

DEEP LEARNING WITH OUTPUT SPACE CONSTRAINTS

YATIN NANDWANI



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY DELHI

APRIL 2024

DEEP LEARNING WITH OUTPUT SPACE CONSTRAINTS

by

YATIN NANDWANI

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
APRIL 2024

Certificate

This is to certify that the thesis titled **Deep Learning with Output Space Constraints** being submitted by **Mr Yatin Nandwani** 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, chapter 4, chapter 5, chapter 6 and chapter 7 were done jointly with undergraduate students. In each case, the part done by the collaborators appeared in their respective bachelor's thesis.

Mausam

Professor

Department of Computer Science and Engg.

Indian Institute of Technology Delhi

New Delhi- 110016

Parag Singla

Professor

Department of Computer Science and Engg.

Indian Institute of Technology Delhi

New Delhi- 110016

Acknowledgements

I decided to leave my comfortable job and challenge myself by enrolling in the PhD program at the age of thirty-two. With two kids and so much responsibility, it was never an easy decision, filled with uncertainty, doubts and doubters! Yet, amidst the sceptics and the challenges, one unwavering presence stood by my side like a steadfast pillar: my beloved wife, Shagun. It is to her unwavering support that I dedicate this thesis, for without her, the very thought of seeking this academic pursuit at this stage of life would have been inconceivable.

With the responsibility of two kids, *Ziven* and *Zaina*, it could have been difficult for both Shagun and me to handle it all alone, especially during the initial years when Zaina was just an infant. Thanks to the invaluable assistance of our parents, sisters, and brothers-in-law, who played pivotal roles in tending to our children, I could get the needed time and focus for my research and coursework. During COVID, I often went to Shagun's sister's home, or sometimes my own sister's home, to study. This provided a much-needed break and a welcome change of place.

Despite the added responsibility that comes with children, witnessing their growth is a source of constant joy. Engaging in playtime and spending quality moments with them often served as a welcome distraction from my studies. Their presence proved to be a fantastic equalizer and an effective stress reliever.

This journey wouldn't have been as incredible as it turned out to be without my supervisors, *Mausam* and *Parag*. They are truly inspiring, excelling not only in their roles as researchers and educators but also as people deeply committed to their profession and society. Their strong sense of duty towards the academic community and humanity sets a remarkable example for all. Apart from our research endeavours, I cherished engaging in philosophical conversations with both of them on various topics, ranging from our societal responsibilities to the meaning of life and beyond.

I had the privilege of serving as one of the workflow chairs for AAAI'21 when Mausam was the program co-chair. His dedication and resilience were truly remarkable; even while battling COVID in the hospital, he had conference-related work in the back of his mind, displaying an unparalleled commitment to his responsibilities. This was the first time I

saw him not just as an advisor but also as a dedicated researcher. Besides this, I worked very closely with him to edit the video content for the NPTEL course on AI. This is when I realized that a lot of effort goes behind the scenes to cater to such a big course! We jointly presented a tutorial on neuro-symbolic AI at a workshop. The flow of the tutorial significantly influenced the narrative and the story that connects the different works presented in my thesis!

I thoroughly enjoyed attending Parag's lectures, particularly his course on Deep Learning, which was the first course I attended. He teaches with a lot of patience, and I can't imagine anyone else delivering the material better than him. Initially, our discussions revolved around research topics. I later learned about the A.I.N.A. (An Initiative for National Advancement) initiative that Parag has been running since 2012. This led to an opportunity to join one of the trips to Uttarakhand, where many volunteer students participated, accompanied by both Parag and me, along with our families. During this trip, I got to engage in philosophical discussions with Parag, which added depth to our interactions. Witnessing the dedication of the students involved in the initiative was truly inspiring and served as a motivating force for me.

Besides Mausam and Parag, I enjoyed my interactions with Prathosh and Rohan. Deep Learning was the first course that Prathosh taught at IIT Delhi, and I enjoyed the couple of discussions I had with him about the course content. I was one of the TAs for Rohan's AI course, and working closely with such a humble and down-to-earth person was a wonderful experience.

At NeurIPS'18, my very first international conference, I had the chance to interact with many researchers. One name that stands out is Alexander Rush (Shasha). My interactions with him proved crucial for my first work, which eventually got published at NeurIPS'19. His tutorial on dual decomposition and Lagrangian relaxation in NLP [Rush and Collins, 2012] provided a foundational understanding that laid the groundwork for my research.

