

**NOVEL TECHNIQUES FOR TRANSFER
IN STRUCTURED MODELS FOR
REASONING, PLANNING AND
REINFORCEMENT LEARNING**

VISHAL SHARMA



**DEPARTMENT OF COMPUTER SCIENCE &
ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY DELHI
JULY 2024**

©Indian Institute of Technology Delhi - 2024
All rights reserved.

NOVEL TECHNIQUES FOR TRANSFER IN STRUCTURED MODELS FOR REASONING, PLANNING AND REINFORCEMENT LEARNING

by

VISHAL SHARMA

Department of Computer Science and Engineering

Submitted

in fulfillment of the requirements of the degree of Doctor of Philosophy

to the



INDIAN INSTITUTE OF TECHNOLOGY DELHI

July 2024

Certificate

This is to certify that the thesis titled **Novel Techniques for Transfer in Structured Models for Reasoning, Planning and Reinforcement Learning** being submitted by **Mr Vishal Sharma** for the award of **Doctor of Philosophy** in Department of Computer Science and Engineering is a record of bonafide work carried out by him under my guidance and supervision at the **Department of Computer Science and Engineering, Indian Institute of Technology Delhi**. The work presented in this thesis has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma unless otherwise stated explicitly. In particular, work done in chapter 3 and chapter 6 was done jointly with undergraduate students. Work done in chapter 4 and chapter 5 was done with a master's student. In each case, the part done by the collaborators appeared in their respective bachelor's or master's thesis.



Parag Singla

Professor

Department of Computer Science and Engg.

Indian Institute of Technology Delhi

New Delhi- 110016

Dedication(s)

This thesis is dedicated to

*The mother of all, **Goddess Maa Durga**, besides whom there is no one else*

*The destroyer of all, **Goddess Maa Chamunda**, the one who resides in Lahat*

*My grandfather, **Late Shri Neel Kanth**, the one who fought for India's freedom*

*My grandmother, **Late Shri Lalita Devi**, the one who always loved me*

*My mother, **Sunita Sharma**, the one who is the basis of my life*

*My father, **Kuldeep Sharma**, the one on whose shoulders I stand*

*My sister, **Priya Sharma**, the one who is my guiding light*

Acknowledgements

I start by expressing my deepest gratitude to my supervisor, Prof. Parag Singla, whose expertise, guidance, and unwavering support have been instrumental in the completion of my PhD. I find it difficult to explicitly list all the things I absorbed while working with him so closely, but I will still give it a shot. His insightful feedback and encouragement at every step of the process have profoundly shaped my research, pushing me to reach new heights and expand my academic horizons. His dedication to fostering an environment of intellectual and mathematical rigour has enriched not only this work but also my growth as a scholar. His insistence on perseverance until the very last moment has instilled in me the importance of tenacity and determination. Furthermore, my experience as a Teaching Assistant for many of his courses provided me with a deep understanding of the intricate details of academia. Through these experiences, I have gained invaluable insights into both teaching and research, for which I am immensely grateful.

Next, I would like to extend my heartfelt thanks to Prof. Mausam for his invaluable mentorship as my unofficial supervisor during the second half of my PhD. His critical and expert inputs on planning and reinforcement learning became key ingredients for the core of this thesis. His honest and direct feedback served as a powerful tool for overcoming cognitive bottlenecks, enabling me to reach new levels in my thinking process.

I was very fortunate to collaborate with many esteemed professors during my PhD, including Prof. Vibhav Gogate (UT Dallas), Prof. Guy Van Den Broeck (UCLA), and Florian Geißer. Their research contributions were crucial in strengthening this work. I also thank my SRC members (Prof. Naveen, Prof. Mausam, and Prof. Sumeet) for their key insights throughout my PhD. Additionally, I had the privilege of collaborating with some of the brightest student minds in India: Daman Arora, Prayushi Faldu, Yash Jain, Mohit Gupta, Noman Ahmed Sheikh, Pranav Pratap Singh, Jayant Reddy, Shobhit Gupta, Gobind Singh, Siddhant Mago, Rushil Gupta, Aniket Gupta, and Sanket Gandhi. While I learned from all of them, I would like to extend special thanks to Noman and Daman for our intense technical discussions, late-night work sessions, casual conversations, unwavering companionship and mutual love for tea. Their camaraderie made the journey enjoyable and memorable.

