Report from Dagstuhl Seminar 22291

Machine Learning and Logical Reasoning: The New Frontier

Sébastien Bardin[*1], **Somesh Jha**[*2], **and Vijay Ganesh**[*3]

1 **CEA LIST, FR.** `sebastien.bardin@cea.fr`
2 **University of Wisconsin-Madison, US.** `jha@cs.wisc.edu`
3 **University of Waterloo, CA.** `vijay.ganesh@uwaterloo.ca`

Abstract

Machine learning (ML) and logical reasoning have been the two key pillars of AI since its inception, and yet, there has been little interaction between these two sub-fields over the years. At the same time, each of them has been very influential in their own way. ML has revolutionized many sub-fields of AI including image recognition, language translation, and game playing, to name just a few. Independently, the field of logical reasoning (e.g., SAT/SMT/CP/first-order solvers and knowledge representation) has been equally impactful in many contexts in software engineering, verification, security, AI, and mathematics. Despite this progress, there are new problems, as well as opportunities, on the horizon that seem solvable only via a combination of ML and logic.

One such problem that requires one to consider combinations of logic and ML is the question of reliability, robustness, and security of ML models. For example, in recent years, many adversarial attacks against ML models have been developed, demonstrating their extraordinary brittleness. How can we leverage logic-based methods to analyze such ML systems with the aim of ensuring their reliability and security? What kind of logical language do we use to specify properties of ML models? How can we ensure that ML models are explainable and interpretable?

In the reverse direction, ML methods have already been successfully applied to making solvers more efficient. In particular, solvers can be modeled as complex combinations of proof systems and ML optimization methods, wherein ML-based heuristics are used to optimally select and sequence proof rules. How can we further deepen this connection between solvers and ML? Can we develop tools that automatically construct proofs for higher mathematics?

This Dagstuhl seminar seeks to answer these and related questions, with the aim of bringing together the many world-leading scientists who are conducting pioneering research at the intersection of logical reasoning and ML, enabling development of novel solutions to problems deemed impossible otherwise.

Seminar July 17–22, 2022 – http://www.dagstuhl.de/22291
2012 ACM Subject Classification Theory of computation → Automated reasoning; Computing methodologies → Knowledge representation and reasoning; Theory of computation → Logic; Computing methodologies → Machine learning
Keywords and phrases Logic for ML, ML-based heuristics for solvers, SAT/SMT/CP solvers and theorem provers, Security, reliability and privacy of ML-based systems
Digital Object Identifier 10.4230/DagRep.12.7.80

[*] Editor / Organizer

1 Executive Summary

Sébastien Bardin
Somesh Jha
Vijay Ganesh

This Dagstuhl seminar is meant to be the first in a series, bringing together researchers from the two main pillars of AI, namely, logical reasoning and machine learning (ML), with a sharp focus on solver-based testing, analysis, and verification (TAV) methods aimed at improving the reliability and security of ML-based systems, and conversely, the use of ML heuristics in improving the power of solvers/provers. A third, albeit smaller focus is neuro-symbolic reasoning (NSR), that aims to combine the power of ML to learn deep correlations with the ability of solvers to perform logical inference as applied to many domains (including but not limited to math and logic).

While many previous Dagstuhl seminars focus on sub-fields of this particular seminar (SAT, SMT, CP or machine learning), we focus here on the synergies and interplay between all them. Our goal is to deepen the understanding of the connections between learning and reasoning, and draw mutually beneficial research directions.

General context: Bringing ML and Logic Reasoning Closer

Since its very inception, Artificial Intelligence (AI) has largely been divided into two broad fields, namely, machine learning (ML) and logical reasoning, that have developed relatively independent of each other. Each of these sub-fields has had a deep and sustained impact on many topics in computer science and beyond, despite the limited interaction between them over the years. However, in recent years new problems and opportunities have come to fore that point towards combinations of ML and logical reasoning as the way forward [1] [1]. In this seminar, we aim to explore combinations of ML and logical reasoning, under the following three specific themes:

Logic Reasoning for ML. Neural Networks (NN) today are ubiquitous and are being deployed as part of critical civilian and defense infrastructure, business processes, automotive software, and governmental decision-making systems. Unfortunately, despite their efficacy in solving many problems, NNs are brittle, unreliable, and pose significant security/privacy challenges [2]. The question of safety and security of NNs has therefore become a great concern to scientists, companies, and governments. In response to this problem, a nascent field of TAV methods for NNs is developing [3]. Key research directions in this context include logics aimed at symbolically representing NNs and their properties [4], novel solving methods [5], as well as solver-based TAV techniques specifically tailored for NNs [6]. A related set of questions focus on explainability and interpretability of NNs [7]. Finally, researchers are also exploring methods that combine logical reasoning within NN learning processes, with the aim of making them adversarially robust [8, 9]. The seminar aims to bring together leading researchers in these topics, enabling cross-fertilization of ideas at a critical juncture in the development of the field.

[1] It goes without saying that it is infeasible to consider all possible combinations of ML and logical reasoning in this seminar. Hence, we focus primarily on problems inspired by testing, analysis, and verification (TAV) of ML and ML-based heuristics for logic solvers, with some forays into neuro-symbolic (a.k.a., neural-symbolic) AI.

ML for Logic Reasoning. In recent years, there has been considerable effort aimed at developing ML-based heuristics for logic reasoning engines such as SAT, SMT, and CP solvers. The premise of this line of research is that logic solvers are a combination of methods that implement proof rules and ML-based heuristics aimed at optimally selecting, sequencing, and initializing such proof rules [10]. This has led to new efficient solving algorithms that can solve real-world formulas with millions of variables and clauses in them. One of the many questions that will be explored in the seminar is how can we further deepen and strengthen this relation between ML and reasoning methods. Yet another line of research being explored is that of replacing rule-based solvers with NN-based logic reasoning (e.g., NeuroSAT [11]). Finally, methods are being developed to combine rule-based methods with reinforcement learning to automatically prove mathematical conjectures [12]. The seminar aims to foster deeper interaction and collaboration among researchers who are pioneers in this intersection of ML-based methods and logical reasoning.

Neuro-symbolic Reasoning. The field of neuro-symbolic reasoning (NSR) aims to combine NNs with symbolic reasoning for the purposes of improving reasoning for many domains (including but not limited to pure math or logic). While at a high-level the field of NSR and logic solvers (with ML heuristics) may seem similar, they employ very different kinds of techniques and have differing goals [1]. For example, NSR researchers have developed methods for translating logical representations of knowledge into neural networks. Others have developed neuro-symbolic methods for concept-learning, and yet others have recently applied NSR to program synthesis. Can these concept-learning methods be adapted to the setting of logic solvers? Could it be that graph neural network (GNN) based representations of mathematical knowledge are easier to analyze? The seminar aims to bring these two disparate communities closer together, that otherwise rarely interact with each other. In a nutshell, the aim of the seminar is to foster cross-fertilization of ideas between the logic reasoning, TAV, and ML communities.

In-depth Description of Focus Areas

Logic Reasoning for ML. As stated above, the reliability, safety, and security of NNs is a critical challenge for society at large. An example of a specific problem in this context is that of adversarial input generation methods against NNs. Many methods have been proposed to address this question, from randomized defense mechanisms to adversarial training to symbolic analysis of NNs via solvers, such as Reluplex [5] that are specifically designed to reason about NNs with ReLU units. Another line of work proposes verification of Binarized Neural Networks (BNNs) via SAT solvers [6]. These initial forays into reasoning for ML bring to fore new challenges, especially having to do with scalability of solvers for NN analysis. Traditional solver methods that scale well for typical software systems, do not seem to scale well for NNs. For example, it is known that solvers, such as Reluplex, are capable of analyzing NNs with only a few thousand nodes. The pressing question of this area of research then is *"How can we develop methods that enable solvers to scale to NNs with millions of nodes in them?"*

A related question has to do with appropriate logics to represent NNs and their properties. Recent work by Soutedeh and Thakur suggests that NNs can be represented symbolically as piecewise linear functions, even though they may use non-linear activation functions such as ReLU [4]. This suggests that there may be efficient solving methods capable of analyzing very large NNs. Yet another question in this setting is how do logic-based methods aimed at testing and verifying NNs compare against hybrid methods that do not require translation of NNs into logic. What are the tradeoffs in this setting?

Another interesting direction where logic reasoning can play a role is in explainability and interpretability of ML models. While both these questions have been long studied in AI and are closely related, they take particular importance in the context of NNs. We say a ML model is explainable, if there is discernable causal relationship between its input and output. Explanations for the behavior of NN, when presented in symbolic form, can be analyzed and debugged using solvers. Researchers have also developed solver-based xAI methods that aim to provide explanations for behavior of NNs [7]. By contrast, interpretable models are ones that have mathematical guarantees regarding their approximation or generalization errors. Solvers can play a role in this context as well via methods for generating counterfactuals (or adversarial examples) [1].

Strong points:
- *Adversarial attacks and defense mechanisms*
- *Neural network testing, analysis, and verification methods*
- *Piecewise linear symbolic representation of NNs*
- *Solvers for NNs*
- *Logic-guided machine learning*
- *Adversarial training*
- *Logic-based explainability and interpretability of NNs*

ML-based Heuristics for Logic Solvers. In recent years, ML-based methods have had a considerable impact on logic solvers. The key premise of this line of research is that logic solvers are a combination of proof systems and ML-based heuristics aimed at optimally selecting, sequencing, and initializing proof rules with the goal of constructing short proofs (if one exists) [10]. A dominant paradigm in this setting is modeling branching heuristics as RL methods to solve the multi-arm bandit (MAB) problem [10]. While this connection seems quite natural today and MAB-style methods have been shown to be empirically powerful, an important question remains as to why these heuristics are effective for industrial instances. A theoretical answer to this question can open up new connections between ML and logic solvers. Another direction of research that has been explored is solving SAT using NNs, *a la* NeuroSAT [11]. Finally, higher-order theorem provers have been developed recently at Google and elsewhere that combine RL with logic reasoning in order to automatically prove theorems from a variety of mathematical fields [13, 12]. The seminar will focus on these recent developments and the next steps in the research on combinations of ML-based methods with logic reasoning with the goal of achieving greater solver efficiency as well as expressive power.