My first publication at NeurIPS'19 introduced me to the research community focusing on deep learning with constraints. I had the opportunity to connect with Maneesh Singh and Vivek Srikumar, whose work overlapped with mine. This fruitful connection led to a joint internship with Vivek and Maneesh at Verisk, where I tackled the problem of identifying temporal relationships between events in documents. While the internship transitioned to a work-from-home format due to COVID, the experience was invaluable.

During my stint as AAAI'21 workflow chair, I got an opportunity to work closely with Prof. Kevin Leyton-Brown (he was the co-chair along with Mausam) and his students at UBC - Heddy, Chris, Neil. We ended up publishing our findings, and I remember writing the first draft of the technical section. One night before the deadline, the paper was in bad shape, and I was almost sure that we wouldn't be able to submit it. Kevin started

polishing the writing (actually, rewrote most of it) just 24 hours before the deadline, and the transformation of the paper was unbelievable! Thanks, Kevin, for the wonderful exposure!

One person who warrants a special mention here is Ankit Anand. I published my first paper with him, attended my first conference with him, and stayed at his home in Montreal countless times! Thanks, Ankit and Pallavi, for hosting me with smiles! What a nice way of giving back for the home food that we sometimes shared together ;-).

I shared most of my time in the lab with Keshav, who happens to be ten years younger than me! I always felt younger in his company! Thanks, Keshav, for the wonderful camaraderie we shared.

Before COVID, we used to play cricket on campus, and on many days, it was the primary motivation to come to college! Thanks to all the players who made it happen, especially Happy, who would happily give me a lift back home!

As we prepared for our job interviews, Prachi, Keshav, Arnab, Vishal, Kunal, and I decided to form a reading group. It turned out to be a fantastic decision! We met weekly, and each person took turns presenting a topic. Each of us had expertise in different areas – Kunal in extreme multi-label learning, Prachi and Keshav in NLP, Vishal in RL, and Arnab in VAEs. Learning from each other’s expert knowledge turned our preparation into an exciting journey of learning. Thanks to everyone’s enthusiasm and insights, our sessions were not only informative but also a lot of fun. Thanks, guys, for sharing your invaluable insights and making the process rewarding and enjoyable! I hope that our friendship continues and we stay in touch!

While working for AAAI’21, I formed a close collaboration with Dinesh Raghu. Little did I know that this partnership would extend far beyond the conference, with Dinesh eventually playing a pivotal role in getting me a position at the IBM Research Lab (IRL). Thanks, Dinesh, for an amazing collaboration! The interactions with my manager, Sachin, and other teammates at IRL have been incredibly stimulating. Thanks, Sachin, for offering me the opportunity and for also granting me the necessary time to complete my thesis! Special thanks to Gaurav, Mayank, and Kushal for discussing and helping with the additional experiment I did to incorporate one of the thesis reviewer’s suggestions.

To write the abstract of this thesis in Hindi, I first used GPT-4. Sadly, it didn’t do a good job. Fortunately, one of my uncles, Prof. Harish Sethi, who specializes in translating technical documents to Hindi, visited us on a Sunday and helped me with the translation. Thank you, uncle, for your timely help!

Finally, I would like to thank all my collaborators – Bhavesh, Abhishek, Aman, Ankesh, Mayank, Deepanshu, Vidit, and Rishabh. I spent a lot of time brainstorming, arguing, and discussing various topics with them. Thanks, guys, for the many sleepless nights close

to the conference deadlines! I wish you all the success and happiness in life. With your support and company, my PhD journey became really enjoyable. I hope that our paths will cross again!

In the end, I would once again like to thank –

Parag and Mausam, who guided me on my academic journey ...

Zaina, whose birth marked the journey of my re-discovery ...

Ziven, whose birth marked the journey of my parenthood ...

Shagun, who marked the journey of my partnership ...

My spiritual masters, who marked my inner journey of self-discovery ...

My parents, who marked my outer journey in this world ...

Abstract

Since the emergence of Deep Learning, the field of Artificial Intelligence has experienced remarkable progress across multiple domains, including Computer Vision and Natural Language Processing. Neural Networks have consistently achieved state-of-the-art performance across various tasks within these fields. The output of these neural networks is often symbolic, and there may exist known or unknown constraints between these symbols. These constraints may provide a natural way of representing domain knowledge related to the underlying task. We can categorize these constraints based on various dimensions, such as their representation (explicit vs implicit), knowledge (known vs unknown), hard vs soft constraints *etc.*. We use the first two dimensions of ‘constraint representation’ and ‘knowledge of constraints’ to categorize our work into three broad buckets of “explicitly known constraints”, “unknown constraints with implicit representation”, and “unknown constraints with explicit representation”.