I was fortunate to have made many friends in the lab, hostel, and department. The list is endless, but I would like to start with Shailja Pandey, who supported me during tough times and listened to my daily (sometimes hourly) complaints, especially when I switched my research area. Her guidance and encouragement were instrumental in my decision to continue my PhD. Our mutual hardships were made bearable over samosas, cups of tea, and poha. Among my seniors, I have made lifelong friends: Dinesh Khandelwal, Ankit Anand, Happy Mittal, Rajesh Kedia, and Dilpreet. Long discussions with Dinesh and Ankit were incredibly valuable in navigating the ups and downs of PhD life. Among my batchmates, I would like to thank Kunal, Sanjana, and Madhukar for our chai-time chit-chats. Among my juniors, I thank Vipul, Anshul and Shivansh for their endless questions about PhD and life in general; answering those always made me feel superior to them. Some of my great friends come from my hostel, Nilgiri: Rijul, Jagdeep, Vishal, Anjali and Negi Ji. I will miss the morning ritual of banging on Rijul's door for breakfast and the intense political planning on hostel student positions and room allotments. I also thank Arnab Kumar Mondal for his invaluable support in thesis writing while we worked as colleagues at Fujitsu.

I extend my heartfelt gratitude to the dedicated staff of the CSE department: Rekha Ma'am, Vandana Ma'am, Manju Ma'am, Aditi Ma'am, Hemant Bhaiya, Suresh Ji, Rajesh Ji and Arun Ji. Their unwavering support has been important for my long journey. I fondly remember late Abhishek Sharma, whose casual talks and cricket tournaments brought joy and camaraderie into my life. I express my deep appreciation to IIT Delhi for providing an inspiring environment to pursue my PhD, and to the MHRD fellowship for its crucial support during the initial years. A sincere thanks also goes to TCS for the research scholar fellowship. I also extend my thanks to Aggarwal Poha, Thaapa's chai, and Aunt's chai for providing a daily dose of happiness throughout these years. I would like to thank the Punjabi music industry, specifically Late Sidhu Moose Wala for pulling me up in low times.

Next, I thank my parents, Sunita Sharma and Kuldeep Sharma, for bearing with me throughout my life and ensuring I got the best I could get at every point of my life. Their complete support and presence in my life are all I could have asked for. I also thank my sister, Priya Sharma, and brother-in-law, Devraj, for their love and support. The presence of my *Nani ji* constantly reminded me that independent of what I do in life, I will still be loved. I also thank Lovey, Deven, and their daughter Ishi for their support in a particularly tough phase of my life. I also thank Shashank Jain for not doing much but just being there in my life. I also thank my childhood friends Bunt *paji*, Ricky, and Kiki for showing faith in me.

Finally, I thank myself for blindly jumping to this challenge without realizing its

gravity. I am grateful for doing all the hard work, being a vessel for creative ideas and managing anxiety-prone sleepless nights. I am astonished that I could achieve so much!

Vishal Sharma

Abstract

Ever since the dawn of computing devices, making machines intelligent has been an ambitious dream of humans. While the primary focus in the last decade of research has been to develop data-driven Machine Learning (ML) solutions, the design decisions while modelling both the problem definition and the solution approach have been critical in developing effective ML solutions. One such design choice is to add structure to the problem definition and/or in the solution approach by introducing symbols and their relations, like that in predicate and first-order logic and similarly in object-centric Deep Learning. Various problem definition frameworks and solution approaches in ML leverage structured representations to define and study a class of problems where individual problems are distinct from each other but are closely related to each other. For example, in Markov Logic Networks in Statistical Relational Learning, we can write a single first-order domain (called theory) that can be grounded using constants to create unique domain instances. Similarly, in the case of the task of planning, the structure is leveraged, for example, in Relational Markov Decision Processes, to define a set of (possibly infinite) domain instances such that these are related to each other by sharing the same underlying first-order transition function. In this work, we study the use of structure for learning in three crucial ML tasks: inference, planning and reinforcement learning. Specifically, we focus on the task of learning solutions that can be transferred from one instance of domain to another.

