Application of AI to formal methods — an analysis of current trends

Sebastian Stock, Jannik Dunkelau & Atif Mashkoor

Article - Version of Record

Suggested Citation:
Stock, S., Dunkelau, J., & Mashkoor, A. (2025). Application of AI to formal methods — an analysis of current trends. Empirical Software Engineering, 30(6), Article 175.
https://doi.org/10.1007/s10664-025-10729-8

Empirical Software Engineering (2025) 30:175
https://doi.org/10.1007/s10664-025-10729-8

Application of AI to formal methods — an analysis of current trends

Sebastian Stock[1] · **Jannik Dunkelau**[2] · **Atif Mashkoor**[1]

Received: 2 December 2024 / Accepted: 3 September 2025 / Published online: 14 October 2025

Abstract

Context With artificial intelligence (AI) being well established within the daily lives of research communities, we turn our gaze toward formal methods (FM). FM aim to provide sound and verifiable reasoning about problems in computer science.

Objective We conduct a systematic mapping study to overview the current landscape of research publications that apply AI to FM. We aim to identify how FM can benefit from AI techniques and highlight areas for further research. Our focus lies on the previous five years (2019–2023) of research.

Method Following the proposed guidelines for systematic mapping studies, we searched for relevant publications in four major databases, defined inclusion and exclusion criteria, and applied extensive snowballing to uncover potential additional sources.

Results This investigation results in 189 entries which we explored to find current trends and highlight research gaps. We find a strong focus on AI in the area of theorem proving while other subfields of FM are less represented.

Conclusions The mapping study provides a quantitative overview of the modern state of AI application in FM. The current trend of the field is yet to mature. Many primary studies focus on practical application, yet we identify a lack of theoretical groundwork, standard benchmarks, or case studies. Further, we identify issues regarding shared training data sets and standard benchmarks.

Keywords Formal methods · Artificial intelligence · Machine learning · Systematic mapping study

1 Introduction

Artificial intelligence (AI), and especially the subdomain of machine learning (ML), is becoming increasingly relevant for all industries and increasingly penetrates research communities, as several studies show (Lu 2019; Elahi et al. 2023; Jiang et al. 2023; Chang 2023).

Communicated by: Carlo A. Furia.

Extended author information available on the last page of the article

Consequently, AI applications also gained popularity in software development (Barenkamp et al. 2020; Shafiq et al. 2021).

This study asks whether this trend is also observable within the formal methods communities. Formal methods (FM) correspond to a mathematically rigorous approach to software and systems development in which the concern is to provide a formal representation of hardware, software, or systems engineering problem (Abran et al. 2004). This representation then lends itself to mathematical assessments such as correctness, soundness, or well-definedness. These assessments can be easily reproduced and followed by the practitioner.

However, the needed guarantees and the rigorousness of these methods seem to conflict with AI applications, which often show nondeterministic behavior and mostly correspond to black-box techniques. The trade-off appears obvious: Injecting AI's nondeterminism and unpredictability into FM's rigorousness could lead to faster results and proofs with less human intervention at the price of fundamental guarantees. This point of tension seems interesting and needs further investigation, but it is not the only way how AI can enrich FM. Much like in software development, AI-driven tools might still supplement the formal development process without impeding the correctness of the results. From personal experience, we noticed increased publications applying AI techniques to FM in recent years, suggesting that the FM field also follows the overall trend.

To substantiate this and to obtain an overview of which FM subdomains already utilize AI, we performed this systematic mapping study (SMS) (Petersen et al. 2015). The primary goal is to observe the development of AI applications quantitatively to the field of FM.

In this work, we report on our journey and the results of our SMS, accessing the quantitative nature of the topic of AI applications in the FM domain. This represents a typical goal for this kind of study, as pointed out by Kitchenham et al. (2010). Mainly, we are interested in which FM domains are targeted by AI improvement already, which AI or ML applications have been investigated in the literature so far, where potential research gaps exist, and whether there appears to be consensus within the FM domains of which AI techniques seem most beneficial. A challenge in this assessment is the wide variety of topics on both sides. AI and FM are huge topics, and their potential overlap may be considerable. This study aims to be the first assessment of the general research landscape, i.e., we assume a mainly quantitative lens on the topic. Consequently, we highlight areas that received more attention in the literature, as well as those that received less attention. However, we refrain from doing a qualitative assessment, e.g., a systematic review of the literature. Our goal is to gain an initial overview of the field's landscape.

While our search process resulted in a data set of 457 highly relevant publications covering more than 50 years of research applying AI to FM, we investigated only articles from the last five years in more detail. This period is chosen as it corresponds to a peak in the number of publications as indicated by previously mentioned studies and also allows us to focus on the most recent developments. Consequently, we have 191 primary studies, two of which are pure data set presentations. We analyzed the remaining 189 studies to determine which AI techniques were found to be applicable in which FM areas. Further, we derive research suggestions from our insights by pointing out research gaps that might further mature the field. Both the data set of the last five years and the complete data set of 457 publications are made publicly available to interested researchers at https://github.com/hhu-stups/ai4fm-studies.

The rest of this mapping study is structured as follows: Section 2 introduces the relevant subdomains of FM and AI. It will also provide an overview of our understanding of AI. We present our leading research questions in Section 3 and detail our search and selection process in Section 4. Section 5 presents the results of the search while Section 6 engages in a deeper discussion about these results. In Section 7, we discuss potential threats to the validity of our conducted systematic mapping study. Section 8 presents related studies we found during our research and compares them against our efforts. Finally, we conclude in Section 9.

2 Background

2.1 Systematic Mapping Studies

Systematic mapping studies (SMS), as proposed by Kitchenham et al. (2007), aim to assess the quantitative nature of a research field. Here, the goal is to give an overview of the available research. The counterpart to an SMS would be a systematic literature review where the main goal is to assess the quality of contributions.

SMS are typically performed using a multi-step approach. First, the field of interest is selected. Then, research questions are formulated, and inclusion criteria (IC) and exclusion criteria (EC) are defined. IC and EC serve as filters over the collected primary studies to help select relevant results.

After that, the search query is crafted. The goal is to apply the search query to the scientific meta-search engines to retrieve a good corpus of primary studies that answer the research questions. With the retrieved corpus, the IC and EC are applied. If indicated, snowballing is done, usually until a stable closure of the corpus is reached. With a corpus constructed, the aim is to answer the research questions.

2.2 Formal Methods

Formal methods (FM) describe a mathematically rigorous approach to design and assess software as well as hardware systems, and are concerned with analysis, validation, and verification at any part of the respective system's life cycle (Woodcock et al. 2009; Clarke and Wing 1996). Employing FM allows to formulate precise statements of desired functionality or requirements in form of a *formal model* (or *formal specification*) while not constraining the possible implementation thereof. The formal model is represented in a mathematically approachable form which lends itself to reasoning and hence allows rigorous analysis of critical properties such as correctness, safety, soundness, or well-definedness. The following section overviews FM's creation and this work's most relevant reasoning methods.

Model Checking Model checking (Baier and Katoen 2008; Clarke et al. 2018) explores a program's or system's state space. The state space sets all possible value constellations that the program can achieve. In its most basic form, model checking aims to explore all reachable states and check if they are faulty, i.e., violate any specified properties. If this is the case, a counterexample is found. Otherwise, the system works correctly, as no faulty states are reachable.

There are different types of model checking, besides classical, explicit state model checking. Noteworthy for this study are linear temporal logic (LTL) model checking, symbolic model checking, and statistical model checking. While explicitly visiting states, LTL model checking focuses on temporal properties encoded in LTL formulas, which are checked against execution traces of the state space. Symbolic model checking abstracts the state space and makes a symbolic evaluation of a complex expression; thus, it shrinks the overall state space for the price of precision. With enough abstraction, it can handle even infinite state spaces for the price of precision. Statistical model checking investigates state spaces created by assigning probabilities on transitions between states and provide probabilistic guarantees.

Theorem Proving A core application of theorem proving (TP) is to evaluate statements about a formal model's internal consistency and behavioral integrity. When talking about theorem proving, we usually assume some formal representation of the problem for which proofs are formulated and discharged. Discharging proofs can either be done by a fully automatic theorem prover (ATP) (Gallier 2015) that finds a solution on its own, or an interactive theorem prover (ITP) (Bertot and Castéran 2013) that requires human input for individual proof steps and transformations. ITPs provide a user interface to keep track of progress, sub-goals, and available properties.