In the first part, ‘[Explicitly Known Constraints](#)’, we explore if neural models can be better trained using known symbolic domain knowledge expressed as hard-constraints on the input/output variables. When constraints are hard, most of the existing works focus on constrained inference [[Rush and Collins, 2012](#), [Ning et al., 2019](#)]. Few works do exploit constraints during training [[Hu et al., 2016a](#), [Xu et al., 2018](#)], however, their formulation is capable of handling only soft constraints. In contrast, we propose a primal-dual formulation that can potentially deal with hard constraints [[Nandwani et al., 2019](#)].

In the next two parts, we focus on automated learning of unknown constraints from the data. In many scenarios, it is difficult to encode our intuition explicitly in the form of a constraint, or we may miss certain constraints while enumerating them, warranting the need for their automated learning. Here we focus on combinatorial problems, such as sudoku, that involve satisfaction of the underlying hard constraints defining the problem. Based on the representation of constraints, we categorize existing approaches for learning of unknown constraints into two categories: *implicit* and *explicit*. Works in the former category propose purely neural models [[Palm et al., 2018](#), [Dong et al., 2019](#)] for solving these tasks, representing the underlying constraints implicitly in their weights. In the second part of the thesis, ‘[Unknown Constraints with Implicit Representation](#)’, we identify

a couple of potential issues in such models and propose appropriate solutions. First, we identify the issue of solution multiplicity (one input having many correct solutions) while training neural models and propose appropriate loss functions and an RL based technique to optimize it [Nandwani et al., 2021]. Next, we observe that existing architectures, such as SATNet [Wang et al., 2019], Relational Recurrent Networks (RRN) [Palm et al., 2018], fail to generalize across the output space of variables (can not solve 16 x 16 sudoku after training on only 9 x 9 sudoku). In response, we design two neural architectures for output space invariance in combinatorial problems [Nandwani et al., 2022a].

The third part of the thesis, ‘Explicit and Unknown Constraints’, focuses on the *explicit* representation of constraints, *e.g.*, linear inequalities over output variables as in an Integer Linear Program (ILP) that can express many combinatorial problems. One shortcoming of the existing approaches [Paulus et al., 2021, Wilder et al., 2019] is that they need to solve the ILP in every learning iteration, thereby considerably slowing down the training process. In the third part, we first propose a solver-free framework for scalable learning of constraints in an ILP [Nandwani et al., 2022b]. Finally, we take a completely different view of constraints as an explanation for the output of a black-box neural model. Specifically, we focus on explaining the output of Tensor-Factorization based models for the task of Knowledge Base Completion. We propose a post-hoc explanation engine that produces constraints in the form of horn clauses as an explanation. These constraints may not always hold for all the samples in the data, thereby making them ‘soft’, unlike all our previous works where it is not allowed to violate the constraints, making them ‘hard’.

In the end, we conclude by summarizing our contributions and proposing directions for future work that builds on the techniques and methods introduced in this work.

सार

डीप लर्निंग के उद्भव और विकास के कारण आर्टिफिशियल इंटेलिजेंस के कंप्यूटर विज्ञान और प्राकृतिक भाषा संसाधन (नेचुरल लैंग्वेज प्रोसेसिंग) जैसे विविध क्षेत्रों में उल्लेखनीय प्रगति देखी गई है। न्यूरल नेटवर्क्स ने इन क्षेत्रों के भीतर विभिन्न कार्यों में लगातार अत्याधुनिक (स्टेट-ऑफ-द-आर्ट) प्रगति की है। इन न्यूरल नेटवर्क्स का आउटपुट अक्सर प्रतीकात्मक होता है, और इन प्रतीकों के बीच ज्ञात या अज्ञात कॉन्स्ट्रेंट्स हो सकते हैं। ये कंस्ट्रेंट्स अंतर्निहित कार्य से संबंधित क्षेत्र-विशेष के ज्ञान को दर्शाने का एक स्वाभाविक तरीका प्रदान कर सकते हैं। हम इन कंस्ट्रेंट्स को विभिन्न आयामों के आधार पर वर्गीकृत कर सकते हैं। जैसे, उनका रिप्रेजेंटेशन (एक्सप्लिसिट बनाम इम्प्लिसिट), ज्ञान (ज्ञात बनाम अज्ञात), हार्ड बनाम सॉफ्ट कंस्ट्रेंट्स आदि। इस शोध अध्ययन को पहले दो आयामों – 'कंस्ट्रेंट रिप्रेजेंटेशन' और 'कंस्ट्रेंट्स की नॉलेज' – का उपयोग करके तीन मुख्य वर्गों में विभाजित किया गया है। ये हैं – "एक्सप्लिसिटली ज्ञात कंस्ट्रेंट्स", "अज्ञात कंस्ट्रेंट्स विद इम्प्लिसिट रिप्रेजेंटेशन"; और "अज्ञात कंस्ट्रेंट्स विद एक्सप्लिसिट रिप्रेजेंटेशन"।