In chapter 3, we study the task of transfer in Lifted inference that reduces the complexity of inference in relational probabilistic models by identifying groups of constants (or atoms) which behave symmetric to each other. We present the first application of lifting rules for marginal-MAP (MMAP). We define a new equivalence class of (logical) variables, called Single Occurrence for MAX (SOM), and show that the solution lies at the extreme with respect to the SOM variables, i.e., predicate groundings differing only in the instantiation of the SOM variables take the same truth value. Further, we define a sub-class *SOM-R* (SOM Reduce) and exploit properties of extreme assignments to show that MMAP inference can be performed by reducing the domain of SOM-R variables to a single constant. We refer to our lifting technique as the *SOM-R* rule for lifted MMAP.

In the chapters 4 and 5, we study how to learn generalized neural policies in Relational Markov Decision Processes (RMDP) such that they can be transferred from one domain instance (variation) to an unseen one in a zero-shot manner. First, in chapter 4, we carefully identify problems in the existing state-of-the-art approach, called *Symbolic Network* (SYMNET), that first converts a given instance of RMDP into a graph (called *instance-graph*) and then uses a Graph Neural Network (GNN) based architecture to learn a policy network. We then propose SYMNET2.0 that systematically handles these problems in SYMNET by introducing a new instance graph and augments the GNN-based policy network to learn better policies than SYMNET. Next, in chapter 5, we identify that due to using a GNN of fixed depth, SYMNET2.0 struggles to learn policies that exploit long-range dependencies. As a remedy, we propose SYMNET3.0, where we first construct a novel *influence graph* characterized by edges capturing one-step influence (dependence) between nodes based on the transition model. We then define *influence distance* between two nodes as the shortest path between them in this graph – a feature we exploit to represent long-range dependencies in the instance-graph of SYMNET3.0.

In the final chapter 6, we focus on the task of learning generalized policies in Reinforcement learning that are capable of transferring the neural policy learned on a set of training environments to a set of unseen but related environments. We present *Object Centric Reinforcement Learning Agent (ORLA)*, an object-centric approach for model-free RL in perceptual domains. Given a perceptual input state, ORLA first identifies the set of objects in the scene; it then constructs a symbolic graph constituting various identified objects and their relations. Finally, a Graph Attention Network (GAT) based architecture is employed over the extracted object positions to learn a dense state embedding, which is then decoded to get the final policy that generalizes to unseen environments.

Finally, we conclude the thesis by outlining our contributions and discussing various directions of future work to extend the work presented in the thesis.

सार

कंप्यूटिंग उपकरणों की शुरुआत से ही, मशीनों को बुद्धिमान बनाने का सपना मानवों का एक महत्वाकांक्षी सपना रहा है। पिछले दशक में शोध का मुख्य ध्यान डेटा-चालित मशीन लर्निंग (एमएल) समाधानों को विकसित करने पर रहा है, लेकिन समस्या की परिभाषा और समाधान के दृष्टिकोण की मॉडलिंग प्रभावी एमएल समाधानों को विकसित करने में महत्वपूर्ण रहे हैं। ऐसा ही एक डिजाइन विकल्प समस्या की परिभाषा और/या समाधान के दृष्टिकोण में स्ट्रक्चर को प्रतीकों और उनके संबंधों के माध्यम से जोड़ना है, जैसे कि प्रेडीकेट और फर्स्ट-आर्डर लॉजिक में और इसी तरह ऑब्जेक्ट-सेंट्रिक डीप लर्निंग में। एमएल में विभिन्न समस्या परिभाषा फ्रेमवर्क और समाधान दृष्टिकोण स्ट्रक्चर प्रतिनिधित्वों का लाभ उठाते हैं ताकि समस्याओं के एक वर्ग को परिभाषित और अध्ययन किया जा सके, जहां व्यक्तिगत समस्याएं एक-दूसरे से अलग होती हैं लेकिन आपस में घनिष्ठ रूप से संबंधित होती हैं। उदाहरण के लिए, स्टैटिस्टिकल रिलेशनल लर्निंग में मार्कोव लॉजिक नेटवर्क्स में, हम एक ही फर्स्ट-आर्डर लॉजिक डोमेन (जिसे थ्योरी कहा जाता है) लिख सकते हैं जिसे ग्राउंड करके अद्वितीय डोमेन इंस्टेंस बनाया जा सकता है। इसी तरह, योजना बनाने के कार्य में, स्ट्रक्चर का उपयोग किया जाता है, उदाहरण के लिए, रिलेशनल मार्कोव निर्णय प्रक्रियाओं में, एक सेट (संभवतः अनंत) डोमेन उदाहरणों को परिभाषित करने के लिए, ताकि ये सभी एक ही आधारभूत फर्स्ट-आर्डर ट्रांजिशन फंक्शन को साझा करके आपस में संबंधित हों। इस काम में, हम तीन महत्वपूर्ण एमएल कार्यों में सीखने के लिए स्ट्रक्चर के उपयोग का अध्ययन करते हैं: अनुमान, योजना और रिइंफोर्समेंट लर्निंग। विशेष रूप से, हम उन समाधानों को सीखने के कार्य पर ध्यान केंद्रित करते हैं जिन्हें एक डोमेन के उदाहरण से दूसरे में ट्रान्सफर किया जा सकता है।