SAT/SMT Solving The Boolean satisfiability problem (SAT) (Biere et al. 2015) refers to finding the right model for a Boolean formula to satisfy the formula. Historically, SAT was the first problem found to be NP-complete and has since drawn a broad research interest.

Satisfiability modulo theories (SMT) (Barrett and Tinelli 2018) is a more generalized form of the SAT, which enriches the problem setting with various theories such as arithmetic, data structures, or set theory.

In the context of FM, SAT and SMT solving find applications in theorem proving (Brown 2013), bounded model checking (Shtrichman 2000), equivalence checking (Goldberg et al. 2001) or test generation (Zeng et al. 2005).

Synthesis Under the term of synthesis, we group all attempts automatically or semi-automatically create (parts of) formal models during the formal development process, such as the generation of models from natural language specification or vice versa. We further group the idea of automated model repair under this aspect, where, for a model with some violation, a fix is synthesized that rectifies the violation.

Other Categories In the scope of this mapping study, we may encounter FM approaches and techniques that do not belong to any previous group. We will group these as *other* FM techniques. For instance, this includes meta-approaches such as selecting the right verification algorithm, general program analysis, or termination analysis.

2.3 Artificial Intelligence

Artificial Intelligence (AI) is a catch-all phrase for various techniques and approaches that aim to imitate seemingly *intelligent* behavior. Simmons and Chappell (1988) define the term

as "[...] behavior of a machine which, if a human behaves in the same way, is considered intelligent". They further emphasize the problem with the term itself, pointing out the difficulty of precisely defining what it means for a human to be intelligent. Nevertheless, AI is regularly used in daily life and the research community.

An important AI paradigm is machine learning (ML). The goal of ML algorithms is to observe and extract patterns within the data which can then be applied to the automation of a given task (Mitchell 1997). Instead of manually defining a set of rules for pattern recognition, machine learning algorithms uncover such rules automatically, a process referred to as *learning*. ML algorithms experience the given data in some manner and improve their exhibited performance to solve the task at hand gradually, the more experience they gather.

Below, we list and briefly explain AI and ML algorithms which are most relevant to this mapping study. For the reader familiar with AI, the selection seems mostly ML rather than AI. Indeed, we aimed to find applications of AI to FM, and the search query we will discuss later was prepared accordingly. However, to bring a result up front, most contributions use some form of ML.

Neural Networks and Deep Learning Neural networks (NNs) (Rosenblatt 1962) build the foundation of what is known today as the area of deep learning (Goodfellow et al. 2016). A deep neural network (DNN) consists of a layer of inputs, a layer of outputs, and one or more hidden layers. Inputs are forwarded layer-wise and combined first by a weighted sum and a follow-up non-linear transformation. The clue is the training of a DNN, which starts the network with randomly initialized weights for the summations above, but adjusts them gradually in the learning process to converge to the desired function.

A considerable benefit of neural networks lies in their variety. Different approaches and architectures exist for deep learning in specialized environments with differently shaped data. The most important for this work are convolutional neural networks (CNNs) (LeCun et al. 2010) for images, recurrent neural networks (RNNs) (Jordan 1997) for sequential data, the transformer architecture (Vaswani et al. 2017) for language processing, and graph neural networks (GNNs) (Zhou et al. 2020). Today, deep learning is considered one of the most popular topics in ML (Sarker 2021).

Reinforcement Learning Reinforcement learning (RL) (Kaelbling et al. 1996) is foremost a machine learning paradigm. Given an environment, a set of actions that change the environment, and a reward function that quantifies a given environmental state, the RL agent aims to learn a policy over their available actions that maximizes the cumulative reward over time. Initially, the agent does not know the environment or the task it is supposed to solve, but can only learn from feedback through the received rewards (Sutton and Barto 2018).

Natural Language Processing Natural language processing (NLP) (Chowdhary 2020) describes the research area of, as the name suggests, processing and evaluating texts of human language. This includes information extraction, language translation tasks, text classification, semantic analysis, and natural language generation and dialogue systems (Chowdhary 2020; Khurana et al. 2023).

One of the most recent developments is large language models (LLMs) (Chang et al. 2024) which gained significant popularity and even worldwide attention with the release of ChatGPT (Wu et al. 2023). For the sake of compactness, we count contributions with LLMs towards NLP.

Genetic/evolutionary Algorithms Genetic and evolutionary algorithms (EAs) (Mitchell 1998) describe a family of optimization algorithms rooted in evolution. An EA starts with a randomly assembled population of candidates for a given optimization problem and a fitness function. The fitness function measures how well a candidate solves the problem. Parent candidates are chosen via a random selection process that is nonetheless influenced by each individual's fitness. These are adapted using mutation or recombination to produce a new child population. Repeating this process converges the individuals toward suitable solutions for the problem at hand.

Statistical Approaches In contrast to the abovementioned techniques, we consider techniques from more classical, statistical ML. These differ from the above methods as they do not depend on deep learning and are typically faster to compute in comparison. This does, however, not imply that they are not competitive. The considered statistical approaches include support vector machines (SVM) (Kecman 2005), logistic and linear regression (LR) (Montgomery et al. 2021), k nearest neighbors (KNN) (Cover and Hart 1967), decision trees (DT) (Breiman et al. 1984) and random forests (RF) (Breiman 2001), or Bayesian inference approaches (Tipping 2004).

2.4 Contribution Classification

We classified the contributions into various types. The aim is to show the different approaches to target the same research area. The different types of contributions are distinguished as follows:

Tool Contributions that provide practical means to solve an FM problem, for instance, in the form of a binary or source code release.

Tool Enhancement Contributions that aim to enhance or complement an existing tool.

Approach Contributions that propose an overall style or idea to overcome a problem. An approach is a generalized concept that details overcoming and resolving a problem. The approach remains more theoretical and does not involve tested or empirically proven steps.

Methodology Contributions that realize and test an approach.

Framework Contributions that propose a conceptual structure intended to support or guide the construction or expansion of a solution for an actual problem.

Case Study Contributions apply an approach, a methodology, or a framework to a real-world problem.

Benchmark Contributions that test an existing tool or methodology.

Idea Contributions that focus on a brief overview or discussion of an idea to solve a problem. Compared to an approach, it does not lay out larger theories but serves as a quick pitch.

Training Data An openly accessible data set that claims to be reusable outside of the replication of the contribution it was originally used in.

2.5 Publication Venue Classification

We further divide the publication venues into the following categories to better assess the respective maturity of the contributions:

Workshop and Short Papers We grouped contributions in this category if they are (a) rather short (4 pages or less) or (b) if the venue they appeared in has workshop character, i.e., is classified as a workshop, calls itself a workshop, or focuses on the in-person presentation of results but requires submission of an extended abstract.

Conference Proceedings Here, we group all the contributions that are published in the proceedings of a conference. From a full conference paper, we expect a more in-depth explanation of the result than compared to a workshop or short paper, ultimately resulting in an overall more sizable contribution.

Journal Articles Here, we group all contributions published as part of a journal volume. As journal articles tend to consist of more pages and go through a more thorough reviewing process, the results of journal articles are expected to have a broader scientific basis for their claims.

Book Chapters As the name suggests, these contributions are part of a larger collection of articles bundled in one book. As the production of scientific books takes a significant amount of time, the reader expects a high quality of the provided scientific contribution that is considered, by that time, state of the art.

3 Research Questions

To get an overview of the field, we divide our research questions (RQs) into two main areas. First, RQs 1.1 to 1.3 focus on the overall demographics of the research field. With these questions, we want a basic quantitative overview to explore the field's maturity. Second, RQs 2.1 to 2.4 focus on the content of the contributions and aim to map out the field, highlighting potential gaps and showcasing quantitative interest in respective subtopics. Our research questions are defined as follows.

RQ 1: Demographics of the research area

1.1 What is the research publication timeline, and is there a trend? *Rationale:* We are interested in knowing whether the field experiences a growth or decline in interest, or if no trends are detectable at all.
1.2 Which publication venues are most frequent in the field? *Rationale:* We want to explore the field's maturity, whereby books, book chapters, and journal articles indicate more mature results.
1.3 What are the main contributions provided by the primary studies? *Rationale:* We want to learn where the community focuses its efforts, e.g., more on the theoretical groundwork or practical application.