Strong points:
- *ML-techniques for branching and restarts in SAT, SMT, and CP solvers*
- *Supervised learning methods for splitting and initialization in solvers*
- *NN-based methods for logical reasoning*
- *RL for higher-order theorem proving*

Neuro-symbolic Reasoning. Researchers in neuro-symbolic reasoning (NSR) have been independently developing algorithms that combine ML with symbolic reasoning methods with a slightly different focus than solver and theorem prover developers. NSR research has been focused on concept learning in a broader setting than math or logic, and the cross-fertilization of these ideas with logic-based methods can have deep impact both on NSR as well as solver research [1]. One of the key ideas we plan to explore in this context is that of concept learning, i.e., learning of relations or concepts represented in a logical language directly from data. One interesting direction to explore would be how we can incorporate these methods in logic solvers? Another direction is to explore the synergy between NSR and synthesis of programs from examples. The seminar will focus on bringing NSR and solver researchers closer together, given that they rarely interact in other settings.

Strong points:

- *Concept-learning, with possible applications in higher-order theorem provers*
- *Neuro-symbolic methods for program synthesis*
- *Concept learning for predicate abstraction*

Goals of the Seminar

The aim of this seminar is to bring together the logic reasoning and ML communities, thus shaping and setting the research agenda for ML-based solvers, TAV methods aimed at NNs, and NSR for many years to come.

The seminar will highlight the current challenges with symbolic analysis of NNs, scalability issues with solvers tailored for NNs, state-of-the-art ML-based heuristics for solvers, adapting NSR ideas to the setting of solvers and vice-versa, as well as bring to fore competing TAV methods that don't necessarily rely on symbolic representation of NNs.

Research questions. We highlight some of the main challenges at the intersection of ML and logic reasoning that will be addressed during the seminar from different research perspectives, and discuss how we seek to combine or adapt current techniques to attack them.

- **Symbolic representation of NNs:** Recent work suggests that, while NNs are non-linear functions, they can be effectively modelled symbolically as piecewise linear functions. This is a significant advance since it dramatically simplifies the design of solvers for analyzing NNs. Some of the challenges that remain are algorithmic, i.e., how can NNs be efficiently converted into a symbolic representation.
- **Solvers for NNs:** As of this writing, Reluplex and its successors seem to be among the best solvers for analyzing the symbolic representations of NNs. Unfortunately, these tools scale to NNs with at most a few thousand nodes. There is an urgent need for novel ideas for solving algorithms that enable us to scale to real-world NNs with millions of nodes. Can hybrid methods that combine ML techniques with solvers scale more effectively than pure logic methods?
- **Combining Constraints and NN Learning:** Another set of questions we plan to address is how can we improve the process via which NNs learn using logical constraints. In other words, can the back propagation algorithm be modified to take constraint or domain-specific knowledge into account? Can NNs be combined with logic solvers in a CEGAR-style feedback loop for the purposes of adversarial training?
- **Next steps in ML-based Heuristics for Solvers:** As stated earlier, modern solvers rely in significant ways on ML-based heuristics for their performance. We plan to focus on how we could strengthen this interaction further. For example, are there supervised learning methods for improving the performance of divide-and-conquer parallel SAT solvers. Can we develop ML-based methods for clause sharing in portfolio solvers? How about ML-based restarts and clause deletion policies?
- **Reinforcement learning (RL) and Theorem Provers:** There has been some recent success in combining basic RL methods with reasoning methods in the context of higher-order theorem provers. How can this combination be strengthened further to prove math theorems in a completely automated fashion.

- **Comparison of NN Verification with Testing and Fuzzing Methods:** Researchers have developed a variety of fuzzing methods aimed at NNs. These methods often scale better than verification techniques. On the other hand, unlike verification, testing techniques do not give any guarantees. What are the tradeoffs in this context of complete verification vs. scalability? Can we develop hybrid methods and light-weight verification techniques?
- **Concept Learning and Solvers:** Can we lift the ideas of concept learning from NSR to the setting of solvers, especially in the context of higher-order and combinatorial mathematics?

Synergies. We have also identified the following potential synergies between the ML, Solver, TAV, and NSR communities and expect strong interactions around these points:
- ML researchers in general (and RL in particular) can help refine the ML-based methods used by solver developers;
- Solver developers can propose constraint-based learning strategies for NNs (e.g., combining constraints with gradient-descent in the back propagation algorithm);
- Researchers who work in the space of TAV for NN can benefit greatly by better understanding the realistic security and safety concerns of the ML community;
- Solver developer can substantially benefit by better understanding concept learning from NSR researchers.

Expected results and impact on the research community. One of the core goals of the seminar is to bring together the many different research communities that work in the logic reasoning and ML fields, who unfortunately rarely talk to each other. We believe that the exchange of ideas between them – each with their own methods and perspectives – will help accelerate the future development of combinations of ML and logic reasoning. In terms of concrete outcomes, we believe the workshop is likely to lead to several collaboration projects, especially between members of different communities working on similar or related problems. Common benchmarks and regular meeting forums will also be discussed and we expect for some progress there as well.

References

1 Amel, Kay R. From shallow to deep interactions between knowledge representation, reasoning and machine learning. 13th International Conference Scala Uncertainity Mgmt (SUM 2019), Compiègne, LNCS. 2019.

2 Goodfellow, Ian J and Shlens, Jonathon and Szegedy, Christian. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572. 2014

3 Leofante, Francesco and Narodytska, Nina and Pulina, Luca and Tacchella, Armando. Automated verification of neural networks: Advances, challenges and perspectives. arXiv preprint arXiv:1805.09938. 2018

4 Sotoudeh, Matthew and Thakur, Aditya V. A symbolic neural network representation and its application to understanding, verifying, and patching networks. arXiv preprint arXiv:1908.06223. 2019

5 Katz, Guy and Barrett, Clark and Dill, David L and Julian, Kyle and Kochenderfer, Mykel J. Reluplex: An efficient SMT solver for verifying deep neural networks. In CAV 2017

6 Narodytska, Nina and Kasiviswanathan, Shiva and Ryzhyk, Leonid and Sagiv, Mooly and Walsh, Toby. Verifying properties of binarized deep neural networks. AAAI 2018

7 Samek, Wojciech and Wiegand, Thomas and Müller, Klaus-Robert. . Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. arXiv preprint arXiv:1708.08296. 2017

8 Fischer, Marc and Balunovic, Mislav and Drachsler-Cohen, Dana and Gehr, Timon and Zhang, Ce and Vechev, Martin. Dl2: Training and querying neural networks with logic. ICML 2019

9 LGML: Logic Guided Machine Learning. Scott, Joseph and Panju, Maysum and Ganesh, Vijay. AAAI 2020.

10 Jia Hui Liang, Hari Govind V. K., Pascal Poupart, Krzysztof Czarnecki and Vijay Ganesh. An Empirical Study of Branching Heuristics Through the Lens of Global Learning Rate. In SAT 2017.

11 Selsam, Daniel and Lamm, Matthew and Bünz, Benedikt and Liang, Percy and de Moura, Leonardo and Dill, David L. Learning a SAT solver from single-bit supervision. arXiv preprint arXiv:1802.03685. 2018

12 Kaliszyk, Cezary and Urban, Josef and Michalewski, Henryk and Olšák, Miroslav. Reinforcement learning of theorem proving. In Advances in Neural Information Processing Systems. 2018

13 Bansal, Kshitij and Loos, Sarah and Rabe, Markus and Szegedy, Christian and Wilcox, Stewart. HOList: An environment for machine learning of higher order logic theorem proving. ICML 2019.

2 Table of Contents

3 Planning

Monday July 18: (NeuroSymbolic AI Day)

9:00 am – 9:05 am	Welcome and seminar overview (Vijay Ganesh)
9:05 am – 10:05 am	NeuroSymbolic AI: The 3rd Wave (Artur d'Avila Garcez)
10:30 am – 11:00 am	Two birds with one stone? Successes and lessons in building Neuro-Symbolic systems (Xujie Si)
11:00 am – 12:00 pm	Empower Deep Networks with Combinatorial Algorithms (Georg Martius)
2:00 pm – 2:30 pm	CombOptNet: Fit the Right NP-Hard Problem by Learning Integer Programming Constraints (Anselm Paulus)
2:30 pm – 3:00 pm	Learning Modulo Theories (Matt Fredrikson)
3:30 pm – 4:30 pm	Panel on NeuroSymbolic AI (Artur d'Avila Garcez, Georg Martius, Matt Fredrikson) (Moderator: Xujie Si)

Tuesday July 19: (ML for Logic Day)

9:00 am – 10:00 am	Tutorial on ML for Solvers (Vijay Ganesh)
10:00 am – 10:30 am	Machine Learning for Instance Selection in SMT Solving (Pascal Fontaine)
11:00 am – 11:30 am	CrystalBall: Gazing into the Future of SAT Solving (Kuldeep Meel)
11:30 am – 12:00 pm	The importance of memory for mathematical reasoning (Markus Rabe)
2:00 pm – 2:30 pm	AlphaZero from Games to Mathematics (Alhussein Fawzi)
2:30 pm – 3:00 pm	PL vs. AI: The Case for Automated Code Deobfuscation (Sébastien Bardin)
3:30 pm – 4:00 pm	Machine Learning Algorithm Selection for Logic Solvers (Joseph Scott)
5:00 pm – 6:00 pm	Panel on ML for solvers and theorem provers (Markus Rabe, Kuldeep Meel, Alhussein Fawzi, Pascal Fontaine) (Moderator: Sébastien Bardin)

Wednesday July 20: (Testing, Analysis, Verification of ML Systems)

9:00 am – 10:00 am	Tutorial on Temporal Logic for RL (Rajeev Alur)
10:00 am – 10:30 am	Safety Assurance, ML, and Logic – Some Lessons Learned (Chih-Hong Cheng)
11:00 am – 12:00 pm	Perspectives on NeuroSymbolic AI (Luis C. Lamb)
2:00 pm – 2:30 pm	The Foundations of Deep Learning Verification (Matthew Mirman)
2:30 pm – 3:00 pm	Data Usage across the Machine Learning Pipeline (Caterina Urban)

Thursday July 21: (Testing, Analysis, Verification of ML Systems 2)