अध्याय 3 में, हम लिफ्टेड अनुमान में स्ट्रक्चर के कार्य का अध्ययन करते हैं, जो रिलेशनल प्रोबबिलिस्टिक मॉडल में उन स्थिरांक (या परमाणुओं) के समूहों की पहचान करके अनुमान की जटिलता को कम करता है जो एक-दूसरे के प्रति सममित व्यवहार करते हैं। हम 'मार्जिनल-एमएपी' (एमएमएपी) के लिए लिफ्टिंग नियमों के पहले अनुप्रयोग को प्रस्तुत करते हैं। हम एक नए 'इक्विवैलेन्स क्लास' (तार्किक) चर, जिसे 'सिंगल ऑकरेंस फॉर मैक्स' (एसओएम) कहते हैं, को परिभाषित करते हैं और दिखाते हैं कि समाधान एसओएम चर के संबंध में 'एक्सट्रीम' होता है, यानी, केवल एसओएम चर की अभिव्यक्ति में भिन्न 'प्रेडीकेट' ग्राउंडिंग्स एक ही सत्य मान लेते हैं। आगे, हम एक उप-वर्ग 'एसओएम-आर' ('एसओएम रिड्यूस') को परिभाषित करते हैं और यह दिखाने के लिए 'एक्सट्रीम असाइनमेंट्स' के गुणों का उपयोग करते हैं कि 'एसओएम-आर' चर के डोमेन को एक स्थिरांक में घटाकर एमएमएपी अनुमान किया जा सकता है। हम अपनी लिफ्टिंग तकनीक को लिफ्टेड एमएमएपी के लिए 'एसओएम-आर' नियम कहते हैं।

अध्याय 4 और 5 में, हम 'रिलेशनल मार्कोव निर्णय प्रक्रियाओं (आरएम्डीपी)' में जनरलाइज्ड न्यूरल नीतियों को सीखने का अध्ययन करते हैं ताकि उन्हें एक डोमेन उदाहरण से एक पहले कभी ना देखे गए उदाहरण में 'ज़ीरो-शॉट' तरीके से स्थानांतरित किया जा सके। पहले, अध्याय 4 में, हम मौजूदा अत्याधुनिक