RQ 2: Content type of contributions

2.1 Which AI techniques and tools were used? Are there any prevalent choices within the community? *Rationale:* We want to know if there is an indication that some AI techniques are more relevant (or at least popular) than others.
2.2 What are the application areas of AI in FM? Are there any commonly found FM techniques or tools? *Rationale:* We are curious about which FM techniques currently get the most attention or if there are dark spots in the literature.
2.3 What is the distribution of AI types in the different FM application areas? *Rationale:* Here we want to overlap the results of RQs 2.1 and 2.2..
2.4 Are the studies' employed data sets publicly available? *Rationale:* Especially in machine learning, having the source material available and accessible for experiments benefits the scientific community.

4 Search Strategy

In the following, we explain our approach for aggregating the contributions for this mapping study which we have visualized in Fig. 1. For this, we follow the suggested multistep approach by Petersen et al. (2015), which consists of an initial search in meta-search engines, followed by a snowballing process. As our selection of topics will show, it provided a challenging environment with overlapping search terms and related research areas. We tweaked our search strategy to address these challenges and avoid a result set of tens of thousands of potential studies.

A vital role was played by the IC & EC, which we applied twice. After the initial search, the first application was made to remove unrelated contributions that would introduce overheating into the snowballing. The second time was again after the snowballing to filter out wrongly selected contributions.

4.1 Employed Search Query

Finding a search query that would return a manageable amount of relevant contributions proved difficult. The main problem was harnessing the bandwidth of terms used in the AI and FM domains. Hereby, the difficulty lies in the often ambiguous use of terms like "formal

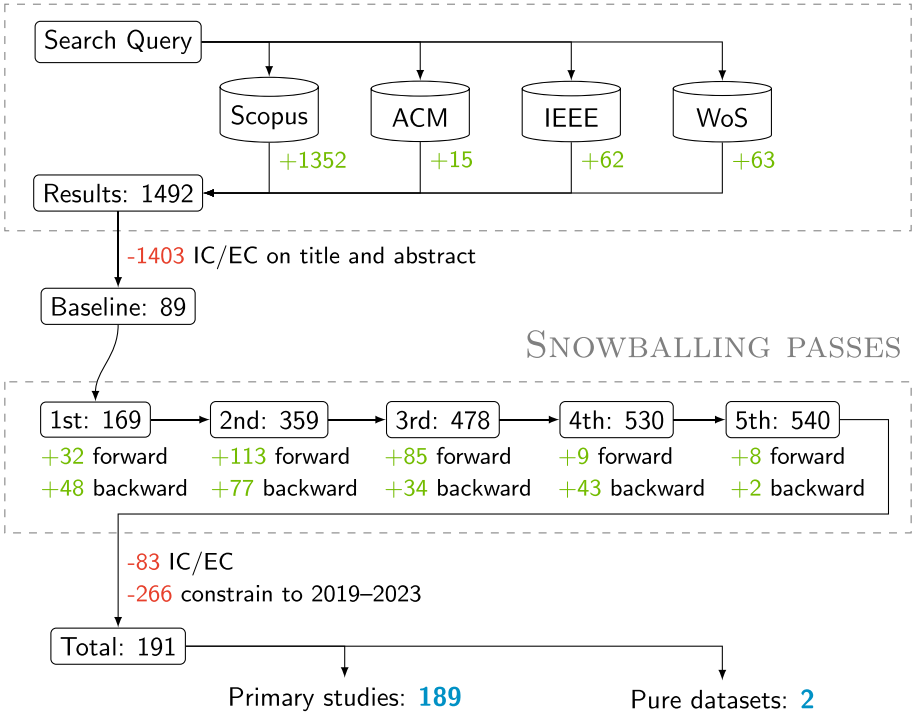


Fig. 1 Visualization of the search process

method", which can be used in the sense of formal methods as outlined in Section 2.2 or as a term to describe an only somewhat formal approach to a problem. The situation is even more difficult for AI, as AI and ML are often used interchangeably. In contrast, essential publications may only use the specific name of an applied technique without naming the fields of AI or ML.

Another issue was the cross-pollution of our results with studies from the related research field of verification for AI and ML systems. This is essentially the other direction in which FM is applied to AI. This field is also a highly relevant topic and shares the majority of potential search key phrases, further inflating our results tally.

As a reaction, we experimented with different approaches and probed the quantity and quality of their respective results to find a suitable search query. Highly abstract queries such as

"formal methods & artificial intelligence"

produce multiple thousands of results which we deemed impossible to process in a reasonable amount of time and which contained many studies outside of our intended scope. From probing some results as well as personal experience, we also knew that many relevant publications tend not to use these high-level terms, but use nomenclature specific to their sub-community. For instance, research regarding the formal B method tends to simply use "B method" instead of "formal method" as keyword. On the other hand, we noticed how

relevant publications also seemed to prefer to use the concrete names of the applied AI algorithms, rather than stating they used AI in a superficial manner. Therefore, we were concerned that a search query that was too abstract would not be able to penetrate the relevant fields well enough.

As a result, we settled on a more complex query containing the precise terminology of specific algorithms. For this, we distinguish between two sets of terms, the AI-terms (Fig. 2b) and the FM-terms (Fig. 2a). Both correspond to a disjunction of a set of selected

FM-terms := "formal method" OR "formal model" OR "specification" OR "rigorous" OR "prove*" OR "solv*" OR "automated reason*" OR "formal synth*" OR "model check*" OR "model repair" OR "premise selec*" OR "SAT" OR "SMT" OR "smtlib"

(a) FM keywords used in the query

AI-terms := "artificial intelligence" OR "AI" OR "machine learn*" OR "supervised learn*" OR "unsupervised learn*" OR "classification" OR "regression" OR "portfolio" OR "deep learn*" OR "neural net*" OR "bayesian net*" OR "reinforcement learn*" OR "reinforcement agent" OR "multi-armed bandit" OR "knn" OR "k nearest neighbours" OR "support vector" OR svm OR "decision tree" OR "random forest" OR "gradient boost*" OR "xgboost" OR "logistic regress*" OR "linear regress*" OR "cluster*" OR "computer vision" OR "nlp" OR "natural language proces*" OR "genetic prog*" OR "genetic algorithm" OR "expert system" OR "inductive logic progr*"

(b) AI keywords used in the query

TITLE(AI-terms) AND TITLE(FM-terms) AND
ABSTRACT(AI-terms) AND ABSTRACT(FM-terms) AND
KEYWORDS(AI-terms) AND KEYWORDS(FM-terms)

(c) Unified search query

Fig. 2 Construction of the used search query. The query consists of a disjunction of search terms for AI and FM, respectively. The unified query (c) enforces that at least one keyword from each subfield is present in the title, the abstract, and the keywords of a respective publication. An asterisk (*) indicates a wildcard character that can match any sub-word. For instance, "solv*" matches with *solver* but also with *solving*

terms from the respective field. The terms themselves were selected based on previous experiences in the field following the PICO (**P**opulation **I**ntervention **C**omparision **O**utcome) approach as suggested by Kitchenham et al. (2007).

Here **P**opulation may refer to a specific field of FM or AI, such as formal model(ing) or deep learn(ing). **I**ntervention may refer to specific techniques or methodologies of FM or AI, such as SAT or SVM. **C**omparision was not applicable as we focused on a direct connection, while **O**utcomes shall contain at least strongly correlated AI and FM terms.

The final search query (Fig. 2c) now uses both term aggregates to produce a corpus of primary studies in a strict and targeted manner. We only looked for studies that use at least one AI term and at least one FM term in their titles, abstracts, and keywords. The reasoning is to get a good initial penetration of all key fields, while relevant but missed entries are found in the later search stages via snowballing.

4.2 Database Search and Processing

The search was conducted in the last quarter of 2023. We used four meta-search engines[1] as suggested by Petersen et al. (2015): IEEE[2], Scopus [3], ACM[4] and Web of Science (WoS)[5]. We used the *Guide to Computing Literature* for ACM. Furthermore, as the ACM search engine limits the number of wildcards, we split the search terms into subsets, performed the subsearches, and merged them into the required superset. For WoS, results seemed to differ depending on which institution had access. Therefore, we decided to take the more extensive result set. As far as the search engines allowed, we applied the EC, such as the area of publishing and the publication language. The result size after this step was 1492 entries. This number and all later numbers are without duplicates.