9:00 am – 10:00 am	Perspectives on Verified AI (Sanjit Seshia)
10:00 am – 10:30 am	There is Plenty of Room at the Bottom : Verification (and Repair) of Small-scale Learning Models (Armando Tacchella)
11:00 am – 11:30 am	Automated Program Analysis: Revisiting Precondition Inference through Constraint Acquisition (Grégoire Menguy)
11:30 am – 12:00 pm	Verification of DNN Controllers (Rajeev Alur)
2:00 pm – 2:30 pm	The Necessity of Run-time Techniques for Safe ML (Ravi Mangal)
2:30 pm – 3:00 pm	Inductive Proofs for Probabilistic Verification and Opportunities in ML (Sebastian Junges)
3:30 pm – 4:00 pm	CGDTest: Constraint-based Gradient Descent Fuzzer for DNNs (Vineel Nagisetty)
4:00 pm – 4:30 pm	Logic for Adversarial ML (Marc Fisher)

Friday July 22: (Synthesis and ML Day)

9:00 am – 10:00 am	Functional Synthesis via Combination of Learning and Reasoning (Kuldeep Meel)
10:00 am – 10:30 am	Synthesizing Pareto-Optimal Interpretations for Black-box Models (Hazem Torfah)
11:00 am – 12:00 pm	Panel on Testing, Analysis, Verification, Security, and Synthesis of AI (Kuldeep Meel, Armando Tacchella, Rajeev Alur, Matt Frederikson) (Moderator: Hazem Torfah)

4 Overview of Talks

4.1 Formal verification for safe autonomy

Rajeev Alur (University of Pennsylvania – Philadelphia, US)

Autonomous systems interacting with the physical world, collecting data, processing it using machine learning algorithms, and making decisions, have the potential to transform a wide range of applications including medicine and transportation. Realizing this potential requires that the system designers can provide high assurance regarding safe and predictable behavior. This motivates research on formally verifying safety (such as avoidance of collisions) of closed-loop systems with controllers based on learning algorithms. In this talk, I will use the experimental platform of the autonomous F1/10 racing car to highlight research challenges for verifying safety for systems with neural-network-based controllers. Our solution to safety verification, incorporated in the tool Verisig at Penn, builds upon techniques for symbolic computation of the set of reachable states of hybrid (mixed discrete-continuous) systems. The case study consists of training the controller using reinforcement learning in a simulation environment, verifying the trained controller using Verisig, and validating the controller by deploying it on the F1/10 racing car.

4.2 Automated Program Analysis: Revisiting Precondition Inference through Constraint Acquisition

Grégoire Menguy (CEA LIST, FR), Sébastien Bardin (CEA LIST, FR)

Joint work of Grégoire Menguy, Sébastien Bardin, Nadjib Lazaar, Arnaud Gotlieb
Main reference Grégoire Menguy, Sébastien Bardin, Nadjib Lazaar, Arnaud Gotlieb: "Automated Program Analysis: Revisiting Precondition Inference through Constraint Acquisition", in Proc. of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022, pp. 1873–1879, ijcai.org, 2022.
URL https://doi.org/10.24963/ijcai.2022/260

Automated program analysis enables to prove code properties like correctness or incorrectness and more generally to help understanding software. Such methods are usually white-box, i.e., they rely on the code syntax to deduce code properties through logical reasoning. While white-box methods have proven to be very powerful, being used for example at Microsoft, Facebook and Airbus, they also suffer from some limitations. First, they need the source code, which is not always available (e.g., proprietary software, malware). Second, the code size and the complexity of data structures manipulated degrade their efficiency drastically. Third, they are highly impacted by syntactic code complexity, which can be amplified by optimizations (improving code speed and memory consumption) and obfuscation (impeding end-users from extracting intellectual property contained in the code).

In this talk, we propose a new method, completely black-box, which infers code annotations from observed code executions only. Indeed, annotations, under the form of function pre/postconditions, are crucial for many software engineering and program verification applications. Unfortunately, they are rarely available and must be retrofit by hand. Thus, we explore how Constraint Acquisition (CA), a learning framework from Constraint Programming, can be leveraged to automatically infer program preconditions in a black-box manner, from input-output observations. We propose PreCA, the first ever framework based on active constraint acquisition dedicated to infer memory-related preconditions. PreCA overpasses prior techniques based on program analysis and formal methods, offering well-identified guarantees and returning more precise results in practice.

References

1 Bessiere, C., Koriche, F., Lazaar, N., O'Sullivan, B. (2017). Constraint acquisition. Artificial Intelligence, 244, 315-342.
2 Rossi, F., Van Beek, P., Walsh, T. (Eds.). (2006). Handbook of constraint programming. Elsevier.
3 Hoare, C. A. R. (1969). An axiomatic basis for computer programming. Communications of the ACM, 12(10), 576-580.
4 Dijkstra, E. W. (1968). A constructive approach to the problem of program correctness. BIT Numerical Mathematics, 8(3), 174-186.
5 Floyd, R. W. (1993). Assigning meanings to programs. In Program Verification (pp. 65-81). Springer, Dordrecht.
6 Menguy, G., Bardin, S., Lazaar, N., Gotlieb, A. Automated Program Analysis: Revisiting Precondition Inference through Constraint Acquisition. IJCAI 2022

4.3 PL vs. AI: The Case of Automated Code Deobfuscation

Sébastien Bardin (CEA LIST, FR)

Joint work of Grégoire Menguy, Sébastien Bardin, Richard Bonichon, Cauim de Souza Lima
Main reference Grégoire Menguy, Sébastien Bardin, Richard Bonichon, Cauim de Souza Lima: "Search-Based Local Black-Box Deobfuscation: Understand, Improve and Mitigate", in Proc. of the CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 – 19, 2021, pp. 2513–2525, ACM, 2021.
URL https://doi.org/10.1145/3460120.3485250

In this talk, we slightly deviate from the main *"Logic # Machine Learning"* topic of the seminar, to consider another closely related inference vs. deduction scheme, namely the link between Program Analysis (PL) and Artificial Intelligence (AI), with a focus on the case of reverse engineering attacks and code deobfuscation. Especially, we discuss how PL and AI both hep in the field, highlight their complementarity strengths and weaknesses and draw lines for future research directions.

Reverse attacks consist in trying to retrieve sensitive information (e.g., secret data, secret algorithms, sensitive business details, etc.) form a program under analysis. We usually consider a very strong attacker with unlimited access to the executable code. Hence, the attacker can for example perform static analysis other the executable, trace the execution, rewrite part of the binary, etc. The goal for the defender is to delay as much as possible the information retrieval, through the use of so-called obfuscation techniques aiming at making the code "hard to understand". We call deobfuscation the effort of removing obfuscations from a program, or helping a reverse engineer to understand an obfuscated program.

Since the middle of the 2010's, several authors managed to adapt PL techniques to perform deobfuscation, with very strong and unexpected results other standard protections. Still, as these *white-box* attacks are based on deduction from the code syntax, they can at some point be fooled by dedicated protections aiming to increase the syntactic complexity of the code in such ways that program analyzer become ineffective.

More recently, *black-box* attacks, based on the observations of input-output relationships together with AI-based synthesis methods in order to rebuild a simple view of the beahviour of some obfuscated parts of the code, show very effective against certain kinds of local obfuscations – even anti-PL obfuscations. Still, these methods are sensitive to semantic complexity, and we show how dedicated protections can take advantage of that.

Finally, as future direction, it seems natural to try combining these dual trends (AI & PL, deduction & inference, blackbox & whitebox) into some combined form of hybrid attacks. From a more theoretical point of view, we could also see this problem as an instance of "AI vs. PL", as PL techniques are also used for the protection side, with questions such as how to train an AI to bypass code protections, or how to create code resistant to AI-augmented attackers.

References

1 Grégoire Menguy, Sébastien Bardin, Richard Bonichon, Cauim de Souza Lima: Search-Based Local Black-Box Deobfuscation: Understand, Improve and Mitigate. CCS 2021
2 Sébastien Bardin, Robin David, Jean-Yves Marion: Backward-Bounded DSE: Targeting Infeasibility Questions on Obfuscated Codes. Symposium on Security and Privacy 2017
3 Mathilde Ollivier, Sébastien Bardin, Richard Bonichon, Jean-Yves Marion: How to kill symbolic deobfuscation for free (or: unleashing the potential of path-oriented protections). ACSAC 2019

4 Mathilde Ollivier, Sébastien Bardin, Richard Bonichon, Jean-Yves Marion. Obfuscation: Where Are We in Anti-DSE Protections? In Proceedings of the 9th Software Security, Protection, and Reverse Engineering Workshop (SSPREW 2019).
5 Sebastian Schrittwieser, Stefan Katzenbeisser, Johannes Kinder, Georg Merz- dovnik, and Edgar Weippl. 2016. Protecting Software Through Obfuscation: Can It Keep Pace with Progress in Code Analysis? ACM Comput. Surv. 49, 1, Article 4 (2016)
6 Babak Yadegari, Brian Johannesmeyer, Ben Whitely, and Saumya Debray. A Generic Approach to Automatic Deobfuscation of Executable Code. In Symposium on Security and Privacy, 2015.
7 Tim Blazytko, Moritz Contag, Cornelius Aschermann, and Thorsten Holz. Syntia: Synthesizing the Semantics of Obfuscated Code. In Usenix Security. 2017

4.4 Safety Assurance, ML, and Logic – Some Lessons Learned

Chih-Hong Cheng (Fraunhofer IKS – München, DE)

Main reference Chih-Hong Cheng, Chung-Hao Huang, Thomas Brunner, Vahid Hashemi: "Towards Safety Verification of Direct Perception Neural Networks", in Proc. of the 2020 Design, Automation & Test in Europe Conference & Exhibition, DATE 2020, Grenoble, France, March 9-13, 2020, pp. 1640–1643, IEEE, 2020.
URL https://doi.org/10.23919/DATE48585.2020.9116205

In this talk, I summarize some of my experiences in engineering ML in safety-critical applications. Within the industry, the emerging consensus is that one requires a systematic & holistic approach to address all potential problems that might occur in the complete life cycle. One can use logic and theorem-proving to tighten the "leap of faith" in the safety argumentation. For formal verification of DNN in autonomous driving, the real challenge lies in creating an implicit specification that characterizes the operational design domain. We use an assume-guarantee reasoning approach, where we learn the operational design domain via abstracting the feature vectors collected by the training data. The formal proof is conditional to the assumption that any input in the operational domain falls inside the abstraction. The abstraction is then deployed in the field as an OoD detector.