दृष्टिकोण, जिसे 'सिंबॉलिक नेटवर्क (सिम्नेट)' कहा जाता है, में समस्याओं की सावधानीपूर्वक पहचान करते हैं, जो पहले आर्म्पूडीपी के दिए गए उदाहरण को एक ग्राफ (जिसे 'इंस्टेंस-ग्राफ' कहा जाता है) में परिवर्तित करता है और फिर एक 'ग्राफ न्यूरल नेटवर्क (जीएनएन)' आधारित आर्किटेक्चर का उपयोग करके एक नीति नेटवर्क सीखता है। इसके बाद, हम 'सिम्नेट2' का प्रस्ताव करते हैं, जो सिम्नेट में इन समस्याओं को व्यवस्थित रूप से संभालता है, एक नया इंस्टेंस ग्राफ पेश करके और जीएनएन-आधारित नीति नेटवर्क को सिम्नेट से बेहतर नीतियों को सीखने के लिए बदलता है। अगले अध्याय 5 में, हम पहचानते हैं कि एक निश्चित गहराई के जीएनएन का उपयोग करने के कारण, सिम्नेट2 लंबी दूरी की निर्भरताओं का लाभ उठाने वाली नीतियों को सीखने में संघर्ष करता है। इसके समाधान के रूप में, हम सिम्नेट3 का प्रस्ताव करते हैं, जहां हम पहले एक नए 'प्रभाव ग्राफ' का निर्माण करते हैं जिसे 'ट्रांजीशन मॉडल' के आधार पर 'नोड्स' के बीच एक-चरणीय प्रभाव (निर्भरता) को पकड़ने वाले 'ग्राफ एज' द्वारा चित्रित किया जाता है। फिर हम इस ग्राफ में दो नोड्स के बीच 'प्रभाव दूरी' को उनके बीच के सबसे छोटे पथ के रूप में परिभाषित करते हैं - एक विशेषता जिसे हम सिम्नेट3 के इंस्टेंस-ग्राफ में लंबी दूरी की निर्भरताओं का लाभ उठाने के लिए उपयोग करते हैं।

अंतिम अध्याय 6 में, हम 'जनरलाइज्ड न्यूरल नीति' को सीखने के कार्य पर ध्यान केंद्रित करते हैं जो प्रशिक्षण वातावरणों के एक सेट पर सीखी गई न्यूरल नीति को पहले कभी ना देखे गए लेकिन संबंधित वातावरणों के एक सेट में स्थानांतरित करने में सक्षम हैं। हम 'ऑब्जेक्ट सेंट्रिक रिइंफोर्समेंट लर्निंग एजेंट (ओरला)' प्रस्तुत करते हैं, जो 'परसेप्युअल' डोमेन में 'मॉडल-फ्री रिइंफोर्समेंट लर्निंग' के लिए एक वस्तु-केंद्रित दृष्टिकोण है। दिए गए परसेप्युअल इनपुट स्थिति में, ओरला पहले दृश्य में वस्तुओं के सेट की पहचान करता है; फिर यह विभिन्न पहचानी गई वस्तुओं और उनके संबंधों का एक प्रतीकात्मक ग्राफ बनाता है। अंत में, एक 'ग्राफ अटेंशन नेटवर्क (जीएटी)' आधारित आर्किटेक्चर निकाले गए वस्तु पदों पर लागू किया जाता है ताकि एक सघन स्थिति एम्बेडिंग सीखी जा सके, जिसे फिर 'डिकोड' किया जाता है ताकि अंतिम नीति प्राप्त हो सके जो पहले कभी ना देखे गए वातावरणों में जनरलाइज होती है।

अंत में, हम अपने योगदानों को संक्षेप में प्रस्तुत करते हैं और भविष्य में किये जा सकने वाले कार्य की विभिन्न दिशाओं पर चर्चा करते हैं ताकि इस शोध प्रबंध में प्रस्तुत कार्य को बढ़ाया जा सके।

Contents

Certificate	i
Acknowledgements	v
Abstract	ix
Abstract in Hindi	xi
1 Introduction	1
1.1 Structured Representations in Machine Learning	3
1.1.1 Reasoning	3
1.1.2 Planning	5
1.1.3 Reinforcement Learning	7
1.2 Contributions	10
1.2.1 Lifted Marginal MAP Inference	10
1.2.2 Learning Generalized Neural Policies for RDDDL RMDPs	11
1.2.3 Learning Generalized Neural Policies for RDDDL RMDPs with Long Range Dependencies	11
1.2.4 Learning Object-centric Reinforcement Learning Agent for zero-shot transfer learning	12
1.3 Thesis Outline	13
2 Background and Notations	15
2.1 Markov Logic Network	15
2.2 Markov Decision Processes	17
2.3 Relational Markov Decision Process	18
3 Lifted Marginal MAP Inference	23
3.1 Introduction	23
3.2 Preliminaries	25