4.3 Inclusion and Exclusion Criteria

For the 1492 resulting contributions, we made two observations. First, the amount was too large for a snowballing procedure to be feasible. Second, while briefly looking at the corpus, we discovered many contributions that should not have been selected, i.e., contributions that included the term "specification" but meant it in a purely requirement engineering-minded context. Therefore, we decided to apply our IC and EC prematurely to the corpus as far as they applied, i.e., to the title and abstract. For this, we read the titles of our studies. If we could not decide whether the contributions should be included, we also conducted an abstract review, and very rarely, the contributions themselves were skimmed.

To achieve this for such a large corpus, the first and second authors passed over all entries individually and decided whether they should be included or excluded. In agreement cases, the respective studies were kept or discarded. For disagreement, both authors discussed each instance together to reach an agreement. The disagreement could not be settled this way in

[1] Springer's engine was not used because by the time of the search it performed extremely poor in dealing with large search queries, large result sets and filtering.

[2] https://ieeexplore.ieee.org/Xplore/home.jsp

[3] https://www.scopus.com/search/form.uri?display=basic#basic

[4] https://dl.acm.org/

[5] https://www.webofknowledge.com/

two cases, and the third author was consulted as a tiebreaker. This procedure reduced the number of primary studies to 89, but took time.

The applied IC and EC are detailed in the following.

IC 1: The contribution focuses on applying techniques from the domain of AI to the domain of FM.
IC 2: The contribution is in English.
IC 3: The contribution has at least two pages of content (excluding references).
IC 4: If the contribution was submitted in different versions, we took longer. For instance, the journal version was taken for a conference contribution and subsequently published as an extended version. The journal version was taken for a journal-first submission and an invited version.
IC 5: The contribution is a data set used for ML.

EC 1: The contribution appears not to have been reviewed in any form.
EC 2: The contribution is not available.
EC 3: The contribution is not posed in computer science.
EC 4: The contribution is unclear (even after reviewing the full text).
EC 5: The contribution solely focuses on solving a mathematical optimization problem.
EC 6: (During initial skim) The contribution focuses only on common or general constraint-solving techniques (without specializing in SMT/SAT)
EC 7: (During initial skim) The contribution focuses on portfolios for SAT solvers without explicitly mentioning any AI/ML technology
EC 8: (During initial skim) The contribution is not a primary study, i.e., it is another mapping study, survey, etc.
EC 9: (Snowballing) The contribution is concerned with 2Sat, which is solvable in polynomial time.
EC 10: (Snowballing) If the contribution proposes a way to synthesize some automaton (e.g., a Markov process), then the produced result must explicitly be used within the context of FM. This means that the synthesized automaton and its analysis with FM's help are described within the contribution.

For the EC, some additional criteria were added in later steps in response to contributions encountered that were of low value for this study. For instance, EC 6 was added because many contributions focused on constraint solving, thus being closer to mathematics. EC 7 was added in response to a large influx of SAT contributions that mentioned *learning* but focused on remembering already-seen clauses. EC 8 is a rule that was introduced to not have secondary studies in the result set but to use those studies to ensure that even with a restrictive query, we cover as much of the field as possible. EC 10 was added in the fifth snowballing round, where we got many contributions that synthesized automatons, but no further analysis was mentioned. While an automaton may represent a formal model, analysis of larger automatons is very human-unfriendly compared to mathematical formulas or formal models written in a modeling language. Therefore, we expected the authors to subject the generated automaton to some validation, e.g., model checking or SAT/SMT solv-

ing, to enable reasoning about the validity of the generated automaton. EC 9 was added as we found 2Sat as a problem irrelevant for this study, as solutions are already achievable in polynomial time.

4.4 Snowballing

For the 89 entries, we conducted an extensive snowballing procedure (Wohlin 2014) where we conducted both forward and backward snowballing. Five iterations were needed until we eventually reached closure. This extensive snowballing approach was meant to complement the strict search query and to uncover relevant yet initially missed studies or even subfields of research. Therefore, snowballing served as a means to alleviate these threats to validity.

For the backward snowballing, we consulted the references of a given contribution and selected promising titles or publications that were explicitly highlighted in the respective related work sections. If a publication appeared promising, but the contribution was unclear from its title alone, the abstract was also consulted. For the forward snowballing, we used Google Scholar and applied the same title and abstract procedure. This brought the number of contributions up to 540. After concluding the snowballing, we applied the IC and EC a second time, reducing the number of contributions to 457.

4.5 Filtering by Years

While the 457 results are all highly relevant, the number of contributions was too large to conduct a deeper investigation necessary to answer all the RQs that require a more in-depth consideration of the contributions, i.e., RQs 1.2 and 1.3 and RQ 2. Therefore, we restricted the result tally to studies published between 2019 and 2023. While this shifts our focus on the most recent developments in the field, it aligns with the peak in this period we discussed back in Section 1. By only considering studies from 2019 to 2023, we reduced the corpus to 191 primary studies. Of these, 2 studies purely provide data sets for future AI applications in the field of FM without showcasing any form of direct AI application themselves (IC 5). That leaves us with a final tally of 189 primary studies to conduct this mapping study.

5 Results

In the following section, we will discuss the results. For this, we will answer the individual research questions and aim to make a cross-analysis between individual research questions, thus extracting as much information as possible from the existing dataset. We refer to Section 6 for a discussion of the observed results. An overview of the results matched to their respective application domain in FM can be found in Table 1.

5.1 RQ 1: Demographics of the Research Area

5.1.1 RQ 1.1: What is the Research Publication Timeline, and is there a Trend?

The whole corpus of 457 primary studies we found was published in the years 1972–2023. Figure 3 displays the histogram of publications per year and highlights that AI applications

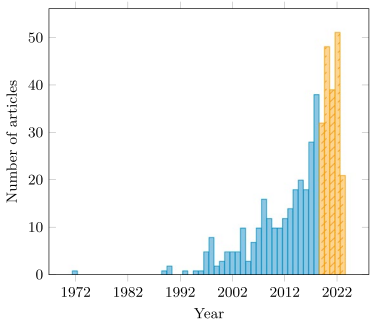


Fig. 3 Distribution of contributions over years

on FM appear to mirror the trend of general AI applications mentioned in Section 1. That is, we can observe a steadily growing interest in the field, especially since around 2005.

Focusing on the last five years only (recall Section 4.5), there still appears to be growing interest overall, while not each year surpasses the respective previous one. We have 32 contributions for 2019, 48 for 2020, 39 for 2021, 51 for 2022, and 21 for 2023. Noteworthy are two observations: First, for the last five years alone, there has been no strict upward trend in the number of publications, We can observe a decline from 2020 to 2021. Second, there were few publications in 2023. We explain this because we searched in the last quarter of 2023, when not all possible studies were published. Given that we do not yet have complete data for 2023, it is impossible to forecast whether the overall trend will continue to grow or if the current peak from 2022 marks a global maximum.

5.1.2 RQ 1.2: Which Publication Venues are Most Frequent in the Field?

Figure 4 displays the publications over the years, divided by publication venues. The majority of studies were published as conference papers while journal papers seem to be on the decline again since their peak in 2018. Within our five-year observation range, 124 contributions were conference papers, followed by 37 journal articles 29 workshop or short papers, and only one (1) book chapter with no full books.