4.5 Neurosymbolic AI

Artur d'Avila Garcez (City – University of London, GB), Luis C. Lamb (Federal University of Rio Grande do Sul, BR)

Main reference Artur S. d'Avila Garcez, Luís C. Lamb: "Neurosymbolic AI: The 3rd Wave", CoRR, Vol. abs/2012.05876, 2020.
URL https://arxiv.org/abs/2012.05876

Current advances in Artificial Intelligence (AI) and Machine Learning (DL) have achieved unprecedented impact across research communities and industry. Nevertheless, concerns around trust, safety, interpretability and accountability of AI were raised by influential thinkers. Many identified the need for well-founded knowledge representation and reasoning to be integrated with Deep Learning (DL). Neural-symbolic computing has been an active area of research for many years seeking to bring together robust learning in neural networks with reasoning and explainability by offering symbolic representations for neural models. In [5], recent and early research in neurosymbolic AI is analysed with the objective of identifying the most important ingredients of neurosymbolic AI systems. Our focus is on research that integrate in a principled way neural network learning with symbolic knowledge representation and logical reasoning. Insights from the past 20 years of research in neural-symbolic computing were discussed and shown to shed new light onto the increasingly prominent role of safety, trust, interpretability and accountability of AI. We also identify promising directions and challenges for the next decade of AI research from the perspective of neural-symbolic computing, commonsense reasoning and causal explanation.

Over the past decade, AI and in particular DL has attracted media attention, has become the focus of increasingly large research endeavours and changed businesses. This led to influential debates on the impact of AI in academia and industry [3]. It has been claimed that deep learning caused a paradigm shift not only in AI, but in several Computer Science fields, including speech recognition, computer vision, image understanding, natural language processing (NLP), and machine translation [2]. Others have argued eloquently that the building of a rich AI system, semantically sound, explainable and ultimately trustworthy, will require a sound reasoning layer in combination with deep learning. Parallels have been drawn between Daniel Kahneman's research on human reasoning and decision making, reflected in his book "Thinking, Fast and Slow [1], and so-called "AI systems 1 and 2, which would in principle be modelled by deep learning and symbolic reasoning, respectively.

We seek to place 20 years of research in the area of neurosymbolic AI, known as neural-symbolic integration, in the context of the recent explosion of interest and excitement around the combination of deep learning and symbolic reasoning. We revisit early theoretical results of fundamental relevance to shaping the latest research, such as the proof that recurrent neural networks can compute the semantics of logic programs, and identify bottlenecks and the most promising technical directions for the sound representation of learning and reasoning in neural networks. As well as pointing to the various related and promising techniques, such as [4], we aim to help organise some of the terminology commonly used around AI, ML and DL. This is important at this exciting time when AI becomes popularized among researchers and practitioners from other areas of Computer Science and from other fields altogether: psychology, cognitive science, economics, medicine, engineering and neuroscience, to name a few.

The first wave of AI in the 1980's was symbolic – based on symbolic logic and logic programming, and later Bayesian networks; the second wave of AI in the 2010's was neural (or connectionist), based on deep learning. Having lived through both waves and having seen the contributions and drawbacks of each technology, we argue that the time is right for the third wave of AI: neurosymbolic AI. Specifically, we summarise the current debate around neurons vs. symbols from the perspective of the long-standing challenges of variable grounding and commonsense reasoning. We survey some of the prominent forms of neural-symbolic integration. We address neural-symbolic integration from the perspective of distributed and localist forms of representation, and argue for a focus on logical representation based on the assumption that representation precedes learning and reasoning. We delve into the fundamentals of current neurosymbolic AI methods and systems and identify promising aspects of neurosymbolic AI to address exciting challenges for learning, reasoning and explainability. Finally, based on all of the above, we propose the list of ingredients for neurosymbolic AI and discuss promising directions for future research to address the challenges of AI.

References

1 D. Kahneman, Thinking, Fast and Slow, 2011, Farrar, Straus and Giroux, New York.
2 Y. LeCun and Y. Bengio and G. Hinton, Deep Learning, Nature 521(7553):436-444, 2015.

3 G. Marcus, The Next Decade in AI: Four Steps Towards Robust Artificial Intelligence, CoRR abs/1801.00631, 2020.

4 Son N. Tran and Artur d'Avila Garcez, Logical Boltzmann Machines, CoRR abs/2112.05841, 2021. https://arxiv.org/abs/2112.05841.

5 Artur d'Avila Garcez and Luis C. Lamb, Neurosymbolic AI: The 3rd Wave, CoRR abs/2012.05876, 2020. https://arxiv.org/abs/2012.05876.

4.6 AlphaZero from games to maths

Alhussein Fawzi (Google DeepMind – London, GB)

Main reference Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J. R. Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, David Silver, Demis Hassabis, Pushmeet Kohli: "Discovering faster matrix multiplication algorithms with reinforcement learning", Nat., Vol. 610(7930), pp. 47–53, 2022.
URL https://doi.org/10.1038/s41586-022-05172-4

In this talk, I will describe how we can extend AlphaZero, a reinforcement learning agent developed for playing games such as Go and Chess, to mathematical problems. I will go in detail through the different components of AlphaZero, and focus on some of the challenges of applying it to mathematical problems. To illustrate my talk, I will focus precisely on "decomposition problems", where the task is to decompose a hard mathematical object (e.g., a tensor) into a sum of atoms (e.g., rank one tensors).

4.7 Logic & Adversarial Machine Learning

Marc Fischer (ETH Zürich, CH)

Joint work of Christian Sprecher, Anian Ruoss, Dimitar I. Dimitrov, Mislav Balunović, Timon Gehr, Gagandeep Singh, Dana Drachsler-Cohen, Ce Zhang, Martin Vechev
Main reference Dana Drachsler-Cohen Marc Fischer Mislav Balunović: "DL2: Training and Querying Neural Networks with Logic", in Proc. of the International Conference on Machine Learning, 2019.
URL https://www.sri.inf.ethz.ch/publications/fischer2019dl2

The discovery of adversarial examples, small semantic-preserving perturbations that mislead neural networks, highlighted the need to study the robustness of machine learning systems. In this talk, I discuss three perspectives connecting this study of robustness with logic:

- First, how can we use techniques for finding and defending against adversarial examples to query and train neural networks with logic specifications?
- Second, how can we leverage relations between different inputs and input specifications in a robustness analysis based on abstract interpretation – the symbolic propagation of input sets through programs?
- Third, how can we combine these techniques to enforce and verify notions of individual fairness?

References

1 Anian Ruoss, Mislav Balunovic, Marc Fischer, Martin T. Vechev: Learning Certified Individually Fair Representations. NeurIPS 2020

2 Marc Fischer, Christian Sprecher, Dimitar I. Dimitrov, Gagandeep Singh, Martin T. Vechev: Shared Certificates for Neural Network Verification. CAV 2022
3 Marc Fischer, Mislav Balunovic, Dana Drachsler-Cohen, Timon Gehr, Ce Zhang, Martin T. Vechev: DL2: Training and Querying Neural Networks with Logic. ICML 2019

4.8 Learning Modulo Theories: Leveraging Theory Solvers for Machine Learning

Matt Fredrikson (Carnegie Mellon University – Pittsburgh, US)

A recently proposed class of techniques, which aim to integrate solver layers within Deep Neural Networks (DNNs), has shown considerable promise in bridging a long-standing gap between inductive learning and symbolic reasoning techniques. This approach brings the capabilities of a decision procedure to a learned model, both during training and inference. Such an approach is particularly useful in solving problems that have both a perceptual as well as logical or combinatorial sub-tasks. Statistical learning excels at the perceptual, while progress in solver technology continues to open new horizons for the logical.

We will present a new framework, ERM(ϕ), and an associated set of methods for integrating solver layers that encompass a broad range of symbolic knowledge into an ML system. Using this framework, we demonstrate several fundamental challenges and opportunities for this direction. Further, we provide a set of algorithms for computing the forward (inference) and backward (training) passes of a DNN layer that makes calls to an SMT solver, with few restrictions on the user-provided constraints that the solver can query. For example, the theory solver does not need to be differentiable. The talk will conclude by giving an overview of an implementation of our approach within Pytorch, using Z3 to solve constraints, and show how to construct vision and natural language models that incorporate symbolic knowledge during training and inference, and can outperform conventional models – especially in settings where training data is limited or when the cost of fine-grained labeling is prohibitive.

4.9 Opportunities for Neurosymbolic Approaches in the Context of Probabilistic Verification

Sebastian Junges (Radboud University Nijmegen, NL)

Joint work of Steven Carr, Nils Jansen, Ufuk Tocu, Kevin Batz, Mingshuai Chen, Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja

In this overview, we outline some challenges for neurosymbolic approaches in the context of probabilistic verification. In short, probabilistic verification aims to show that a specification holds on a system with probabilistic uncertainties. Such systems can be modelled as probabilistic programs, as stochastic Petri nets, as Bayesian dynamic networks, or any other description language to describe a Markov models. One step beyond verification is synthesis, in this context often policy synthesis, in which the goal is to find policies for agents making decisions under uncertainty, such that the joint behavior of agent and environment satisfies the given specification. We discuss two directions: *"Solver Inside"* and *"Learning Inside"*.

For solver inside, we discuss shielding in reinforcement learning [1]. In particular, we report on shielding for partially observable Markov decision processes and the integration with state-of-the-art deep reinforcement learning [2]. We briefly discuss how shields are computed by an iterative application of SAT-solvers [3]. We discuss the framework and the relative strengths and weaknesses of addings shields in sparse-reward settings. Among others, we show how bootstrapping with a shield can help guide the learning process.

For learning inside, we consider various inductive synthesis frameworks. We may aim to learn inductive invariants for probabilistic models and programs, alike to learning inductive invariants for deterministic programs. A major challenge over learning deterministic invariants is the continuous search space. While results for property-directed-reachability (PDR) on Markov decision processes are mixed [4], the use of a CEGIS-style loop are more promissing [5]. It remains an important challenge how to guess the right form of templates. Data-driven approaches may be helpful. We furthermore briefly discuss inductive synthesis for policies described by small finite-state controllers [6]. Data-driven approaches to come up with good, diverse policies will be able to boost the state-of-the-art.