3.3	Single Occurrence for MAP	26
3.3.1	Motivation	26
3.3.2	SOM implies Extreme Solution	27
3.3.3	SOM-R Rule for lifted MMAP	32
3.4	Algorithmic Framework	34
3.5	Experiments	35
3.6	Conclusion	39
4	SymNet 2.0: Effectively handling Non-Fluents and Actions in Generalized Neural Policies for Relational MDPs	41
4.1	Introduction	42
4.2	Background and Related Work	43
4.2.1	Transfer Learning for RMDPs	43
4.2.2	Related Work	44
4.2.3	SYMNET	45
4.3	SYMNET2.0: A New Architecture	46
4.3.1	Running Example	46
4.3.2	Shortcomings in SYMNET	47
4.3.3	Our Approach	48
4.3.4	Training Algorithm	51
4.3.5	Representational Capabilities	52
4.4	Experiments	52
4.4.1	Experimental Setup	53
4.4.2	Results	54
4.5	Conclusion	58
5	SymNet 3.0: Exploiting Long-Range Influences in Learning Generalized Neural Policies for Relational MDPs	59
5.1	Introduction	60
5.2	Related Work	61
5.3	Technical Contributions	62
5.3.1	Limitations of SYMNET2.0	62
5.3.2	SYMNET3.0: Incorporating Influence	64
5.3.3	Representability	67
5.4	Experiments	68
5.4.1	Experimental Setup	68
5.5	Results	71

5.5.1	Insights	72
5.6	Conclusion	73
6	ORLA	75
6.1	Introduction	76
6.2	Related Work	78
6.3	The ORLA Agent	79
6.3.1	Self-Supervised Object Extraction	80
6.3.2	Symbolic State Representation	82
6.3.3	Learning a Generalized Policy	83
6.3.4	Contrasting ORLA with SYMNET line of work	84
6.4	Experiments and Results	84
6.4.1	Case Study 1: Balls and Pedal	85
6.4.2	Case Study 2: Robot Navigation	88
6.5	Conclusion	91
7	Conclusion and Future Work	93
7.1	Conclusion	93
7.2	Future Work	95
7.2.1	Planning and Large Language Models	95
7.2.2	Multi Modal Planners	95
7.2.3	Model-based Object-centric Reinforcement Learning	96
I	Appendices	111
A	Appendix to Chapter 3	113
A.1	Lemmas	113
B	Appendix to Chapter 4	119
B.1	Proofs of Propositions	119
B.2	Architecture Details	119
B.3	Domain Description	120
B.4	Statistical Significance Test	122
B.5	Large instance generation	122
B.6	Standard Error among results	124
B.7	Raw rewards	124

C Appendix to Chapter 5	131
C.1 Proofs	131
C.2 RDDDL Example	133
C.3 Experimental Details	133
C.4 Detailed Results and Attention Maps	134
C.5 Sizes of instances	137
C.6 Domains and Generators	137
D Appendix to Chapter 6	143
D.1 Architectures	143
D.2 Training	144
D.3 Detailed Dueling DQN Results	145
D.4 Detailed PPO Results	145
D.5 Results on using existing methods for expert object extractor	147
List of Publications	151
Biography	153

List of Figures

1.1	Figure shows some of the available game environments in Atari test suit [Bellemare et al., 2013, Machado et al., 2018].	8
2.1	Figure shows the DBN for the 2×2 grid size Recon instance.	21
3.1	Results of Student, FS and IMDB datasets.	37
3.2	Rules of Student, FS and IMDB datasets. "size" gives initial domain sizes for each case.	38
4.1	(left): Graph capturing action-independent effects (ref. 4.2), G_d ; (middle): one of the six action induced graphs (ref. 4.2), G_{down} , for the down action; (right): one of the three position-based graphs (ref. 4.3.3), G_{p2} , for the second position. All nodes have a self loop (not shown for visual clarity). Red nodes are present in both SYMNET2.0 and SYMNET, where as blue nodes are present only in SYMNET2.0. Position-based graphs, e.g., G_{p2} , are present only in SYMNET2.0.	48
4.2	Performance trends on instances of increasing size: PROST deteriorates, but SYMNET2.0 remains robust.	55
4.3	Coverage of SYMNET2.0 (left) and SYMNET-IL (right) on grid size 20×20 when trained on grid size 5×5	57
5.1	Figure shows the three-step process of SYMNET3.0 for policy prediction. The instance graph and influence graph are representative of the Navigation domain (See Appendix C for domain description). The instance graph has nodes for object-tuples $((x_i, x_j), (y_i, y_j), (x_i, y_j), x_i, y_i)$ and the influence graph has nodes for predicates $(R(x_i, y_j))$ denoting the robot_at predicate). In the case of SYMNET2.0, only instance-graph is present.	65