पहले भाग, अर्थात "एक्सप्लिसिटली ज्ञात कंस्ट्रेंट्स" में, हम यह पता लगाते हैं कि क्या न्यूरल मॉडल्स को ज्ञात सिंबोलिक डोमेन नॉलेज का उपयोग करके बेहतर तरीके से प्रशिक्षित किया जा सकता है? इस सिंबोलिक डोमेन नॉलेज को हम इनपुट/आउटपुट वैरिएबल्स पर हार्ड कंस्ट्रेंट्स के रूप में व्यक्त करते हैं। जब कंस्ट्रेंट्स हार्ड होते हैं, तब अधिकांशतः कार्य कंस्ट्रेंट्स इंफरेंस पर केंद्रित होते हैं। कुछ कार्य, प्रशिक्षण के दौरान कंस्ट्रेंट्स का लाभ उठाते हैं, हालांकि, उनका फॉर्मूलेशन केवल सॉफ्ट कंस्ट्रेंट्स का उपयोग करने में सक्षम है। इसके स्थान पर, हमने एक प्राइमल-डुअल फॉर्मूलेशन को प्रस्तावित किया है जो संभवतः प्रशिक्षण के दौरान हार्ड कंस्ट्रेंट्स को भली प्रकार से डील करने में सक्षम हो सकता है।

अगले दो भागों में, डेटा से अज्ञात कंस्ट्रेंट्स को मॉडल के द्वारा स्वतः सीखने पर ध्यान केंद्रित किया गया है। कई परिदृश्यों में, हमारी इंट्यूशन को कंस्ट्रेंट के रूप में स्पष्टतः एन्कोड करना कठिन होता है या हम उनका उल्लेख करते समय कुछ कंस्ट्रेंट्स भूल सकते हैं। इस कारण मॉडल को डेटा से कंस्ट्रेंट्स को स्वतः सीखने की आवश्यकता होती है। हम यहां सुडोकु जैसे कॉम्बिनेटोरियल प्रॉब्लम्स पर ध्यान केंद्रित करते हैं, जिसमें प्रॉब्लम को परिभाषित करने वाले हार्ड कंस्ट्रेंट्स को मॉडल के द्वारा सीखना होता है। कंस्ट्रेंट्स के रिप्रेजेंटेशन के आधार पर, अज्ञात कंस्ट्रेंट्स को सीखने के लिए हमने वर्तमान दृष्टिकोणों को दो श्रेणियों में वर्गीकृत किया है — इम्प्लिसिट और एक्सप्लिसिट। इम्प्लिसिट श्रेणी के काम, कॉम्बिनेटोरियल प्रॉब्लम्स को हल करने के लिए विशुद्ध रूप से न्यूरल मॉडल्स का प्रस्ताव देते हैं, जो अपने वेट्स में इम्प्लिसिट तरीके से कंस्ट्रेंट्स को रिप्रेजेंट करते हैं। शोध-प्रबंध के दूसरे भाग — 'अज्ञात कंस्ट्रेंट्स विद इम्प्लिसिट रिप्रेजेंटेशन' — में हमने इन मॉडलों में कुछ संभावित मुद्दों की पहचान की है और उनके समुचित समाधान प्रस्तावित किए हैं। इस प्रक्रिया में हमने सबसे पहले न्यूरल मॉडल्स को प्रशिक्षित करते समय 'सॉल्यूशन मल्टीप्लिसिटी' (एक इनपुट के कई सही आउटपुट होना) की समस्या की पहचान की है। फिर, इसे हल करने के लिए हमने एक उचित लॉस फंक्शन