References

1 Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, Ufuk Topcu: *Safe Reinforcement Learning via Shielding.* AAAI 2018: 2669-2678.
2 Steven Carr, Nils Jansen, Sebastian Junges, Ufuk Topcu: *Safe Reinforcement Learning via Shielding for POMDPs.* CoRR abs/2204.00755 (2022)
3 Sebastian Junges, Nils Jansen, Sanjit A. Seshia: Enforcing Almost-Sure Reachability in POMDPs. CAV 2021: 602-625
4 Kevin Batz, Sebastian Junges, Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, Philipp Schröer: *PrIC3: Property Directed Reachability for MDPs.* CAV 2020: 512-538
5 Kevin Batz, Mingshuai Chen, Sebastian Junges, Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja: *Probabilistic Program Verification via Inductive Synthesis of Inductive Invariants.* CoRR abs/2205.06152 (2022)
6 Roman Andriushchenko, Milan Ceska, Sebastian Junges, Joost-Pieter Katoen: *Inductive synthesis of finite-state controllers for POMDPs.* UAI 2022: 85-95

4.10 From Learning to Reasoning in Neurosymbolic AI

Luis C. Lamb (Federal University of Rio Grande do Sul, BR)

Neurosymbolic AI aims to bring together the statistical nature of machine learning and the logical essence of reasoning in AI systems. Such integration demands a shift as regards research methodology, since the connectionist and symbolic schools of AI have been developed under distinct technical foundations over the last 50 years [1, 4]. Nonetheless, leading technology companies and research groups have put forward agendas for the development of the field, as modern AI systems require sound reasoning, interpretability, and improved explainability [7, 5]. Moreover, AI and deep learning researchers have also pointed out that Neurosymbolic AI is one of *"the most promising approach to a broad AI [..], that is, a bilateral AI that combines methods from symbolic and sub-symbolic AI"*[2]. In this talk, we highlight how the evolution of Neurosymbolic AI research results can lead to applications and novel developments towards building robust, explainable AI systems. We summarize how Neurosymbolic AI evolved over the years and how it might contribute to improved explainability and the effective integration of learning and reasoning in the construction of robust AI systems. This talk is motivated by the evolution of our work on the integration of modal, temporal, and intuitionistic logics and neural learning [4]. Over the years, we showed that the proper integration of logical methods and neural learning can lead to applications in classical benchmarks in multiagent systems [3], modelling the evolution of software requirements specifications, and possibly to a better understanding of the learnability of rich graph-based and optimization problems [1, 6].

References

1 Artur d'Avila Garcez and Luís C. Lamb. Neurosymbolic AI: The 3rd Wave. CoRR, abs/2012.05876. 2020.
2 Sepp Hochreiter. Toward a broad AI. CACM 2022.
3 Ronald Fagin, Joseph Y. Halpern, Yoram Moses and Moshe Y. Vardi. Reasoning About Knowledge. MIT Press, 1995.
4 Artur S. d'Avila Garcez, Luís C. Lamb and Dov M. Gabbay. Neural-Symbolic Cognitive Reasoning. Cognitive Technologies. 2009
5 Leslie Valiant. Probably approximately correct: nature's algorithms for learning and prospering in a complex world. Basic Books (AZ), 2013
6 Luís C. Lamb, Artur S. d'Avila Garcez, Marco Gori, Marcelo O. R. Prates, Pedro H. C. Avelar and Moshe Y. Vardi. Graph Neural Networks Meet Neural-Symbolic Computing: A Survey and Perspective. IJCAI 2020.
7 Gary Marcus. The Next Decade in AI: Four Steps Towards Robust Artificial Intelligence. CoRR, abs/2002.06177, 2020.

4.11 The Necessity of Run-time Techniques for Safe ML (and how to deal with the pitfalls)

Ravi Mangal (Carnegie Mellon University – Pittsburgh, US)

Joint work of Ravi Mangal, Matt Fredrikson, Aymeric Fromherz, Klas Leino, Bryan Parno, Corina Păsăreanu, Chi Zhang
Main reference Klas Leino, Aymeric Fromherz, Ravi Mangal, Matt Fredrikson, Bryan Parno, Corina Păsăreanu: "Self-Correcting Neural Networks For Safe Classification", arXiv, 2021.
URL https://doi.org/10.48550/ARXIV.2107.11445

Neural networks are increasingly being used as components in systems where safety is a critical concern. Although pre-deployment verification of these networks with respect to required specifications is highly desirable, the specifications are not always amenable to verification. In particular, adversarial robustness, a popular specification, requires that a neural network f exhibit local robustness at every input x in the support of its input data distribution D. Local robustness at an input x is the property that $\forall x'.||x - x'|| \leq \epsilon \Rightarrow f(x) = f(x')$. Unfortunately, neither the distribution D nor its support are known in advance. We advocate for the use of run-time or inference-time (i.e., post-deployment) checks to deal with such distribution-dependent specifications. For instance, to ensure adversarial robustness, a network should be used for prediction only if it passes a local robustness check at run-time, otherwise it should abstain from prediction.

While run-time checks can ensure that neural networks do not misbehave, each abstention incurs a cost since one has to default to an expensive fall-back mechanism (typically, human decision-makers). For run-time checks that encode a class of constraints called *safe-ordering constraints*, we propose a mechanism for repairing the outputs of a neural network whenever the run-time check fails. These constraints relate requirements on the order of the classes output by a classifier to conditions on its input. Though local robustness cannot be encoded as a safe-ordering constraint, this fragment is expressive enough to encode various interesting examples of neural network safety specifications from the literature.

Our repair mechanism is based on a self-repairing layer which performs constraint solving and provably yields safe outputs regardless of the characteristics of the network input. We compose this layer with an existing neural network to construct a self-repairing network (SR-Net), and show that in addition to providing safe outputs, the SR-Net is guaranteed to preserve the classification accuracy of the original network whenever possible. Our approach is independent of the size and architecture of the neural network used for classification, depending only on the specified property and the dimension of the network's output; thus it is scalable to large state-of-the-art networks. We show that our approach can be optimized for a GPU, introducing run-time overhead of less than 1ms on current hardware – even on large, widely-used networks containing hundreds of thousands of neurons and millions of parameters. Designing a run-time repair mechanism to handle failures of local robustness checks is an interesting direction for future research.

References

1 Klas Leino, Aymeric Fromherz, Ravi Mangal, Matt Fredrikson, Bryan Parno, and Corina Păsăreanu. *Self-Correcting Neural Networks for Safe Classification*. 5th International Workshop on Software Verification and Formal Methods for ML-Enables Autonomous Systems (FoMLAS), 2022.

2 Klas Leino, Chi Zhang, Ravi Mangal, Matt Fredrikson, Bryan Parno, and Corina Păsăreanu. *Degradation Attacks on Certifiably Robust Neural Networks*. Transactions on Machine Learning Reasearch (TMLR), 2022.

4.12 The importance of memory for mathematical reasoning

Markus Rabe (Google – Mountain View, US)

License Creative Commons BY 4.0 International license
© Markus Rabe
Joint work of Markus Rabe, Christian Szegedy, Yuhuai Tony Wu, DeLesley Hutchins, Charles Staats, and others
Main reference Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, Christian Szegedy: "Memorizing Transformers", CoRR, Vol. abs/2203.08913, 2022.
URL https://doi.org/10.48550/arXiv.2203.08913

In this talk I was discussing astonishing progress in using large language models for mathematical reasoning. One of the main bottlenecks at the moment is the ability to process long documents (books, papers, ...) at once instead of looking at small (page-length) snippets of the data. Our paper on Memorizing Transformers opens a path to equipping existing large language models with the ability to process book-length data, which we improves the performance of large language models on code and mathematical data significantly.

4.13 Blackbox Differentiation: Empower Deep Networks with Combinatorial Algorithms

Georg Martius (MPI für Intelligente Systeme – Tübingen, DE), Anselm Paulus (MPI für Intelligente Systeme – Tübingen, DE)

Joint work of Marin Vlastelica Pogancic, Anselm Paulus, Vít Musil, Georg Martius, Michal Rolínek
Main reference Marin Vlastelica Pogancic, Anselm Paulus, Vít Musil, Georg Martius, Michal Rolínek: "Differentiation of Blackbox Combinatorial Solvers", in Proc. of the 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020, OpenReview.net, 2020.
URL https://openreview.net/forum?id=BkevoJSYPB

Machine Learning has achieved great successes on solving problems that seemed unsolvable just a decade ago. Examples are the mastering of the game of Go, automatic machine translation, and learning in-hand manipulation with a robotic hand. Besides many technical innovations, these advances have been enabled by two main ingredients: highly flexible differentiable function approximators (deep networks) and huge amounts of data. While deep networks can extract very complicated patterns from data, there is a certain sense of dissatisfaction when it comes to their performance on tasks with combinatorial or algorithmic complexity. For example, think of learning to find the shortest path in an environment when provided only with raw birds-eye maps (images). Current, deep networks can learn this task on maps they were trained on, but perform poorly on new maps. The reason is that part of the problem has an algorithmic nature: the same shortest path algorithm works on all maps, if suitably represented as a graph. However a normal deep network cannot perform the same computations and thus can only learn to imitate the process.

The next big step for researchers in machine learning and artificial intelligence is to enhance the ability of the methods to reason. This sentiment was for example expressed by Battaglia et. al. [1] who advocate that "combinatorial generalization must be a top priority for AI".

Importantly, there are decades worth of research contributions in graph algorithms and discrete optimization. We have optimal runtimes for sorting algorithms, clever tricks for various algorithmic problems over graphs/networks such as for shortest path or various cuts or matching-based problems. In other words, when faced with combinatorial or algorithmic problems in isolation and with a clean specification, we already have very strong methods for solving them. This should not be ignored.

While there is some level of success in designing deep learning architectures with "algorithmic behavior", the classical methods are still miles ahead when it comes to performance in purely combinatorial setups. We believe the right approach is to build bridges between the two disciplines so that progress can freely flow from one to another. In that spirit, we would rephrase the earlier sentiment as "merging techniques from combinatorial optimization and deep learning must be a top priority for AI".

We have recently developed a method [2] that allows to embed a large class of combinatorial algorithms in deep neural networks while maintaining the usual training procedure unchanged. In the talk, I will explain the fundamental problem that we had to overcome and show examples of what can be done with the new architecture. This includes the shortest path problem on raw images [2], finding correspondences in pairs of images [3], and directly optimizing for rank-based loss functions [4].

We have two blog-posts on this topic:
`https://towardsdatascience.com/the-fusion-of-deep-learning-and-combinatorics-4d0112a74fa7`
`https://towardsdatascience.com/rambo-ranking-metric-blackbox-optimization-36811a5f52dd`

References

1 P. Battaglia et al. *Relational inductive biases, deep learning, and graph networks.* https://arxiv.org/abs/1806.01261, 2018

2 M. Vlastelica, A. Paulus, V. Musil, G. Martius, and M. Rolínek. *Differentiation of blackbox combinatorial solvers.* In International Conference on Learning Representations, ICLR, 2020.