5.2	(a) Shows the color-coded locations of a 23×23 instance of the DNav domain where R and G are the robot and goal. Figures (b) and (c) show the 2D t-SNE plot for node embeddings of the grid locations for SYMNET2.0 and SYMNET3.0+KL.	72
5.3	The attention map of the influence-layer in the Pizza domain for the R node.	72
6.1	Given interactions in environments with 3 or fewer balls, with the goal to keep the balls from falling, our objective is to learn a policy which would do well in an environment with four or more balls.	76
6.2	(Top Left) Training details of our unsupervised expert Object extractor. (Top Right) Training of YOLO based object extractor using multiple Experts. (Bottom) A forward pass from ORLA that takes an image as input, converts it into a symbolic state graph, and then uses GAT based architecture to get the policy.	81
6.3	Figure shows the zero-shot transfer reward on the test environments vs. #frames seen on the training environments for all domains. The dashed line represents the reward of a uniform random policy over 500 episodes to counter stochasticity in the policy. For all other methods, we take an average of 10 episodes. (See Figure D.1 in the Appendix for graphs on all train and test environments of all domains.)	88
6.4	Figure shows the goal coverage for best-trained models of ORLA and CNN on the Navigation domain when trained on a smaller robot to goal distance and transferred to a much larger distance in a zero-shot manner. Here, the robot always starts from the location (5, 5) (marked as R in the figure), and the goal is kept in the remaining 120 cells one by one. In both images, a green cell represents that the robot was able to reach the goal placed at that cell and a red cell represents the robot could not reach the goal. . .	90
C.1	(left) Figure shows the attention map averaged across all heads for the robot’s location computed by SYMNET3.0+KL for the DNav domain. We note that the attention heads focus on the corners of the grid helping in the localization of all nodes. (right) Figure shows the attention map averaged across all heads for the robot’s location computed by SYMNET3.0+KL for the SRecon domain. Here, 0 and 1 denote the object 0 and object 1. We note that the attention heads focus on one of the corners of the grid. . .	134

C.2	(left) Figure shows the attention map averaged across all heads for the robot’s location computed by SYMNET3.0+KL for the StNav domain. Here, the probability of death of each cell is written on the cell. We note that the attention is focused on the entrance of the column which is safest. (right) Figure shows the attention map averaged across all heads for the robot’s location computed by SYMNET3.0+KL for the StWall domain. Here, the attention is focuses on the goal and the cells near the safe passage cell.	134
D.1	Figure showing performance (average reward vs #frames) of various methods on all domains on each of the train and test environments for dueling DQN.	145
D.2	Figure showing performance (average reward vs #frames) of various methods on all domains on each of the train and test environments for PPO	146
D.3	Figure shows the performance of slot attention [Locatello et al., 2020] on Pizza domain for unsupervised object segmentation. The first column is the original image to be decomposed into objects; the second column is the reconstructed image as output by slot attention. The last five columns show the five slots (4 objects and one background). We notice that while slot attention reconstructs the image fairly well, it fails to assign an object to each slot.	149
D.4	Figure shows the performance of SPACE [Lin et al., 2020] on the Pizza domain for unsupervised object segmentation. The first column is the original image to be decomposed into objects; the second column is the reconstructed image as output by slot attention. The third column is the foreground content (only that part of the scene that contains various objects). The fourth column shows the bounding boxes over various detected objects. The last column shows the background. We notice that while SPACE is able to reconstruct the image fairly well, it fails to assign a bounding box to any of the objects in the scene. Rather, it detects the whole scene as the background.	150