निर्मित किया है और उसे ऑप्टिमाइज़ करने के लिए एक आर.एल. आधारित तकनीक को प्रस्तावित किया है। इसके बाद, हमने यह ध्यान दिया कि रिलेशनल रिकरेंट नेटवर्क्स (आर.आर.एन.) जैसे मौजूदा आर्किटेक्चर्स, आउटपुट स्पेस में जनरलाइज़ करने में सार्थक सिद्ध नहीं होते हैं (जैसे केवल 9×9 सुडोकु पर प्रशिक्षण के बाद 16×16 सुडोकु हल नहीं कर सकते)। इसलिए, इसके स्थान पर, हमने कॉम्बिनेटोरियल प्रॉब्लम्स में आउटपुट स्पेस इन्वेरियन्स के लिए दो न्यूरल आर्किटेक्चर्स को डिज़ाइन किया है।

शोध-प्रबंध का तीसरा भाग, 'एक्सप्लिसिट और अननोन कंस्ट्रेंट्स', कंस्ट्रेंट्स के एक्सप्लिसिट रिप्रेजेंटेशन पर केंद्रित है। उदाहरण के तौर पर, इंटीजर लीनियर प्रोग्राम (आई.एल.पी.) की लीनियर इनइक्वालिटीज़, जो कई कॉम्बिनेटोरियल प्रॉब्लम्स को व्यक्त कर सकती हैं। वर्तमान दृष्टिकोणों की एक कमी यह है कि उन्हें प्रशिक्षण की हर इटरेशन में आई.एल.पी. को हल करना पड़ता है, जिससे प्रशिक्षण प्रक्रिया में काफी समय लगता है। तीसरे भाग में, हमने आई.एल.पी. के कंस्ट्रेंट्स को स्केलेबल तरीके से सीखने के लिए एक सॉल्वर-फ्री फ्रेमवर्क को प्रस्तावित किया है। अंत में, हम कंस्ट्रेंट्स को एक नए नजरिए से देखते हैं जिसमें उनका प्रयोग ब्लैक-बॉक्स न्यूरल मॉडल के आउटपुट को समझाने के लिए किया जाता है। विशेष रूप से, हमने नॉलेज बेस कंप्लीशन के कार्य के लिए टेंसर-फैक्टराइजेशन आधारित मॉडल्स के आउटपुट को समझाने पर ध्यान केंद्रित किया है। हमने एक पोस्ट-हॉक एक्सप्लेनेशन इंजन को प्रस्तावित किया है, जो हॉर्न क्लॉजेस के रूप में कंस्ट्रेंट्स को एक्सप्लेनेशन के तौर पर उत्पन्न करता है। ध्यान देने योग्य बात यह है कि एक्सप्लेनेशन के लिए प्रयोग किए गए कंस्ट्रेंट्स, सभी डेटा सैंपल्स के लिए हमेशा मान्य नहीं हो सकते हैं। इसलिए वे सॉफ्ट कंस्ट्रेंट्स हैं, जबकि इस शोध अध्ययन के अन्य सभी कार्यों में हार्ड कंस्ट्रेंट्स का प्रयोग किया गया है।

अंत में, हमने अपने कार्य के रूप में किए गए योगदानों का उपसंहार प्रस्तुत किया है और इस दिशा में भविष्य में होने वाले उन शोध-कार्यों के लिए दिशा-निर्देश प्रस्तावित किए हैं, जो इस अध्ययन में प्रस्तुत तकनीकों और विधियों पर आधारित होंगे।

Contents

Certificate	i
Acknowledgements	iii
Abstract	vii
Abstract in Hindi	ix
List of Figures	xviii
List of Tables	xxi
I Prologue	1
1 Introduction	3
1.1 Constraints in Machine Learning	5
1.1.1 Benefits of Imposing Constraints	5
1.1.2 Variables in the Constraint Expression	7
1.1.3 Representation of Constraints	8
1.1.4 Known vs Unknown Constraints	9
1.1.5 Hard Vs Soft Constraints	9
1.1.6 Methods for Combining ML Models and Constraints	9
1.2 Contributions	11
1.2.1 Known Constraints	11
1.2.2 Unknown Constraints	13
1.2.3 Implicit Representation of Unknown Constraints	14
1.2.4 Explicit Representation of Unknown Constraints	17
1.3 Thesis Outline	19