Figure 5 shows the publication venues sorted by the main targeted area. For this, we skimmed through the titles of the conferences and sorted them into six different areas. We can see the majority of publications (94) happened in AI-focused venues, 49 contributions were published in software engineering venues, 40 in FM venues, 19 in venues concerned with automated reasoning specifically, 8 in mathematics-focused venues, and 20 in other,

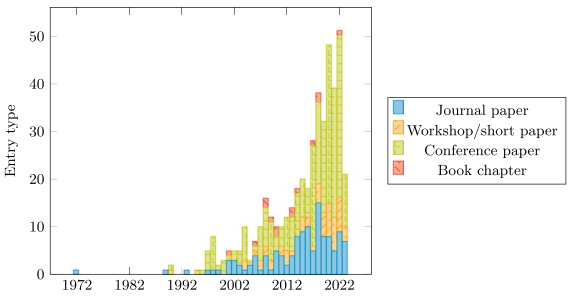


Fig. 4 Distribution of publication venues by submission type

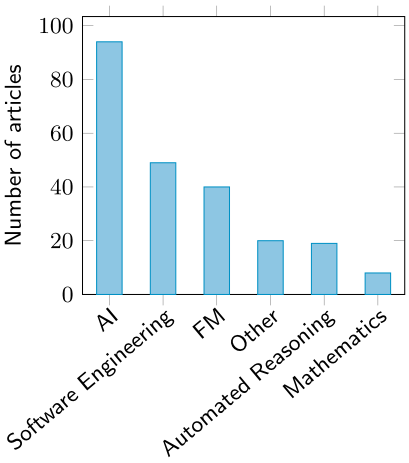
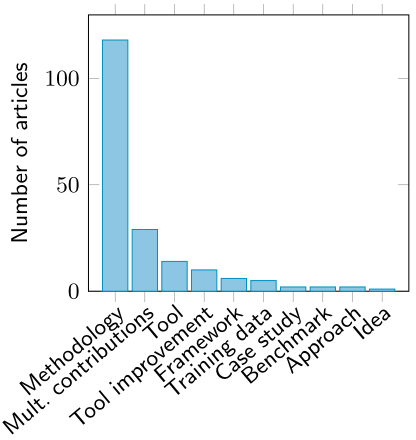


Fig. 5 Distribution of publication venues by area

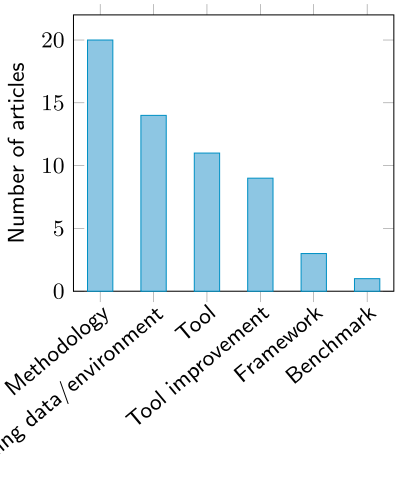
less topical venues. Note that some venues have more than one area of focus, e.g., we counted publications from the conference on artificial intelligence and theorem proving (AITP) for both AI and FM.

5.1.3 RQ 1.3: What are the Main Contributions Provided by the Primary Studies?

Following the definitions of contribution types from Section 2.4, we see the division of the studies into these contribution types in Fig. 6a. We can see that the majority of contributions were methodologies (118), i.e., practical application of AI to FM, followed by studies with multiple types of contributions (29). Third place was tool contributions (14), followed by improvements for tools (10), followed by frameworks (6), training data (5), case studies (2), benchmarks (2), approaches (2), and finally, one (1) idea contribution. Studies with multiple



(a) Singular contribution types



(b) Division of studies' multiple contributions

Fig. 6 Distribution of contribution types of collected primary studies.

contribution types most commonly provide a methodology (20/29), training data (14/29), or a tool (11/29). See Fig. 6b for a complete breakdown.

5.1.4 Intersecting RQs 1.2 and 1.3

We can gain some additional insights by investigating the individual results in context with each other. Figure 7 shows how the publication venues and the contribution types correlate. We can see that for any contribution type, a heavy focus lies on conference papers. Interestingly, we can see that case studies are only conducted within the scope of journal papers and book chapters. Furthermore, workshops and short papers are primarily used to introduce new methodologies, which seem unexpected, as usually, due to limited space, authors aim to restrict themselves to outlining an approach or an idea.

5.2 RQ 2: Content type of Contributions

5.2.1 RQ 2.1: Which AI Techniques and Tools Were Used?

In Fig. 8a, we can see that the majority of contributions use NN (70), followed by RL (32). 27 contributions use multiple techniques, while 16 use NLP methods and 14 use EA. A total of 8 studies presented custom algorithms. The rest of the contributions are divided between utilizing decision trees (7), clustering (3), Bayesian inference (3), KNN (2), random forests (2), data mining approaches (2), automaton learning (2), and SVM (1).

Studies that employed multiple algorithms mostly did so in a contrasting manner, i.e., they trained on numerous models to see which one performed best for their respective applications. A detailed view of the category of multiple algorithms is given in Fig. 8b. Here, we can see that if multiple techniques are used, they mainly rely on NN (13/31), random forest (12/31), and some variant of tree learners (12/31). Interestingly, SVM (6/31) and KNN (5/31) applications are more predominant in a setting with multiple algorithms compared to being the sole focus of a study. We explain this by KNN or SVM, which are simple algo-

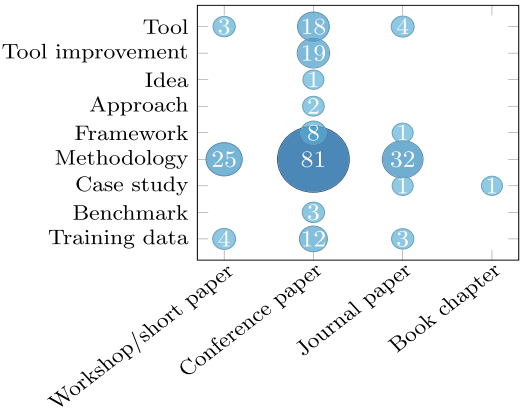
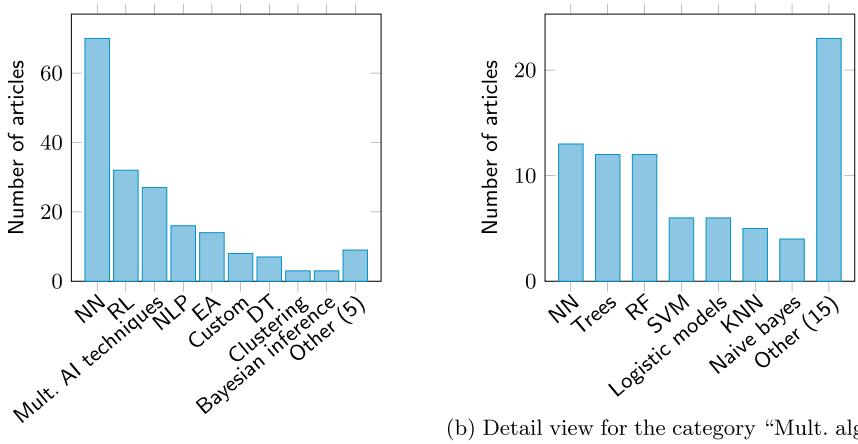


Fig. 7 Contribution type and publication venues. Studies with multiple contribution types are considered individually for each distinct contribution type



(a) Quantity of the AI types in contributions

(b) Detail view for the category "Mult. algorithms" of Fig. 8a

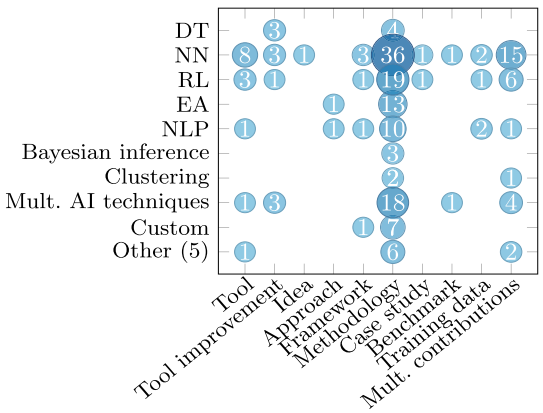
Fig. 8 Overview of AI contributions. Underrepresented AI types are aggregated as *Other* with a number in parentheses indicating how many sub-techniques were included

rithms that authors can train quickly. Further, they need less fine-tuning than NNs, making them a baseline comparison approach.

Intersecting RQs 2.1 and 1.3 Figure 9 shows the intersection of RQs 2.1 and 1.3. The wide focus on methodologies was already assessed with Fig. 6a. In Fig. 9, we can see that the amount of NN contributions remained focused on methodologies (37/189) or as part of a broader application of techniques (13/189) or a tool (8/189).

Intersecting RQs 2.1 and 1.1 Figure 10 show the intersection of RQs 2.1 and 1.1. Here, we can see that the amount of NN contributions remained steady over the years, similar to the amount of RL contributions. Noteworthy is that there is no notable decline in any technique over the observed period. The alternating nature of the overall amount of contributions was already found back in Section 5.1.