3 M. Rolínek, P. Swoboda, D. Zietlow, A. Paulus, V. Musil, and G. Martius. *Deep graph matching via blackbox differentiation of combinatorial solvers.* In European Conference on Computer Vision ECCV, 2020

4 M. Rolínek, V. Musil, A. Paulus, M. Vlastelica, C. Michaelis, and G. Martius. *Optimizing ranking-based metrics with blackbox differentiation.* In Conference on Computer Vision and Pattern Recognition, CVPR, 2020

4.14 Democratizing SAT Solving

Kuldeep S. Meel (National University of Singapore, SG)

Joint work of Mate Soos, Raghav Kulkarni, Kuldeep S. Meel
Main reference Mate Soos, Raghav Kulkarni, Kuldeep S. Meel: "CrystalBall: Gazing in the Black Box of SAT Solving", in Proc. of the Theory and Applications of Satisfiability Testing – SAT 2019 – 22nd International Conference, SAT 2019, Lisbon, Portugal, July 9-12, 2019, Proceedings, Lecture Notes in Computer Science, Vol. 11628, pp. 371–387, Springer, 2019.
URL https://doi.org/10.1007/978-3-030-24258-9_26

Boolean satisfiability is a fundamental problem in computer science with a wide range of applications including planning, configuration management, design and verification of software/hardware systems. The annual SAT competition continues to witness impressive improvements in the performance of the winning SAT solvers largely thanks to the development of new heuristics arising out of intensive collaborative research in the SAT community. Modern SAT solvers achieve scalability and robustness with complex heuristics that are challenging to understand and explain. Consequently, the development of new algorithmic insights has been largely restricted to experts in the SAT community.

I will describe our project that aims to democratize the design of SAT solvers. In particular, our project, called CrystalBall, seeks to develop a framework to provide white-box access to the execution of SAT solver that can aid both SAT solver developers and users to synthesize algorithmic heuristics for modern SAT solvers? We view modern conflict-driven clause learning (CDCL) solvers as a composition of classifiers and regressors for different tasks such as branching, clause memory management, and restarting, and we aim to provide a data-driven automated heuristic design mechanism that can allow experts in domains outside SAT community to contribute to the development of SAT solvers.

4.15 Functional Synthesis – An Ideal Meeting Ground for Formal Methods and Machine Learning

Kuldeep S. Meel (National University of Singapore, SG)

Joint work of Priyanka Golia, Subhajit Roy, Kuldeep S. Meel, Friedrich Slivovsky
Main reference Priyanka Golia, Subhajit Roy, Kuldeep S. Meel: "Manthan: A Data-Driven Approach for Boolean Function Synthesis", in Proc. of the Computer Aided Verification – 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21-24, 2020, Proceedings, Part II, Lecture Notes in Computer Science, Vol. 12225, pp. 611–633, Springer, 2020.
URL https://doi.org/10.1007/978-3-030-53291-8_31

Don't we all dream of the perfect assistant whom we can just tell what to do and the assistant can figure out how to accomplish the tasks? Formally, given a specification $F(X, Y)$ over the set of input variables X and output variables Y, we want the assistant, aka functional synthesis engine, to design a function G such that $F(X, G(X))$ is true. Functional synthesis has been studied for over 150 years, dating back Boole in 1850's and yet scalability remains a core challenge. Motivated by progress in machine learning, we design a new algorithmic framework Manthan, which views functional synthesis as a classification problem, relying on advances in constrained sampling for data generation, and advances in automated reasoning for a novel proof-guided refinement and provable verification. The significant performance improvements call for interesting future work at the intersection of machine learning, constrained sampling, and automated reasoning.

References

1 Priyanka Golia, Subhajit Roy, and Kuldeep S. Meel. Manthan: A data-driven approach for boolean function synthesis. CAV 2020
2 Priyanka Golia, Subhajit Roy, and Kuldeep S. Meel. Program synthesis as dependency quantified formula modulo theory. In IJCAI 2021
3 Priyanka Golia, Friedrich Slivovsky, Subhajit Roy, and Kuldeep S. Meel. Engineering an efficient boolean functional synthesis engine. In ICCAD 2021

4.16 Verifiable Neural Networks: Theoretical Capabilities and Limitations

Matthew Mirman (ETH Zürich, CH)

Famously, deep learning has become an integral part of many high stakes applications, from autonomous driving to health care. As the discovery of vulnerabilities and flaws in these models has become frequent, so has the interest in ensuring their robustness and reliability. In recent years, many methods have been developed to build deep learning models amenable to analysis with efficient formal methods. However, these techniques, known together as provability training, have failed to produce models with nearly the empirical quality as traditional training. This stagnation has opened up questions as to the theoretical foundations of provability training. In this talk I will explain our theoretical results on both the possibility and impossibility of constructing verifiable neural networks. To motivate continued search for provable training methods, I will present our possibility result: a stronger form of the universal approximation theorem for the case of interval-certifiable neural networks. To begin to explain the barriers to provable training, I will present our impossibility results: (i) that for any neural network classifying just three points, there is a valid specification over these points that interval analysis can not prove, and (ii) given any radius, there is a set of points that no one-hidden-layer network can be proven to interval-robustly classify.

4.17 CGDTest: Constraint-based Gradient Descent Fuzzer for DNNs

Vineel Nagisetty (Borealis AI – Toronto, CA)

Joint work of Vineel Nagisetty, Guanting Pan, Piyush Jha, Dhananjay Ashok, Laura Graves, Christopher Srinivasa, Vijay Ganesh

In this work we propose a new Deep Neural Network (DNN) testing algorithm, called the Constrained Gradient Descent (CGD) method, and an implementation we call CGDTest aimed at exposing security issues such as testing for adversarial robustness and fairness in DNNs. Our CGD algorithm is a gradient-descent (GD) method, with the twist that the user can also specify logical properties that characterize a specific type of input that they want. This functionality allows us to specify constraints so as to test DNNs for standard Lp ball-based adversarial robustness as well as other properties such as non-standard adversarial robustness and individual fairness. We perform extensive experiments where we use CGDTest to test for both standard and non-standard adversarial robustness in the vision domain, adversarial robustness in the NLP domain, and individual fairness in the tabular domain, comparing against 18 state-of-the-art methods over the 3 domains. Our results indicate that CGDTest is comparable to state-of-the-art tools in testing for standard definitions of robustness and fairness, and is significantly superior in testing for non-standard robustness, with improvements in PAR2 score of over 1500% in some cases over the next best tool. Our evaluation shows that CGD method outperforms all other methods in terms of scalability (i.e., can be applied to very large real-world models with millions of parameters), expressibility (i.e., test for a variety of properties from disparate domains), and generality (i.e., handle a variety of architectures).

References

1 Fischer M, Balunovic M, Drachsler-Cohen D, Gehr T, Zhang C, Vechev M. Dl2: Training and querying neural networks with logic. InInternational Conference on Machine Learning 2019 May 24 (pp. 1931-1941). PMLR.
2 Kimmig A, Bach S, Broecheler M, Huang B, Getoor L. A short introduction to probabilistic soft logic. InProceedings of the NIPS workshop on probabilistic programming: foundations and applications 2012 (pp. 1-4).
3 Liu C, Arnon T, Lazarus C, Strong C, Barrett C, Kochenderfer MJ. Algorithms for verifying deep neural networks. Foundations and Trends® in Optimization. 2021 Feb 10;4(3-4):244-404.
4 Katz G, Huang DA, Ibeling D, Julian K, Lazarus C, Lim R, Shah P, Thakoor S, Wu H, Zeljić A, Dill DL. The marabou framework for verification and analysis of deep neural networks. InInternational Conference on Computer Aided Verification 2019 Jul 15 (pp. 443-452). Springer, Cham.
5 Singh G, Gehr T, Püschel M, Vechev M. An abstract domain for certifying neural networks. Proceedings of the ACM on Programming Languages. 2019 Jan 2;3(POPL):1-30.

6 Zhang H, Chen H, Xiao C, Gowal S, Stanforth R, Li B, Boning D, Hsieh CJ. Towards stable and efficient training of verifiably robust neural networks. arXiv preprint arXiv:1906.06316. 2019 Jun 14.

4.18 Machine Learning for Instance Selection in SMT Solving

Pascal Fontaine (University of Liège, BE)

Joint work of Daniel El Ouraoui, Cezary Kaliszyk, Jasmin Blanchette, Pascal Fontaine
Main reference Jasmin Christian Blanchette, Daniel El Ouraoui, Pascal Fontaine, Cezary Kaliszyk: "Machine Learning for Instance Selection in SMT Solving", in Proc. of the AITP 2019 – 4th Conference on Artificial Intelligence and Theorem Proving, 2019.
URL https://hal.archives-ouvertes.fr/hal-02381430

Satisfiability Modulo Theories (SMT) solvers are powerful tools used to check specifications of critical systems and to discharge proof obligations in proof assistants. For many such applications, quantifiers are necessary to express the problems. It often happens that SMT solvers fail finding proofs when too many quantifiers occur in the input problem. To deal with quantifiers, SMT solvers rely on instantiation, and use heuristic techniques to generate instances. Often, thousands of instances are generated and since most of them are useless, they impede the solver.

We use machine learning to predict the usefulness of an instance in order to decrease the number of instances generated and handled by the SMT solver. For this, we propose a meaningful way to characterize the state of an SMT solver, we collect instantiation learning data, and we integrate a predictor in the core of a state-of-the-art SMT solver. This ultimately leads to more efficient SMT solving for quantified problems.

4.19 CombOptNet: Fit the Right NP-Hard Problem by Learning Integer Programming Constraints

Anselm Paulus (MPI für Intelligente Systeme – Tübingen, DE) and Georg Martius (MPI für Intelligente Systeme – Tübingen, DE)

Joint work of Anselm Paulus, Michal Rolínek, Vit Musil, Brandon Amos, Georg Martius
Main reference Anselm Paulus, Michal Rolínek, Vit Musil, Brandon Amos, Georg Martius: "CombOptNet: Fit the Right NP-Hard Problem by Learning Integer Programming Constraints", in Proc. of the 38th International Conference on Machine Learning, Proceedings of Machine Learning Research, Vol. 139, pp. 8443–8453, PMLR, 2021.
URL https://proceedings.mlr.press/v139/paulus21a.html

Over recent years, deep learning has revolutionized multiple fields, such as computer vision, robotics, and natural language processing. This progress has predominantly built on the astonishing ability of neural networks to extract valuable information from raw data, an essential skill for approaching real-world problems. However, despite these successes, neural networks still notoriously struggle at algorithmic and logical reasoning tasks.

