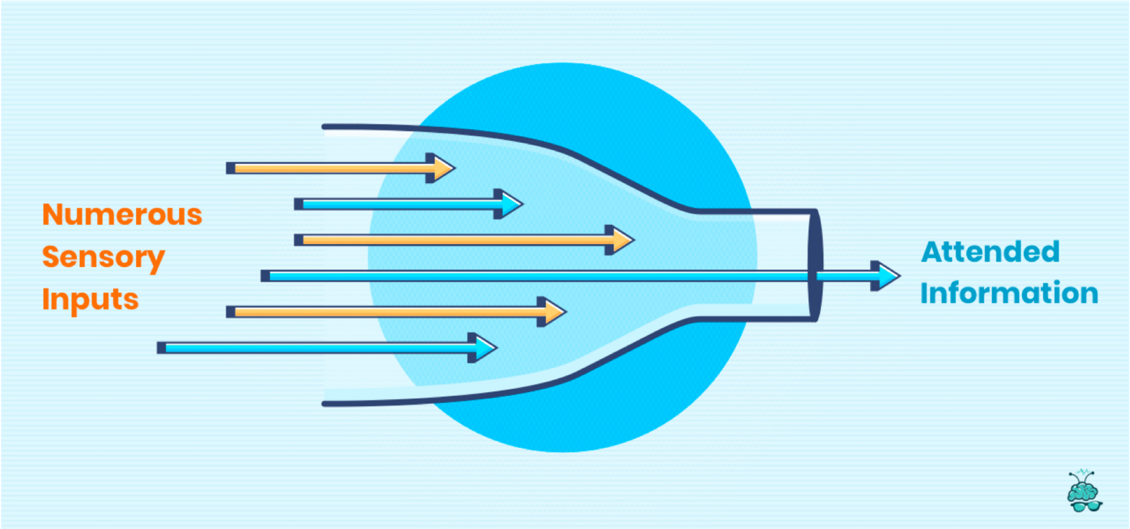
Learning Object-Oriented Dynamics for Planning from Text

Why object information ?

a) Modelling the all the language information is too complex.

b) In a sentence, only objects and their relations matters for a task.

c) Object-oriented information bottleneck (Tishby et al., 2000)



Object-Oriented Partially Observable Markov Decision Process

OO-POMDP is a tuple $< \mathcal{S}, \mathcal{O}, \mathcal{Z}, \Phi, \mathcal{G}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \gamma >$, where:

- $\mathcal{S}$ and $\mathcal{O}$: **low-level** states and observations from the TGB.

> You open the copy of “cooking: a modern approach (3rd ed.)” and start reading: recipe # 1 --------- gather all following **ingredients** and follow the directions to prepare this tasty **meal**. ingredients: **banana**, block of **cheese**, **carrot** directions: **dice** the **banana**, **fry** the **banana**, **chop** the block of **cheese**, **roast** the block of **cheese**, **slice** the **carrot**, **fry** the **carrot**, and prepare **meal**.

$$[o_0 \rightarrow \dots \dots \rightarrow o_{t-2} \longrightarrow o_{t-1} \longrightarrow o_t] \Longrightarrow s_t$$

Object-Oriented Partially Observable Markov Decision Process

OO-POMDP is a tuple $< \mathcal{S}, \mathcal{O}, \mathcal{Z}, \Phi, \mathcal{G}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \gamma >$, where:

- $\mathcal{S}$ and $\mathcal{O}$: **low-level** states and observations from the TGB.

> You open the copy of “cooking: a modern approach (3rd ed.)” and start reading: recipe # 1 --------- gather all following **ingredients** and follow the directions to prepare this tasty **meal**. ingredients: **banana**, block of **cheese**, **carrot** directions: **dice** the **banana**, **fry** the **banana**, **chop** the block of **cheese**, **roast** the block of **cheese**, **slice** the **carrot**, **fry** the **carrot**, and prepare **meal**.

$$[o_0 \longrightarrow \dots\dots \longrightarrow o_{t-2} \longrightarrow o_{t-1} \longrightarrow o_t] \Longrightarrow s_t$$