Fig. 9 Distribution of contribution type and AI techniques



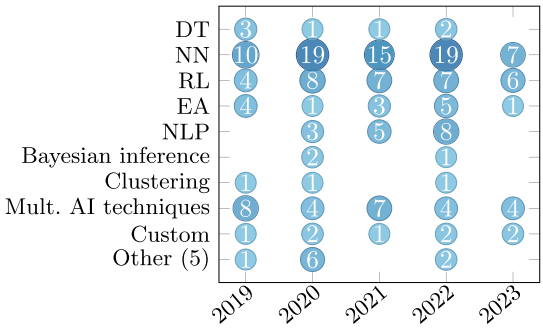


Fig. 10 Distribution of year and AI techniques

5.2.2 RQ 2.2: What are the Application Areas of AI in FM?

Figure 11a gives an overview of the FM techniques in the contributions. TP is leading by a large margin with 85 entries. SAT is next with 45 entries. Synthesis approaches come in third place with 19 entries, followed by model checking (18) and SMT solving (13). The remaining entries fall into algorithm selection, program analysis, and termination analysis.

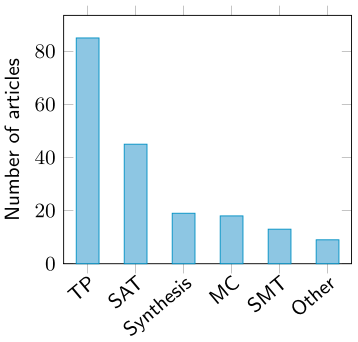
In Fig. 11b, we can see a more detailed investigation of the topic of theorem proving. Here, we can see that premise, axiom, and clause selection received the most attention (27/85), followed by proof search (21/85) and proof synthesis (18/85). After that, a significant gap exists in a row of smaller topics.

In Fig. 11c, we see the division of SAT approaches into their application areas. Overall, 37/45 studies focused on typical SAT solving, 2 on QBFs, and 6 on 3SAT. Out of 45 articles, 9 focused on solving, 8 on finding MaxSAT, 7 on solver selection, 6 on instance selection, 5 aimed to predict solvability, and 4 contributions aimed at analyzing the performance of solvers. A small group of contributions aimed to do multiple things, generated SAT problems, and developed branching heuristics aimed at parameter selection or dependency analysis.

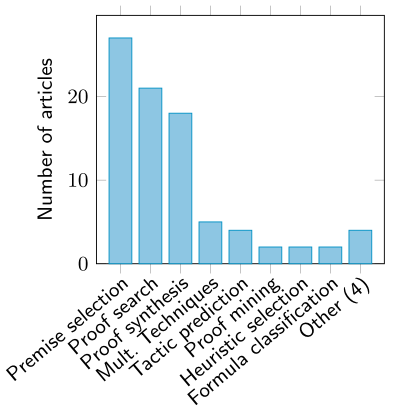
Figure 11d takes a closer look at the topic of synthesis. 8/19 contributions aimed to synthesize a model or specification, 4/19 aimed to learn a loop invariant, 3/19 aimed to repair models, and 3/19 targeted general invariant learning. One contribution targets the generation of annotations for the verification of JML code.

Detailed View on Theorem Proving A closer inspection is warranted, as TP makes up the most significant portion of contributions. For the studies concerned with TP, we distinguished between higher-order logic (47/85) and first-order logic (38/85). We further classified them by their automation levels: fully automatic TP (74/85), interactive TP (10/85), and contributions employing both (1/85). This granularity gives rise to additional observations about the problem structure. Figure 12a shows that the majority of contributions (26/85) were made in the area of premise selection for automatic theorem proving, followed by proof search (21/85) and proof synthesis (18/85).

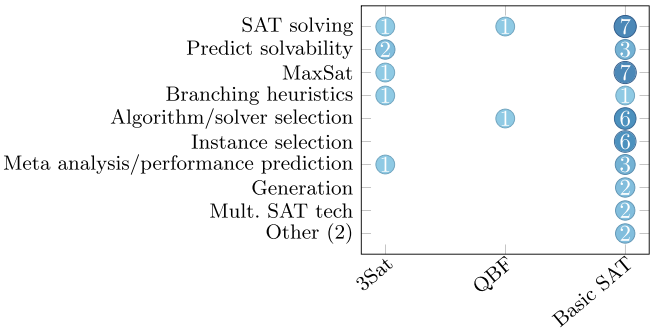
Figure 12b shows that of the 38 first-order entries, there is a tight focus on premise selection (20/38), proof search (8/38), and proof synthesis (7/38). Although not as drastic, a similar spread can be observed for the higher-order contributions. Premise selection has 7/47 entries, proof search 13/47, and proof synthesis 11/47 entries. 4/47 entries focused on tactics prediction.



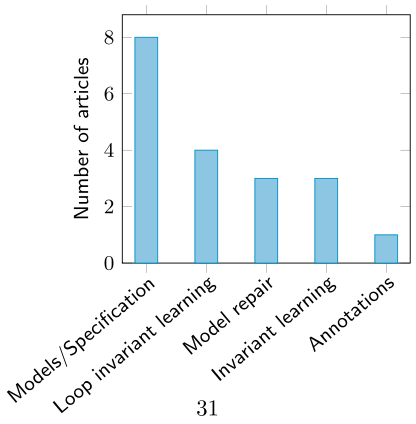
(a) Quantity of targeted FM types



(b) Quantity of targeted theorem-proving topics

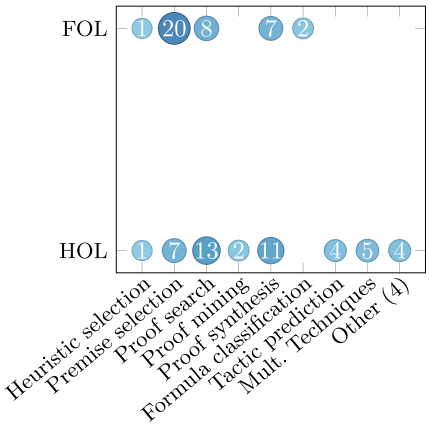


(c) Quantity of targeted SAT topics

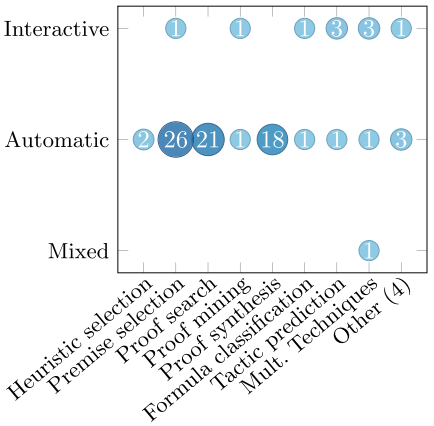


(d) Quantity of targeted synthesis topics

Fig. 11 Quantity of FM approaches with a detailed view of the most relevant contribution types



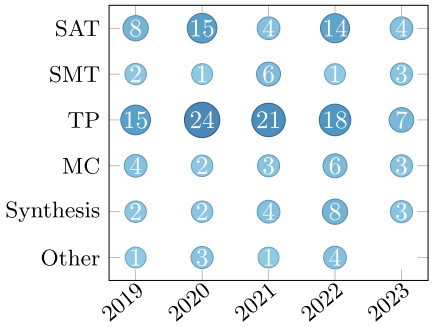
(a) Division of TP applications into first-order logic (FOL) and higher-order logic (HOL).



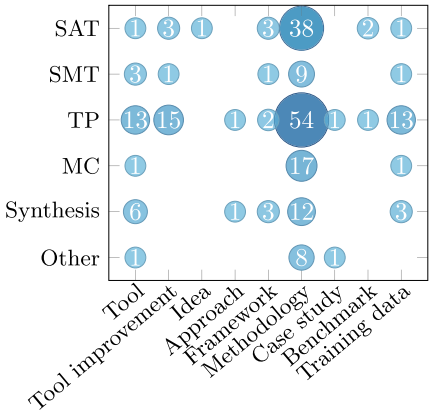
(b) Automation grade against the type of TP subcategory

Fig. 12 Theorem proving in detail

Intersecting RQ 2.2 with RQs 1.1 and 1.3 Figure 13a shows the different types of FM over the years. We can see no particular concentration on one year, similar to Fig. 10, However, there may be some trends. SAT had two solid years in 2020 and 2022, while TP had a slow decline; contributions to model checking and synthesis increased. Nonetheless, given the small time frame of five years, any actual trends might be invisible to us.



(a) Distribution of FM techniques over the years



(b) Distribution of FM techniques over contribution types. Studies with multiple contribution types are considered for each distinct contribution type.