Such tasks can often be solved efficiently by combinatorial solvers, such as SAT or ILP solvers, which can build on a long development history. However, these solvers usually require a clean abstract formulation of the problem, such as boolean clauses or cost and constraint coefficients, instead of operating on raw data.

How can we bridge the gap between these two worlds? Ideally, we would like to use combinatorial solvers as building blocks of neural networks to build hybrid architectures that leverage both the feature extraction and the algorithmic reasoning capabilities of neural networks and combinatorial solvers, respectively. However, as combinatorial solvers typically operate on discrete structures, there is no continuously differentiable relationship between the inputs and outputs. In contrast, deep learning at its core relies on differentiability for end-to-end learning. Overcoming this fundamental conflict poses a significant challenge.

Recently proposed methods have addressed this challenge by considering continuously differentiable relaxations or by relying on informative gradient replacements that exploit the structure of the solver [1]. These methods have enabled the integration of dedicated combinatorial solvers into end-to-end trainable architectures, which extract the cost coefficients of the solver from raw data. While achieving promising results in computer vision and natural language processing applications, the strong prior information required to guide the choice of the problem-tailored combinatorial solver remains a limiting factor.

As an answer to this limitation, this talk introduces the recently developed method CombOptNet [2]. The goal of this method is to remove the restriction of a priori specifying a dedicated solver and to instead rely on a more general combinatorial building block. It is well known that many combinatorial problems can be formulated as ILPs, in which the constraint set determines the nature of the problem. Based on this observation, we aim to integrate a general ILP Solver into deep learning architectures. CombOptNet provides informative gradient replacements for both the cost and constraint coefficients. By learning the constraints from raw data, the architecture infers the nature of the combinatorial problem at hand. Thereby the architecture strives to achieve universal combinatorial expressivity.

References

1 M. Vlastelica, A. Paulus, V. Musil, G. Martius, and M. Rolínek. *Differentiation of BlackBox combinatorial solvers.* In International Conference on Learning Representations, ICLR, 2020.

2 A. Paulus, M. Rolinek, V. Musil, B. Amos, and G. Martius. *CombOptNet: Fit the Right NP-Hard Problem by Learning Integer Programming Constraints.* In International Conference on Machine Learning, ICML, 2021.

4.20 Machine Learning Algorithm Selection for Logic Solvers

Joseph Scott (University of Waterloo, CA)

Joint work of Joseph Scott, Vijay Ganesh, Aina Niemetz, Mathias Preiner, Saeed Nejati, Maysum Panju, Tony Pan

In this two-part talk, I present recent work in machine learning applied to logic and logic applied to machine learning.

First, I present some recent results on algorithm selection for logic solvers, specifically in the context of SMT solvers and neural network verification. As is typical for hard search problems, no single solver is expected to be the fastest on all inputs. This insight suggests using algorithm selection techniques that automatically select the fastest solver for a given input. We present MachSMT, an algorithm selection tool for SatisfiabilityModulo

Theories (SMT) solvers. MachSMT supports the entirety of the SMT-LIB. We provide an extensive empirical evaluation of MachSMT to demonstrate the efficiency and efficacy of MachSMT over three broad usage scenarios on theories and theory combinations of practical relevance (e.g., bit-vectors,(non-)linear integer and real arithmetic, arrays, and floating-point arithmetic). Additionally, we present Goose, an adaptive algorithm selection tool, which we dub a *meta-solver*, for deep neural network verification. We evaluate Goose by simulating VNN-COMP '21 and observe a 37% improvement over the competition winner.

Second, we introduce Logic Guided Machine Learning (LGML), a novel approach that symbiotically combines machine learning (ML) and logic solvers with the goal of learning mathematical functions from data. LGML consists of two phases, namely a learning-phase and a logic-phase with a corrective feedback loop, such that, the learning-phase learns symbolic expressions from input data, and the logic-phase cross verifies the consistency of the learned expression with known auxiliary truths. If inconsistent, the logic-phase feeds back "counterexamples" to the learning-phase. This process is repeated until the learned expression is consistent with auxiliary truth. Using LGML, we were able to learn expressions that correspond to the Pythagorean theorem and the sine function, with several orders of magnitude improvements in data efficiency compared to an approach based on an out-of-the-box multilayered perceptron (MLP).

References

1 Joseph Scott, Maysum Panju and Vijay Ganesh. LGML: Logic Guided Machine Learning (Student Abstract), AAAI 2020
2 Joseph Scott, Aina Niemetz, Mathias Preiner, Saeed Nejati and Vijay Ganesh. MachSMT: A Machine Learning-based Algorithm Selector for SMT Solvers. TACAS 2021.
3 Joseph Scott, Guanting Pan, Elias B. Khalil and Vijay Ganesh. Goose: A Meta-Solver for Deep Neural Network Verification. SMT workshop 2020.
4 Dhananjay Ashok, Joseph Scott, Sebastian Johann Wetzel, Maysum Panju and Vijay Ganesh. Logic Guided Genetic Algorithms (Student Abstract). AAAI 2021.
5 Lin Xu, Frank Hutter, Holger H. Hoos and Kevin Leyton-Brown. SATzilla: Portfolio-based Algorithm Selection for SAT. J. Artif. Intell. Res., 2008.

4.21 Two birds with one stone? Successes and lessons in building Neuro-Symbolic systems

Xujie Si (McGill University – Montréal, CA)

Joint work of Sever Topan, David Rolnick, Xujie Si
Main reference Sever Topan, David Rolnick, Xujie Si: "Techniques for Symbol Grounding with SATNet", in Proc. of the Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual, pp. 20733–20744, 2021.
URL https://proceedings.neurips.cc/paper/2021/hash/ad7ed5d47b9baceb12045a929e7e2f66-Abstract.html

Deep learning models have achieved remarkable successes in many challenging fields but suffer from a lack of interpretability, poor generalization ability, and difficulty in integrating human knowledge. Symbolic systems on the other hand address these limitations by design but heavily rely on hardcoded knowledge and have a very limited capability of learning. A promising design is perhaps building neuro-symbolic systems, combining the benefits of both worlds. However, such a design inevitably combines the challenges from both worlds and also faces some unique challenges in itself. In this talk, I will share some successes and lessons in building two neuro-symbolic systems – a data-driven optimization for symbolic software model checking and an end-to-end visual sudoku solver.

References

1 Sever Topan, David Rolnick, and Xujie Si. *Techniques for Symbol Grounding with SATNet*. Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual

2 Nham Le, Xujie Si, Arie Gurfinkel. *Data-driven Optimization of Inductive Generalization*. Formal Methods in Computer Aided Design, FMCAD 2021, New Haven, CT, USA, October 19-22, 2021

4.22 There is plenty of room at the bottom: verification and repair of small scale learning models

Armando Tacchella (University of Genova, IT)

Joint work of Armando Tacchella, Luca Pulina, Giorgio Metta, Lorenzo Natale, Shashank Pathank, Erika Abraham, Nils Jansen, Dario Guidotti, Francesco Leofante, Joost-Pieter Katoen, Simone Vuotto, Nina Narodytska, Andrea Pitto

Main reference Dario Guidotti, Francesco Leofante, Luca Pulina, Armando Tacchella: "Verification of Neural Networks: Enhancing Scalability Through Pruning", in Proc. of the ECAI 2020 – 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 – September 8, 2020 – Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020), Frontiers in Artificial Intelligence and Applications, Vol. 325, pp. 2505–2512, IOS Press, 2020.

URL https://doi.org/10.3233/FAIA200384

With the growing popularity of machine learning, the quest for verifying data-driven models is attracting more and more attention, and researchers in automated verification are struggling to meet the scalability and expressivity demands imposed by the size and the complexity of state-of-the-art machine learning architectures. However , there are applications where relatively small-scale learning models are enough to achieve industry-standard performances, yet the issue of checking whether those models are reliable remains challenging. Furthermore, in these domains, verification is just half of the game: providing automated ways to repair models that are found to be faulty is also an important task in practice. In this talk, I will touch upon some research directions that I have pursued in the past decade, commenting the results and providing some connections with related efforts. In particular we consider the following case studies:

- The problem of ensuring that a multi-agent robot control system is both safe and effective in the presence of learning components. In particular, we focus on a robot playing the air hockey game against a human opponent, where the robot has to learn how to minimize opponent's goals (defense play). This setup is paradigmatic since the robot must see, decide and move fastly, but, at the same time, it must learn and guarantee that the control system is safe throughout the process. We attack this problem using automata-theoretic formalisms and associated verification tools, showing experimentally that our approach can yield safety without heavily compromising effectiveness.
- Verification of Neural Networks known as Multi-Layer Perceptrons (MLPs), where we propose a solution to verify their safety using abstractions to Boolean combinations of linear arithmetic constraints. We show that our abstractions are consistent, i.e., whenever the abstract MLP is declared to be safe, the same holds for the concrete one. Spurious counterexamples, on the other hand, trigger refinements and can be leveraged to automate the correction of misbehaviors.

- Verification of Reinforcement Learning, a well-known AI paradigm whereby control policies of autonomous agents can be synthesized in an incremental fashion with little or no knowledge about the properties of the environment. We are concerned with safety of agents whose policies are learned by reinforcement, i.e., we wish to bound the risk that, once learning is over, an agent damages either the environment or itself. We propose a general-purpose automated methodology to verify, i.e., establish risk bounds, and repair policies, i.e., fix policies to comply with stated risk bounds. Our approach is based on probabilistic model checking algorithms and tools, which provide theoretical and practical means to verify risk bounds and repair policies. Considering a taxonomy of potential repair approaches tested on an artificially-generated parametric domain, we show that our methodology is also more effective than comparable ones.
- Verification of deep neural networks, particularly when it comes to enabel state-of-the-art verification tools to deal with neural networks of some practical interest. We propose a new training pipeline based on network pruning with the goal of striking a balance between maintaining accuracy and robustness, while also making the resulting networks amenable to formal analysis.