List of Tables

1.1	Rules of Friends and Smoker MLN theory.	4
4.1	Comparison between SYMNET2.0 and the baselines on 12 IPPC domains. All models are trained on (smaller) instances 1-3 and validated on instance 4. Upper part shows results on IPPC test instances 5-10 and lower part shows results on much larger instances than those in the IPPC. Bold values show the best performer among all neural models.	55
5.1	Comparison of SYMNET3.0 with the baselines on 6 LR domains (bold denotes the best-performing neural model)	71
5.2	Comparison of SYMNET3.0 with the baselines on 12 IPPC domains (bold denotes best-performing neural model)	71
B.2	Results showing comparison between SYMNET2.0 and the baselines on 12 IPPC domains when we vary the neighborhood size of GAT (column "GAT").	120
B.3	Number of state-variables (SP_O) per instance for IPPC and large domains	123
B.4	Results showing comparison between SYMNET2.0 and the baselines on 12 IPPC domains. All models are trained on (smaller) instances 1-3 and validated on instance 4. All Rows show results on IPPC instances 5-10. Bold values show the best performer among all neural models. Each entry gives the mean relative score \pm standard error over 3 runs.	123
B.5	Results showing scores of 3 individual runs of SYMNET2.0 and other baselines on 12 IPPC domains on instances 5-10. All models are trained on (smaller) instances 1-3 and validated on instance 4. Each entry shows the relative score on three runs.	125

B.6	Results showing comparison between SYMNET2.0 and the baselines on 12 IPPC domains. All models are trained on (smaller) instances 1-3 and validated on instance 4. All Rows show results on larger instances (11-14) than those in the IPPC. Bold values show the best performer among all neural models. Each entry gives the mean relative score \pm standard error over 3 runs. The standard error is high (only) in some cases as only 3 runs were done due to resource constraints.	125
B.7	Results showing scores of 3 individual runs of SYMNET2.0 and other baselines on larger instances (11-14) of 12 IPPC domains. All models are trained on (smaller) instances 1-3 and validated on instance 4. Each entry shows the relative score on three runs.	125
B.8	Acad : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	126
B.9	CT : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	126
B.10	Elev : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	126
B.11	GoL : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	127
B.12	Nav : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	127
B.13	Recon : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	127
B.14	Skill : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	128
B.15	Sys : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	128

B.16 Tam : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	128
B.17 Traffic : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	129
B.18 TT : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	129
B.19 Wild : Table showing raw long term rewards for each neural model averaged over 200 runs in instances 1-10 and 100 runs in instances 11-14. Each entry shows results for 3 runs.	129
C.1 Performance of all runs of different models on 6 LR domains.	135
C.2 Performance of all runs of different models on 12 IPPC domains.	135
C.3 Comparison of SYMNET3.0 variants with the baselines on 6 LR domains. The last row denotes the score of the best among SYMNET3.0+KL, SYMNET3.0-KL and SYMNET3.0-KL _D chosen on the basis of average validation reward.	136
C.4 Comparison of SYMNET3.0 variants with the baselines on 12 IPPC domains. The last row denotes the score of the best among SYMNET3.0+KL, SYMNET3.0-KL and SYMNET3.0-KL _D chosen on the basis of average validation reward.	136
C.5 Number of state fluents for LR domains for training, validation, and test sets.	137
C.6 Number of state fluents for IPPC domains for training, validation and test sets.	137
C.7 Table shows the parameters used in to generate instances in the DNav domain.	138
C.8 Table shows the parameters used in to generate instances in the EAcad domain.	138
C.9 Table shows the parameters used in to generate instances in the SRecon domain.	138
C.10 Table shows the parameters used in to generate instances in the Pizza domain.	138
C.11 Table shows the parameters used in to generate instances in the StWall domain.	138

C.12 Table shows the parameters used in to generate instances in the StNav domain.	138
D.1 Table shows the number of episodes and frames used to train experts in various domains. For MDB, we do not train any experts. Rather, we use the experts trained on Column Ball	144