Fig. 13 Quantity of FM approaches and comparison with time frame and contribution type

In Fig. 13b, we showcase the distribution of FM types over contribution types. While all FM techniques have found AI applications in the last five years, we now see a fragmented distribution of specific contribution types per discipline. In general, for TP, the different types of available contributions are the broadest, while for model checking, we are restricted to methodologies and one tool. The notable absence of tool improvements outside of TP might indicate developments in the respective subfields. There might be no developments that deem an extension of tools necessary, or no relevant tools can be extended. However, the underlying reasons for this observation are out of the scope of this mapping study. For the synthesis of formal models, we see a slightly better situation regarding a broader landscape of contribution types. Overall, however, the absence of case studies and benchmarks deserves attention and might highlight that the field is still in the early stages of development.

5.2.3 RQ 2.3: What is the Distribution of AI types in the Different FM Application Areas?

In Fig. 14, we see several overlays of FM applications with FM techniques at different granularities. Figure 14a is the most abstract. Here, we can see that the bulk of the theorem-proving techniques apply NNs (38), RL (16), and NLP (12). SAT solvers mainly utilize NNs (24). Model checking frequently uses EA (6) and NN (5). The area of synthesis is mainly divided between NNs (5), RL (4), and NLP (6) as well.

Taking a closer look at TP in Fig. 14b, we can see that NN is primarily used for the premise, axiom, and clause selection (17) proof search (7) and proof synthesis (10). RL techniques are often used for proof search as well (8). With three exceptions for heuristic selection, premise selection, or proof search, EA is absent from the area of theorem proving.

Focusing on SAT in Fig. 14c, we can see that NN is dominant in most subcategories, except for portfolio selection and branching heuristics. EA has higher relevance for MaxSat.

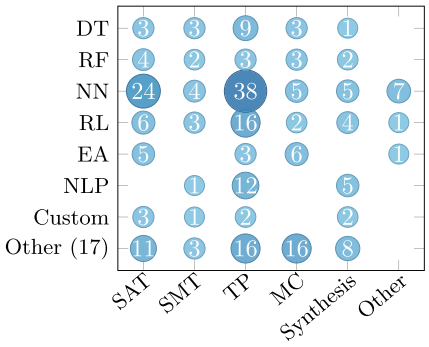
Finally, taking a closer look at the synthesis topic in Fig. 14d, we can see that NLP is relevant for the model generation and the generation of verification annotations. NN and RL are mainly used for (loop) invariant learning. For the repair of formal models, the contributions consider multiple techniques.

5.2.4 RQ 2.4: Are the Studies' Employed Data Sets Publicly Available?

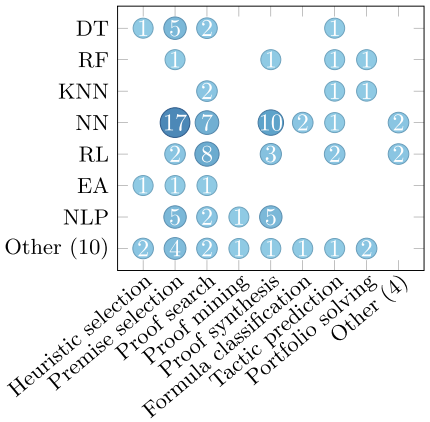
Lastly, we investigated the data sets that were found. Here, multiple observations took place.

First, we searched for high-quality data sets enabling other researchers to conduct their research efficiently. As pointed out earlier, we found 21 data sets. In Fig. 15, we see the respective scopes of the data sets. Of the 21 data sets, 15 focused on TP, while 3 aimed at synthesis, 1 at SAT, 1 at SMT, and 1 at MC.

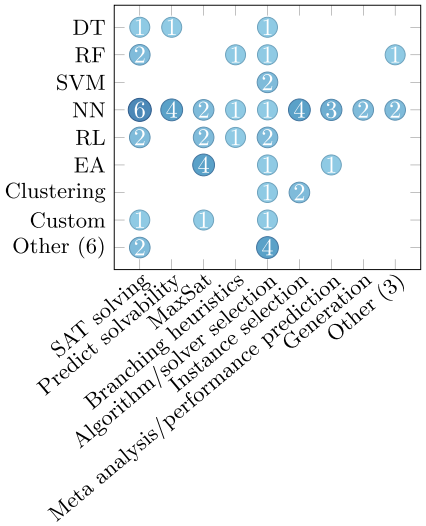
However, we were also interested in datasets that were not novel contributions but were found to be reused in other studies. For this, we skimmed all our studies for dataset mentions. As it turns out, of the 189 articles, 124 used external data sets. The 65 remaining studies generate random samples or do not mention their data source. Noteworthy is that when random data is used, the generated data is seldom shared, hindering the reproduction of results.



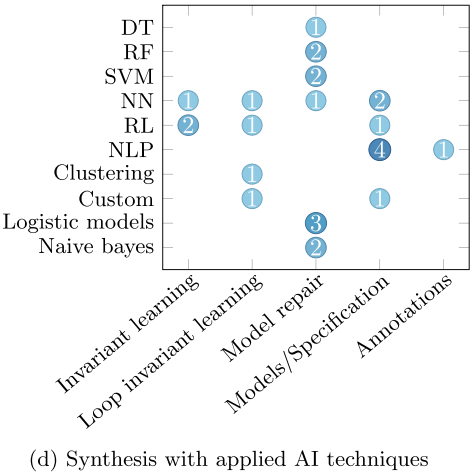
(a) FM techniques with applied AI techniques



(b) Theorem-proving techniques with applied AI techniques



(c) SAT with applied AI techniques



(d) Synthesis with applied AI techniques

Fig. 14 The quantity of FM approaches and comparison with a time frame and contribution type. Studies that applied multiple AI techniques are considered individually for each distinct AI technique. Underrepresented techniques are aggregated as *Other* with a number in parentheses indicating how many subtechniques were included

The datasets found for the 124 contributions are largely heterogeneous. This means that most studies utilized different datasets. However, we could find some data repositories that had multiple usages. The Mizar Library[6] (and associated libraries like, e.g., MPTP (Urban 2003)) saw 23 usages, TPTP Problem Library[7] has 7 mentions, The ProB

[6] https://mizar.uwb.edu.pl/library/

[7] https://tptp.org/TPTP/

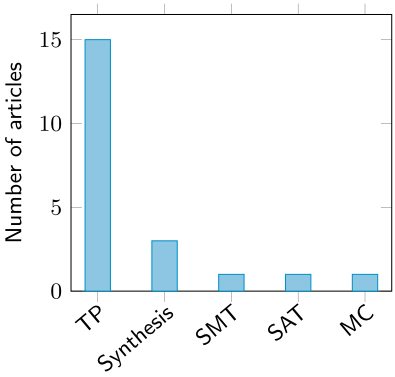


Fig. 15 Overview of the distribution of data sets over FM domains

machine library[8] got 5 mentions, HoL[9] 5, SMTLib[10] 5, SATComp[11] 5, SATLib[12] 4, CoqGym[13] 4. The full list is available in the repository linked in Section 1.

6 Discussion

In the following, we will evaluate the results and conclude by answering our research questions. Further, we will outline lessons learned to provide actionable insights for conducting future studies in the area.

6.1 RQ 1: Demographics of the Research Area

In RQs 1.1 to 1.3, we investigated the publication timeline for possible trends, and assessed the qualitative nature of the studies, i.e., the maturity of the publication format as well as their level of postulated contribution, following our taxonomies as defined in Section 2.4. While we were able to analyze RQs 1.1 and 1.2 over the full corpus of 457 publications, we restricted ourselves to the processed corpus of 189 publications from the last five years for content assessment as outlined in Section 4.5. Consequently, RQ 1.3 is limited to these 189 publications.

For RQ 1.1, we see a general growth in the research field. However, from the presented data points, it cannot be inferred alone whether this trend follows the overall trend of AI's rise in popularity or is independent.

For RQ 1.2, we see a strong focus on conference papers, short papers, and workshops. Together with the singular book chapter, this indicates an emerging field, as there is a tendency to rely on publication methods supporting short review cycles. This is further supported by the fact that the only book chapter was a case study. Furthermore, we see that the

[8] https://prob.hhu.de/w/index.php?title=Download

[9] https://isabelle.in.tum.de/library/

[10] https://smt-lib.org

[11] https://satcompetition.github.io

[12] https://www.cs.ubc.ca/~hoos/SATLIB/benchm.html

[13] https://github.com/princeton-vl/CoqGym

AI side of things mainly drives the topic, and especially FM-based conferences publish less content about the application of AI in FM. This could indicate hesitance to adapt or a lack of AI maturity. Contrasting this with RQ 1.1, we assume that the research interest will at least not plummet in the foreseeable future, but might plateau for the time being.