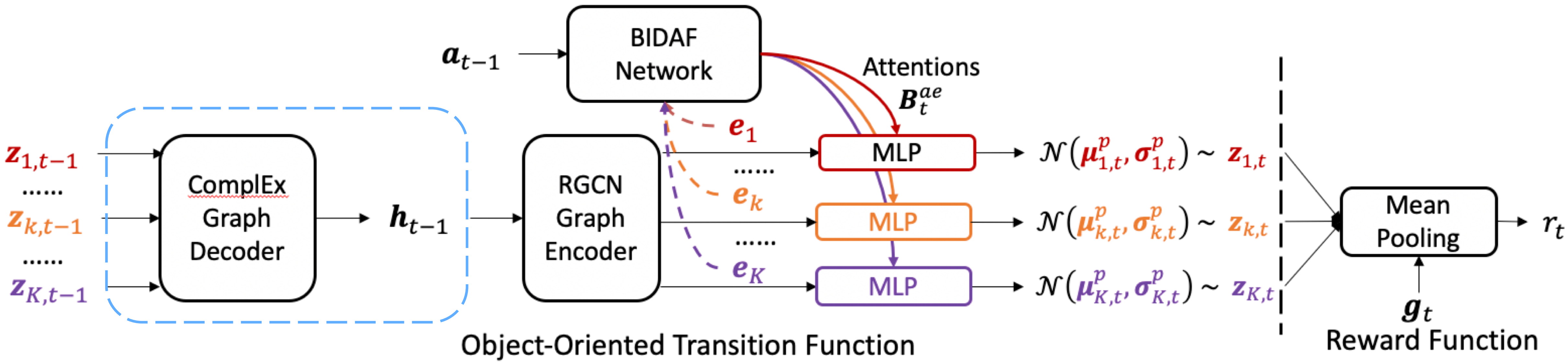
- $\mathcal{Z}$ and Φ: **object-level** states and observations.

ϕ_t [banana , is , ingredients] $\Longrightarrow [\phi_1, \dots, \phi_t] \Longrightarrow$

is → **ingredients**
↑
dice ← **need** ← **banana** → **need** → **fry**
↓
table ← **on**

$\Longrightarrow$ latent representation (e.g., word embeddings) $\Longrightarrow$ $\boldsymbol{z}_{t,\mathbf{b}\dots}$

Object-Oriented Transition Model

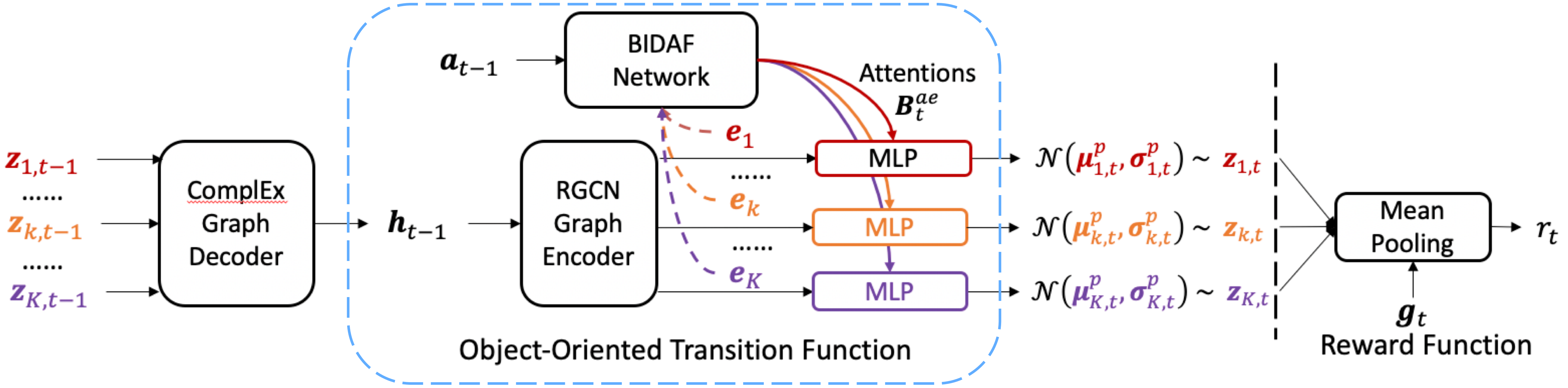


ComplEx Graph Decoder (Motivation**:** graph as a structured information bottleneck**)**:

- Map $\boldsymbol{z}_{t-1} = [z_{1,t-1}, \ldots, z_{K,t-1}]$ (states of K objects) to a graph $\boldsymbol{h}_{t-1}$.
- Apply a ComplEx scoring (Trouillon et al., 2016) function for link prediction.
- Approximate matrix prediction with low-rank decomposition:

$$h_{t-1} = [Re(\boldsymbol{Z}_{t-1}\boldsymbol{W}_1\boldsymbol{Z}_{t-1}^T), \ldots, Re(\boldsymbol{Z}_{t-1}\boldsymbol{W}_C\boldsymbol{Z}_{t-1}^T)] \qquad \boldsymbol{Z}_t \in \mathbb{R}^{K\times E} \qquad W_c \in \mathbb{C}^{E\times E}$$