2	Related Work and Background	21
2.1	Explicitly Known Constraints	21
2.1.1	Constrained Inference	21
2.1.2	Constrained Training	24
2.2	Unknown Constraints	26
2.2.1	Neural Solvers with Implicit Constraint Representation	27
2.2.2	Symbolic Solvers with Explicit Constraint Representation	31
II	Explicitly Known Constraints	33
3	Learning with Known Constraints	35
3.1	Introduction	35
3.2	Related Work	38
3.3	Constrained Learning of Neural Models	39
3.3.1	A Lagrangian-based Formulation	40
3.3.2	Constraint Language for Discrete Output Spaces	41
3.4	Training	42
3.5	Experiments	44
3.5.1	Semantic Role Labeling (SRL)	44
3.5.2	Named Entity Recognition (NER)	46
3.5.3	Fine Grained Entity Typing	48
3.6	Conclusion	50
III	Unknown Constraints with Implicit Representation	51
4	1oML: One of Many Learning	53
4.1	Introduction	54
4.2	Background and Related Work	56
4.3	Theory and Algorithm	58
4.3.1	Problem Definition	58
4.3.2	Objective Function	59
4.3.3	Greedy Formulation: MINLOSS	63
4.3.4	Reinforcement Learning Formulation: SELECTR	64
4.3.5	Training Algorithm	66
4.4	Experiments with Non-autoregressive Models	67
4.4.1	Datasets and Prediction Networks	68

4.4.2	Baselines and Evaluation Metric	69
4.4.3	Results and Discussion	70
4.5	Experiments with Auto-regressive Models	73
4.5.1	Prediction Network	73
4.5.2	Results and Discussion	74
4.6	Conclusion	75
5	Neural Models for Output-Space Invariance in Combinatorial Problems	77
5.1	Introduction	78
5.2	Related Work and Background	79
5.2.1	Generalization of GNNs across Graph Sizes	81
5.3	Preliminaries and Problem Definition	82
5.4	Models Description	84
5.4.1	Binary Model	84
5.4.2	Multi-valued Model	88
5.4.3	Relative Comparison	92
5.5	Experiments	93
5.5.1	Task Description and Datasets	93
5.5.2	Experimental Setup & Baselines	96
5.5.3	Results and Discussion	98
5.6	Conclusion	100
IV	Explicit and Unknown Constraints	101
6	Scalable Learning of Constraints in Neural ILP Architectures	103
6.1	Introduction	104
6.2	Related Works	106
6.3	Differentiable ILP Loss	108
6.3.1	Background and Task Description	108
6.3.2	A Solver-free Framework	109
6.3.3	Negative Sampling	112
6.4	Experiments	113
6.4.1	Symbolic and Visual Sudoku	114
6.4.2	Random constraints	116
6.4.3	Knapsack from Sentence Descriptions	117
6.4.4	Keypoint Matching	118
6.5	Discussion	120

6.6	Conclusion	121
7	OxKBC: Outcome Explanation for Knowledge Base Completion	123
7.1	Introduction	124
7.2	Related Work	125
7.3	OxKBC: The Outcome Explanation Engine	127
7.3.1	Language for generating Explanation Paths	127
7.3.2	Templates: Aggregation of similar Explanation Paths	128
7.3.3	Selection Module	130
7.4	Experiments	133
7.4.1	Internal Evaluation	134
7.4.2	User Evaluation	135
7.4.3	Faithfulness of Explanations	138
7.5	Discussion	139
7.6	Conclusions	140
V	Epilogue	141
8	Conclusion and Future Work	143
8.1	Conclusion and Summary	143
8.2	Future Work	145
8.2.1	Combining Known and Unknown Constraints	145
8.2.2	Solution Multiplicity	146
8.2.3	Output Space Invariance	147
8.2.4	Neuro-symbolic Architectures	147
	Bibliography	179
VI	Appendices	181
A	Appendix to Chapter 3	183
A.1	Experimental Details	183
A.1.1	Semantic Role Labeling (SRL)	183
A.1.2	Named Entity Recognition (NER)	184
A.1.3	Fine Grained Entity Typing	184

B Appendix to Chapter 4	187
B.1 Experimental Details	187
B.1.1 Datasets and Prediction Networks	187
B.1.2 Results and Discussions	193
C Appendix to Chapter 5	195
C.1 Models Description	195
C.1.1 Alternate encoding scheme in Binarized Model	195
C.1.2 Experimental Details	196
D Appendix to Chapter 6	199
D.1 Mean and Std Err. over multiple runs	199
E Appendix to Chapter 7	201
E.1 Related Works	201
E.1.1 Comparison of TF based KBC Models with Path based Approaches	201
E.1.2 Pedagogical Approach	201
E.2 Experiment Details	202
E.3 Mechanical Turk Experiment Details	203
List of Publications	205
Biography	207