References

1 Stefano Demarchi, Dario Guidotti, Andrea Pitto, and Armando Tacchella. Formal verification of neural networks: A case study about adaptive cruise control. In Proceedings of the 36th ECMS International Conference on Modelling and Simulation, ECMS 2022, European Council for Modeling and Simulation, 2022.

2 Dario Guidotti, Francesco Leofante, Luca Pulina, and Armando Tacchella. Verification of neural networks: Enhancing scalability through pruning. In ECAI 2020 – 24th European Conference on Artificial Intelligence, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2505–2512. IOS Press, 2020.

3 Dario Guidotti, Luca Pulina, and Armando Tacchella. pynever: A framework for learning and verification of neural networks. In *Automated Technology for Verification and Analysis* – 19th International Symposium, ATVA 2021

4 Francesco Leofante, Nina Narodytska, Luca Pulina, and Armando Tacchella. Automated verification of neural networks: Advances, challenges and perspectives. *CoRR*, abs/1805.09938, 2018.

5 Francesco Leofante, Simone Vuotto, Erika Ábrahám, Armando Tacchella, and Nils Jansen. Combining static and runtime methods to achieve safe standing-up for humanoid robots. In Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques – 7th International Symposium, ISoLA 2016

6 Shashank Pathak, Erika Ábrahám, Nils Jansen, Armando Tacchella, and Joost-Pieter Katoen. A greedy approach for the efficient repair of stochastic models. In NASA Formal Methods – 7th International Symposium, 2015

7 Shashank Pathak, Luca Pulina, Giorgio Metta, and Armando Tacchella. Ensuring safety of policies learned by reinforcement: Reaching objects in the presence of obstacles with the icub. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems

8 Shashank Pathak, Luca Pulina, and Armando Tacchella. Verification and repair of control policies for safe reinforcement learning. *Appl. Intell.*, 48(4):886–908, 2018.

9 Luca Pulina and Armando Tacchella. An abstraction-refinement approach to verification of artificial neural networks.In Computer Aided Verification, 2010.

10 Luca Pulina and Armando Tacchella. Never: a tool for artificial neural networks verification. *Ann. Math. Artif. Intell.*, 62(3-4):403–425, 2011.

11 Luca Pulina and Armando Tacchella. Challenging SMT solvers to verify neural networks. *AI Commun.*, 25(2):117–135, 2012.

4.23 Synthesizing Pareto-Optimal Interpretations for Black-Box Models

Hazem Torfah (University of California – Berkeley, US)

Joint work of Hazem Torfah, Shetal Shah, Supratik Chakraborty, S Akshay, Sanjit A. Seshia
Main reference Hazem Torfah, Shetal Shah, Supratik Chakraborty, S. Akshay, Sanjit A. Seshia: "Synthesizing Pareto-Optimal Interpretations for Black-Box Models", in Proc. of the Formal Methods in Computer Aided Design, FMCAD 2021, New Haven, CT, USA, October 19-22, 2021, pp. 153–162, IEEE, 2021.
URL https://doi.org/10.34727/2021/isbn.978-3-85448-046-4_24

We present a new multi-objective optimization approach for synthesizing interpretations that "explain" the behavior of black-box machine learning models. Constructing human-understandable interpretations for black-box models often requires balancing conflicting objectives. A simple interpretation may be easier to understand for humans while being less precise in its predictions vis-a-vis a complex interpretation. Existing methods for synthesizing interpretations use a single objective function and are often optimized for a single class of interpretations. In contrast, we provide a more general and multi-objective synthesis framework that allows users to choose (1) the class of syntactic templates from which an interpretation should be synthesized, and (2) quantitative measures on both the correctness and explainability of an interpretation. For a given black-box, our approach yields a set of Pareto-optimal interpretations with respect to the correctness and explainability measures. We show that the underlying multi-objective optimization problem can be solved via a reduction to quantitative constraint solving, such as weighted maximum satisfiability. To demonstrate the benefits of our approach, we have applied it to synthesize interpretations for black-box neural-network classifiers. Our experiments show that there often exists a rich and varied set of choices for interpretations that are missed by existing approaches.

4.24 Data Usage across the Machine Learning Pipeline

Caterina Urban (INRIA – Paris, FR)

In this talk, I give an overview of past and ongoing work in developing abstract interpretation-based static analyses for reasoning about data and input usage across the machine learning development pipeline. I present work targeting data processing software (Python and Jupyter Notebooks) or trained machine learning models (neural networks but also decision tree ensembles and support vector machines), as well as model training itself.

References

1 Caterina Urban, Peter Müller. *An Abstract Interpretation Framework for Input Data Usage*. In Proceedings of the 27th European Symposium on Programming (ESOP 2018).
2 Caterina Urban, Maria Christakis, Valentin Wüstholz, Fuyuan Zhang. *Perfectly Parallel Fairness Certification of Neural Networks*. In Proceedings of the ACM on Programming Languages, Volume 4, Number OOPSLA, 2020.
3 Denis Mazzucato, Caterina Urban. *Reduced Products of Abstract Domains for Fairness Certification of Neural Networks*. In Proceedings of the 28th Static Analysis Symposium (SAS 2021).

4.25 On The Unreasonable Effectiveness of SAT Solvers: Logic + Machine Learning

Vijay Ganesh (University of Waterloo, CA)

Joint work of Vijay Ganesh, Jimmy Liang, Ed Zulkoski, Krzysztof Czarnecki, Pascal Poupart
Main reference Jia Hui Liang, Vijay Ganesh, Pascal Poupart, Krzysztof Czarnecki: "Learning Rate Based Branching Heuristic for SAT Solvers", in Proc. of the Theory and Applications of Satisfiability Testing – SAT 2016 – 19th International Conference, Bordeaux, France, July 5-8, 2016, Proceedings, Lecture Notes in Computer Science, Vol. 9710, pp. 123–140, Springer, 2016.
URL https://doi.org/10.1007/978-3-319-40970-2_9

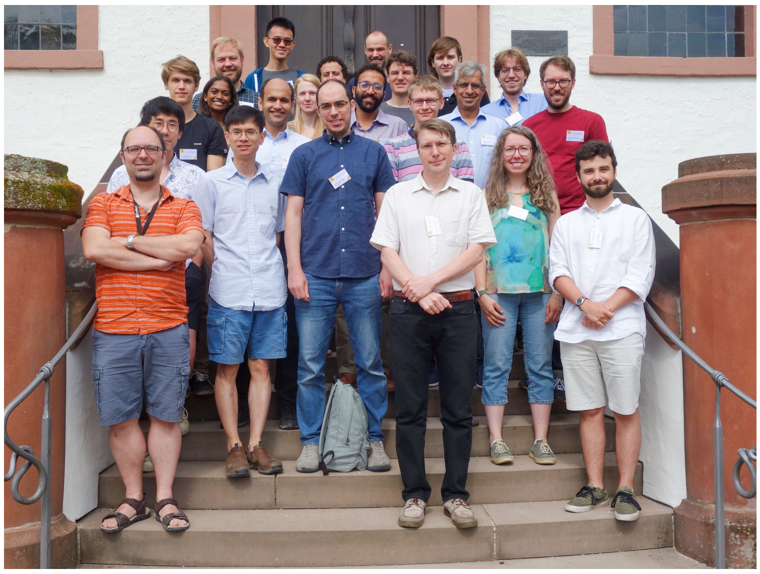
In this talk, we discuss a framework for viewing solver branching heuristics as optimization algorithms where the objective is to maximize the learning rate, defined as the propensity for variables to generate learnt clauses. By viewing online variable selection in SAT solvers as an optimization problem, we can leverage a wide variety of optimization algorithms, especially from machine learning, to design effective branching heuristics. In particular, we model the variable selection optimization problem as an online multi-armed bandit, a special-case of reinforcement learning, to learn branching variables such that the learning rate of the solver is maximized. We develop a branching heuristic that we call learning rate branching or LRB, based on a well-known multi-armed bandit algorithm called exponential recency weighted average.

5 Conclusion

In conclusion, this 5-day Dagstuhl seminar on topics at the intersection of machine learning and logic was very productive, enabling new collaborations and connections between researchers in the two camps of AI. We had over 25 talks that can be broadly categorized into the following four categories: 1) Neurosymbolic AI, 2) machine learning for solvers, 3) the testing, analysis, verification of machine learning systems, and 4) the use of machine learning in program synthesis. Key takeaways from the seminar included a greater need for intensification of collaboration between researchers in both camps, especially given the increasing importance of robust, secure, trustworthy, privacy-preserving, and interpretable AI. Most participants were very happy with the quality and diversity of the talks, the outcomes, new collaborations, and with our plan to continue organizing seminars in this series into the foreseeable future.

Participants

- Rajeev Alur
University of Pennsylvania – Philadelphia, US
- Sébastien Bardin
CEA LIST, FR
- Chih-Hong Cheng
Fraunhofer IKS – München, DE
- Jonathan Chung
University of Waterloo, CA
- Judith Clymo
University of Liverpool, GB
- Artur d'Avila Garcez
City – University of London, GB
- Alhussein Fawzi
Google DeepMind – London, GB
- Marc Fischer
ETH Zürich, CH
- Pascal Fontaine
University of Liège, BE
- Matt Fredrikson
Carnegie Mellon University – Pittsburgh, US
- Vijay Ganesh
University of Waterloo, CA
- Sebastian Junges
Radboud University Nijmegen, NL
- Chunxiao (Ian) Li
University of Waterloo, CA
- Ravi Mangal
Carnegie Mellon University – Pittsburgh, US
- Georg Martius
MPI für Intelligente Systeme – Tübingen, DE
- Kuldeep S. Meel
National University of Singapore, SG
- Grégoire Menguy
CEA LIST – Nano-INNOV, FR
- Matthew Mirman
ETH Zürich, CH
- Anselm Paulus
MPI für Intelligente Systeme – Tübingen, DE
- Markus N. Rabe
Google – Mountain View, US
- Joseph Scott
University of Waterloo, CA
- Xujie Si
McGill University – Montréal, CA
- Armando Tacchella
University of Genova, IT
- Hazem Torfah
University of California – Berkeley, US
- Caterina Urban
INRIA – Paris, FR
- Saranya Vijayakumar
Carnegie Mellon University – Pittsburgh, US



Remote Participants

- Somesh Jha
University of Wisconsin-Madison, US
- Luis C. Lamb
Federal University of Rio Grande do Sul, BR
- Vineel Nagisetty
Borealis AI – Toronto, CA
- Sanjit A. Seshia
University of California – Berkeley, US