Object-Oriented Transition Model



Independent Transition Layers (lots of objects, action affect only several objects):

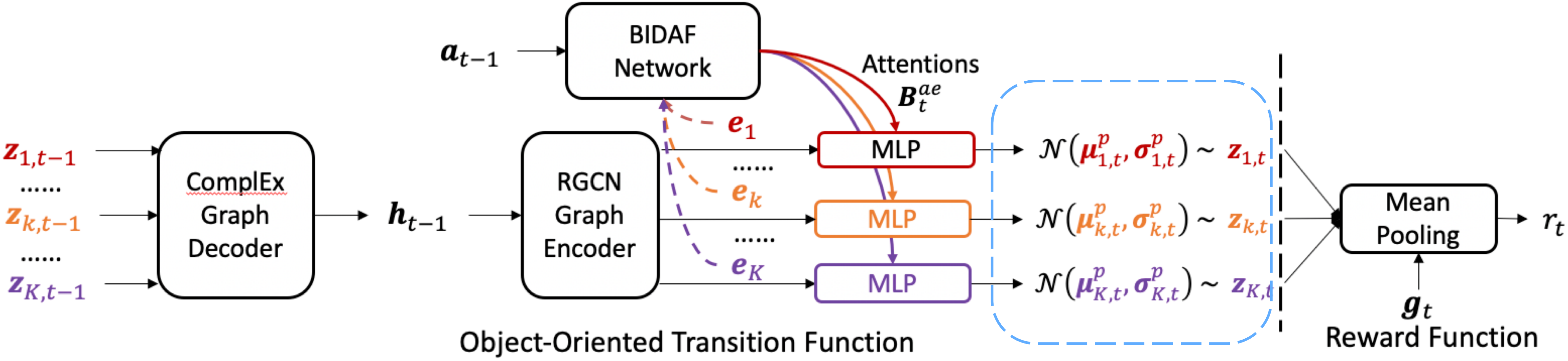
- (BIDAF) (Seo et al., 2017) detected affected actions.

$$\boldsymbol{\nu}_{k,t-1}^{a} = \sum_j b_{k,j}^{a} \psi^{a}(a_{j,t-1}) \quad \text{where} \quad \boldsymbol{b}_k^{a} = softmax(\boldsymbol{B}_{k,:}^{ae}) \in [0,1]^J,$$

$$\boldsymbol{\nu}_{t-1}^{e} = \sum_k b_k^{e} \boldsymbol{e}_{k,t-1} \quad \text{where} \quad \boldsymbol{b}^{e} = softmax(\max_{col}(\boldsymbol{B}^{ae})) \in [0,1]^K,$$

$v_{k,t-1}^{\alpha}$ is the attended action vector, v_{t-1}^{e} is the attended object representation.

Object-Oriented Transition Model



Independent Transition Layers (lots of objects, action affect only several objects):

- A group of **independent** transition layers to predict the belief of objects states (inspired by the Independent Causal Mechanism (IRM) (Pearl, 2009)).

$$p_T^k(z_{k,t}|\boldsymbol{a}_{t-1}, \boldsymbol{z}_{t-1}) = \mathcal{N}(\boldsymbol{\mu}_{k,t}, \boldsymbol{\sigma}_{k,t}) \quad \text{where} \quad [\boldsymbol{\mu}_{k,t}, \boldsymbol{\sigma}_{k,t}] = \psi_k^p([\boldsymbol{\nu}^a_{k,t-1}, \boldsymbol{\nu}^e_{t-1}, \boldsymbol{e}_{k,t-1}])$$

Controlling Performance with Planning

Experiment Setting:

- Text-World benchmark.
- 100/20/20 training /validation/testing games.
- Difficulty level 0-5.