List of Figures

1.1	A schematic diagram placing our different works in a 2×2 grid with one dimension representing if the constraints are <i>Known</i> vs <i>Unknown</i> and the other dimension differentiating based on how the constraints are represented: <i>Explicit</i> or <i>Implicit</i>	12
1.2	A neuro-symbolic pipeline where the input is an image of a sudoku puzzle and the output is the corresponding solved sudoku. The ‘perception model’ decodes the image in each cell into one of the nine classes and the ‘reasoner’ discovers the unknown constraints while training and fills each cell with a digit between 1 and 9.	14
1.3	Solution Multiplicity: A simple sudoku input with only 4 empty cells and its corresponding solutions.	17
1.4	Neural-ILP Architecture: The reasoning component is represented as an optimization problem, which can be solved either by a black-box solver such as Gurobi [Gurobi Optimization, LLC, 2022], or by following steps of a specific algorithm such as cutting-planes [Gomory, 2010]	18
2.1	Known constraints: categorization of related works on the basis of how they formulate the constraints (hard vs soft) and when do they enforce the constraints (training or inference).	22
3.1	NER: Comparison of different training techniques. B: Baseline; CL: Constrained Learning; SCL: Semi-supervised Constrained Learning; CI: Constrained Inference	48
4.1	Decision Boundary learnt by logistic regression guided by MINLOSS. Green vertical line at $\mathbf{x} = 0$ is the initial decision boundary and black vertical line at $\mathbf{x} = -0.55$ is the decision boundary at convergence.	64
4.2	Flow-diagram for our RL Framework	65

4.3	Left: Accuracy vs size of query's solution set (with 95% confidence interval). Right: #givens and #datapoints vs size of query's solution set	71
4.4	Fraction of training samples for which $\arg \max$ of G_ϕ probability distribution is different from the target closest to model prediction. For I-EXPLR, this fraction is 0%	72
5.1	An example Futoshiki Puzzle of size 3×3 and the corresponding graphs. A value of -1 indicates an unassigned variable. Black and red edges are Constraint and Relation edges respectively. The digits 5, 7, 1 in square boxes represent a random 3-permutation of $k_{\max}$, used in multi-valued model for initialization of node embeddings.	84
5.2	Hidden State Update at time-step t : We take a toy graph with 3 nodes, $\{a, b, c\}$, and 2 edge-types, $\{q, r\}$, to illustrate edge-dependent message passing and hidden state update of node b . First, messages along the four edges are calculated as an edge-type dependent function of the sender and receiver hidden state using f_q and f_r . Next, the incoming messages are aggregated by edge-type (<i>e.g.</i> , using attention based mechanism or simple averaging), and the outputs are concatenated to obtain the final message, $m_{t-1}[b]$. The hidden state of node b is updated by function g which takes the previous hidden state $h^{t-1}(b)$, the incoming message $m_{t-1}[b]$, and the initial embedding of the node as its input.	87
5.3	Resource utilization for Futoshiki	100
6.1	An illustration of our framework. Figure on the left shows 4 ground truth constraints that need to be learnt. Blue dots are the only feasible integral points w.r.t. the 4 constraints. Shaded area containing only the dot with green border is the feasible region after we add the cost constraint (dashed line). Figure on the right shows an intermediate scenario while learning. The green-bordered dot (positive) is outside the intermediate 4^{th} constraint and the red dot (negative) is inside the intermediate 3^{rd} constraint. Positive and negative losses encourage the 4^{th} and the 3^{rd} hyperplanes to move in the direction shown by the green and red dotted arrows respectively. . .	111
7.1	Architecture of our Selection Module. The underlying two-layer MLP is same when predicting score for every template.	132
B.1	8-Queens query along with its two possible solution. ¹	188
E.1	Human Intelligence Task on Amazon Mechanical Turk	204