Level	Recipe Size	#Locations	Max Scores	Need Cut	Need Cook	#Action Candidates	#Objects
0	1	1	3	✗	✗	10.5	15.4
1	1	1	4	✓	✗	11.5	17.1
2	1	1	5	✓	✓	11.8	17.5
3	1	9	3	✗	✗	7.2	34.1
4	3	6	11	✓	✓	28.4	33.4
5	Mixture of Levels[1,2,3,4]						

Type	Model	0	1	2	3	4	5	↑
Model-Free Algorithm	DQN	90.0	62.5	32.0	38.3	17.7	34.6	0
	DRQN	95.0	58.8	31.0	36.7	21.4	27.4	-0.8
	DRQN+	95.0	58.8	33.0	33.3	19.5	30.6	-0.8
	KG-A2C	96.7	55.5	31.0	54.3	26.8	30.1	+3.2
	GATA-GTP	95.0	62.5	32.0	51.7	21.8	23.5	+1.9
	GATA-OG	100	66.2	36.0	58.3	14.1	45.0	+7.4
	GATA-COC	96.7	62.5	33.0	46.7	25.9	33.4	+3.9
Model-Based Planning	OOTD learned by the Object-Supervised (OS) ELBo Objective							
	OS-Dyna-Q	100	62.5	42.0	58.3	21.8	48.2	+9.6
	OS-MCTS	95.0	77.5	56.0	63.3	24.9	42.9	+14.1
	OS-Dyna-Q + MCTS	95.0	78.8	**57.0**	71.7	27.7	38.1	+15.5
	OOTD learned by the Self-Supervised (SS) ELBo Objective							
	SS-Dyna-Q	100	62.5	48.0	53.3	30.5	47.0	+11.0
	SS-MCTS	100	70.0	51.0	70.0	27.3	54.4	+16.3
	SS-Dyna-Q + MCTS	100	**81.3**	56.9	**75.0**	**31.4**	**58.4**	**+21.3**

Sample Efficiency

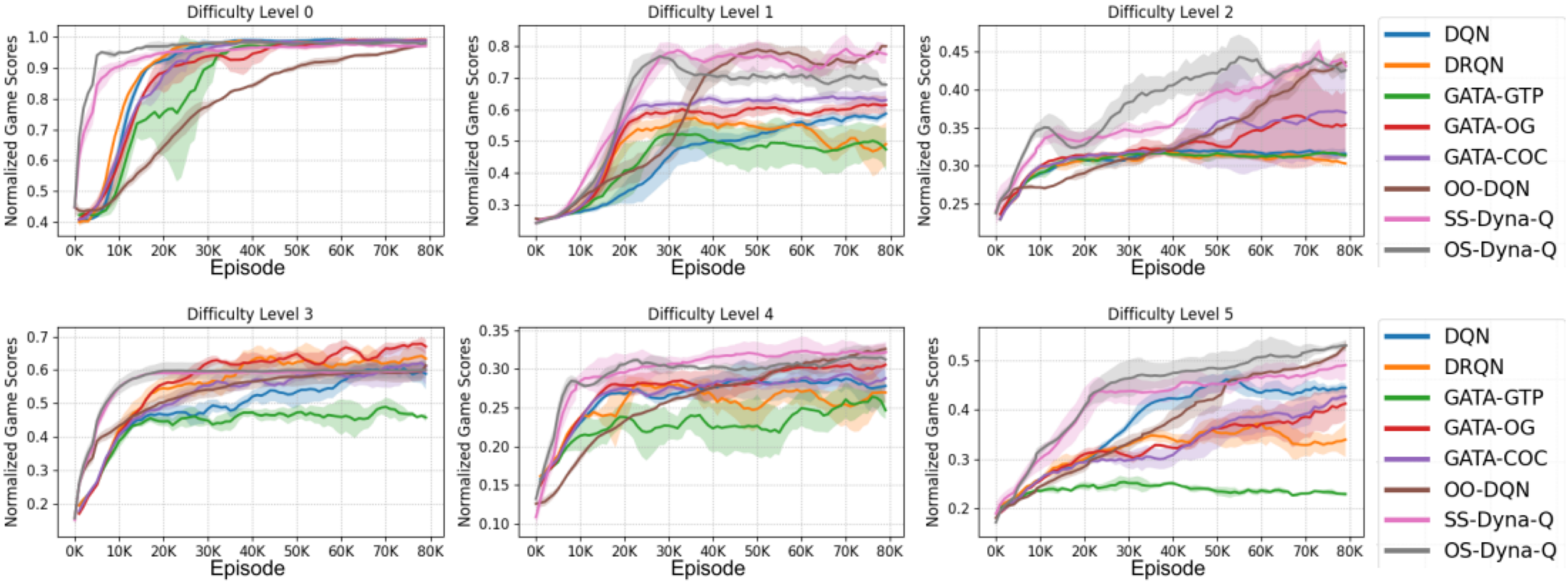


Figure 3: Training Curves: Agents' normalized scores for the games at different difficulty levels. The plot shows $mean \pm std$ normalized scores computed with three independent runs.

Question and Answering (Q&A)