List of Tables

3.1	$g(\mathbf{w})$, $g_1(\mathbf{w})$ and $g_2(\mathbf{w})$ are soft value functions for C , C_1 and C_2 , respectively.	42
3.2	Effect of constrained learning on SRL, with and without constrained inference (CI).	46
3.3	Constraints used in NER task	46
3.4	F-score for different models (mean $\pm$ stdev), along with average number of constraint violations	47
3.5	TypeNet: MAP Scores (in %) and # of constraint violations for different training sizes.	50
4.1	Statistics of datasets. ‘Train’, ‘Test’ and task names are abbreviated. Devset similar to test.	69
4.2	Mean test accuracy ($\pm$ standard error) over three runs for multiplicity aware methods compared with baselines. <i>OS</i> : test queries with only one solution, <i>MS</i> : queries with more than one solution.	70
4.3	Comparing the accuracy of an autoregressive encoder-decoder with a non-autoregressive encoder. For easy reference, we copy from Table 4.2 the mean accuracy obtained by RRN. <i>OS</i> : test queries with only one solution, <i>MS</i> : queries with more than one solution.	75
5.1	Dataset details	93
5.2	Test Data statistics for all three tasks	94
5.3	Range for p for given k and n for GCP data generation	94
5.4	Futoshiki: Mean (Std. dev.) of Board and Pointwise accuracy on different board sizes. MV and BIN correspond to Multi-valued Model and Binarized Model, respectively.	97
5.5	GCP: Mean (Std. dev.) of coloring and pointwise accuracy on graphs with different chromatic number.	97
5.6	Sudoku: Mean (Std. dev.) of board and pointwise accuracy on different board-sizes. Both models trained on 9×9 puzzles	98

5.7	Sudoku: Mean (Std. dev.) of board and pointwise accuracy of models fine-tuned on 24 board-size	98
6.1	Board accuracy and training time for different board sizes of symbolic and visual sudoku (for CombOptNet, “-” denotes time-out after 12 hours) . .	115
6.2	Mean of the vector accuracy ($\mathbf{M}_{\Theta}(x) = \mathbf{y}^*$) and training time over the 10 random datasets in each setting of Random constraints. Number of learnable constraints is twice the number of ground truth constraints. See section D.1 for std. err. over the 10 runs.	117
6.3	Mean accuracy and train-time over 10 runs for various knapsack datasets with different number of items in each instance. For $N = 25, 30$, “-” represents timeout of 12 hours and we evaluate using the latest snapshot of the model obtained within 12 hours of total training. See section D.1 for std. err. over the 10 runs.	118
6.4	Keypoint Matching: Number of train and test samples for datasets with different keypoints	119
6.5	Average point-wise accuracy and training times over 3 runs with different random seeds for varying # of keypoints. Neural + CI denotes ILP inference with known constraints over the cost learnt by neural model. See section D.1 for std. err. over the 3 runs.	120
7.1	Input Features for a prediction $r(s, o)$ from i^{th} template T_i	131
7.2	Micro-F1 score of OxKBC (mean $\pm$ std deviation)	134
7.3	OxKBC vs. rule mining under different scenarios: Overall: when either of the two systems give an explanation; Both: When both give an explanation; OxKBC: When only OxKBC gives an explanation (explained above); Similarly for ‘Rules’. When only one of the systems produces an explanation (last two rows), we ask workers whether it is better or worse than having no explanation. If it is selected as being worse, we mark the other system, producing ‘no explanation’, as better.	136
7.4	OxKBC was preferred to Rule Mining for first two facts, rule mining won in third fact, and none of the explanations were accepted in the last example.	137
7.5	OxKBC explanations from N-GT@1 when the top prediction is correct in real world.	137
7.6	OxKBC explanations for samples from N-GT@1 when the predictions are actually incorrect.	138

7.7	MRR over 100 samples when the probability of the supporting fact in the explanation is reduced. Control MRR reports the MRR when probability of any other related fact in the KB is reduced.	138
A.1	Best hyper-parameters in SRL experiments for different training sizes. . .	183
A.2	Best hyper-parameters in NER for different training sizes in both scenarios: CL and SCL.	184
A.3	Best hyper-parameters in Typenet for different training sizes in both scenarios: CL and SCL.	184
B.1	Seed wise gains of SELECTR over MINLOSS across different random seeds and experiments	193
C.1	Hyperparameters for different models and tasks	196
C.2	Training cost of different models in terms of number of epochs, gradient updates and clock time	197
D.1	Mean $\pm$ std err of the vector accuracy ($\mathbf{M}_{\Theta}(x) = \mathbf{y}^*$) and training time over the 10 random datasets in each setting of Random constraints. Number of learnable constraints is twice the number of ground truth constraints. CombOpt: CombOptNet	200
D.2	Mean $\pm$ std err. of accuracy and train-time over 10 runs for various knapsack datasets with different number of items in each instance. For $N = 25, 30$, TO represents timeout of 12 hours and we evaluate using the latest snapshot of the model obtained within 12 hours of total training. CombOpt: CombOptNet	200
D.3	Mean $\pm$ std error of point-wise accuracy and training times over 3 runs with random seeds for varying number of keypoints. Neural+CI denotes ILP inference with known constraints over the cost learnt by neural model.	200
E.1	Hyper-parameters used for various settings	202