For RQ 1.3, we can see that the bulk of the work lies within the area of methodologies. This could be due to the current paper requirements, which request a novel approach and some evaluation within the paper. Another reason might be AI's applicative nature, which often allows it to serve as a support system to automate parts of the work. We also want to highlight the possibility that we primarily introduced methodology papers based on the nature of our search query, which focuses on the application of AI in the FM domain. However, due to rigorous snowballing, we are confident we correctly assessed the state of research in this field.

What is more interesting is what we do not see: many case studies and benchmarks. Recalling Fig. 7, we only see two publications providing case studies and two providing benchmarks. This may hint at a further gap in maturity, especially as these contribution types are primarily present in journal and book contributions. One would assume more such publications if a set of robust solutions for everyday problems were readily available so that researchers could compare their effectiveness against each other. It also highlights the need for proper and uniform benchmark sets within the community, which can be shared between different research groups.

6.2 RQ 2: Content type of Contributions

In RQ 2, we were interested in how the AI application is distributed in FM. For this, we posed RQs 2.1 to 2.4 to investigate which AI techniques find appreciation in the community, which FM areas are already subject to AI application, if certain AI techniques are predominantly used in specific FM fields, and whether there exist publicly available data sets that can be used in further research.

For RQ 2.1, we saw in Fig. 8 that most studies use NNs, followed by RL, and NLP and EAs are far behind. From a paradigmatic point of view, it makes sense to rank these four at the top, as they all attempt to solve different problem settings. NNs are a stand-in for classification and regression tasks, RL is applied for learning behavioral policies, NLP processes texts resembling natural language or code by extension, and EAs excel in optimization problems. However, it is interesting to see NNs being so far ahead of the rest with 83 publications utilizing them. Especially, since they are not the only algorithms suitable for classification and regression tasks, but competing algorithms such as KNN, SVM, or decision trees are, at least as solely applied AI techniques, vastly underrepresented. These algorithms, which we categorized as classical approaches in Section 2.3, are, however, not forgotten, as we can see in publications that contrast multiple different AI algorithms (Fig. 8b). This focus on NNs when only one technique is used can have multiple underlying reasons. First, implementing, training, and fine-tuning NNs is time-consuming and might require resources needed to investigate further algorithms. Second, we have seen in Fig. 3 that the research area gained traction in the early 2010s. As this coincides with the advent of deep learning (Sejnowski 2018), this might suggest

the need for NNs to overcome the complexities the area of FM offers, and that more classical ML algorithms are unsuitable for most tasks. Third, the community might be biased towards using NNs from the start instead of applying more straightforward techniques first.

However, the prominence of more classical algorithms in the case of multiple AI techniques shows that the community seems to be aware that other algorithms might be suitable for their problems as well. This is especially amplified by Fig. 9, showing that most frequently, methodologies rely on various techniques to solve problems.

Regarding maturity, Fig. 10 indicates no noteworthy increase in techniques besides NLP. For 2023, we established insufficient data points so that we may see a rise in NLP-based contributions, which would fit a more general advent of NLP-related contributions due to the rise of transformer architectures and LLMs.

For RQ 2.2, we have seen in Fig. 11a that TP and SAT are the predominant FM areas in which AI is applied, making up 68.8 % of all publications in our five-year corpus. We explain this by the fact that both areas are also of interest to non-FM communities. A detailed view of the authors' origin domains would be necessary to verify this assumption. However, this is still beneficial for the FM community as a whole. For instance, Fig. 12a shows a strong focus on FM-relevant daily use cases. Figure 12b reveals much current work regarding first-order logic on the especially relevant FM topics of premise selection. While this trend is good, complex FM models rely on complicated data types and require sophisticated reasoning.

Regarding SAT, we can take away from Fig. 11c that much focus lies directly on SAT solving. At the same time, other authors also attempt to predict satisfiability, i.e., whether a formula is SAT or UNSAT. The latter seems to contradict the rigorousness needed in FM, as the probabilistic nature of AI approaches lacks the certainty of the predicted satisfiability. This can be a product of cross-fertilization, as SAT is also relevant for non-FM communities. Here, a reasonable estimate about a formula's SAT or UNSAT may be more appreciated than in the FM community.

Figure 11d shows that the interest in synthesis is low compared to other topics and primarily focuses on model generation. Given the recent rise of generative AI approaches, we see room for promising research in this area.

A similar low interest seems to be in model checking. Figure 13b shows a relatively low number of contributions in that area. The direct comparison with the TP field is interesting, as MC also contributes very little to tool and tool contributions, hinting at less application-focused research. Figure 13a shows a relatively consistent interest in recent years, while Fig. 13b emphasizes how little is done for current practitioners in terms of tool (improvements), benchmarks, training data, and case studies.

For RQ 2.3, we saw in Fig. 14a that NNs are the undeniable favorite in almost all FM areas investigated. We already discussed why NNs might be more popular than other algorithms suitable for the same problem tasks above for RQ 2.1. However, this again suggests that the FM community mainly focuses on solving classification or regression problems rather than execution policies (RL) or direct processing of texts or code (NLP). The slight exception we see is in model checking, which focuses mainly on EAs, which interpret model checking as a sort of optimization problem, i.e., finding a path to a counter-example. From

building the larger corpus of 457 studies, we know from seeing various titles that we missed many EA papers due to our 5-year restriction. Presumably, EA was more prominently used before deep learning took off.

An interesting observation is the substantial absence of any data mining-related results, which might solve other problems still not covered by NNs, RL, NLP, or EA. In the 189 investigated studies, only three applied data mining techniques. It is again worth noting that we only investigated the years 2019–2023. Hence, data mining may have been more widespread before that, much like EA. More thorough data mining would be needed for a deeper insight, and only a secondary study would be required.

Finally, for RQ 2.4, we found 19 datasets (or training environment) contributions in our corpus of 189 studies. Considering the two separate, data set-only publications that did not pass IC 1 (see Section 4.5), this leaves us with 21 total data set contributions.

With the analyzed high heterogeneity of the used datasets, as already mentioned in Section 5.1.3, there may be a general underlying issue. There seems to be no defined benchmark *gold standard* against which to run new insights. Instead, results are run on the data sets at hand. Furthermore, while datasets are publicly available for TP and SAT solving, the number of usable, established, and available benchmark datasets for other research areas that fit the named criteria is minuscule.

Generally, we see a current trend where researchers use datasets as they see fit or even generate data without referencing the procedure or providing the dataset afterward. This trend fits the overall trend of AI being only weakly reproducible (Hutson 2018). However, this trend endangers the whole research branch, as reproducibility is a core part of science.

In conclusion, we strongly need unified defaults to allow reproducibility and introduce familiar benchmarks into the field. Possible issues hindering such a development might be found in the various existing formalisms, which are not necessarily used or even supported by individual research groups. Such boundaries, of course, weaken the impact a data set publication might have onto researchers working with differing formalisms.

6.3 Observed Objectives of AI Applications in FM

Generally, we see that the stark majority of publications use AI as an assistance tool to enable or enhance a respective FM tool's performance. That is, AI is used to find models, clauses, or lemmas. However, the found artifacts are then still checked by a formal tool.

With the notable exception of SAT solving, where 5 contributions directly predict the satisfiability of a formula with AI, we did not observe any focus on finding AI-generated guarantees for formal models to replace an FM tool. This seems only natural, as FM are about believable and reproducible guarantees. Therefore, giving most AI approaches a more prominent role comes with the price of admission: introducing determinism and uncertainty. For strictly formal-safety-related projects, this may be too much of a burden to pass.

Another observation we did not make was the presence of auto-active verification. Analyzing existing program code did not draw much attention, and the only remotely connected contribution was outside our search area.

6.4 Suggested Research Directions

From the given analysis, we can derive five areas of growth potential, which we outline below.

Development of Unified Benchmarks and Data Sets As pointed out in our discussion to RQ 2.4, we are concerned with the lack of established data sets and benchmarks in the FM sub-communities. While many contributions have already tackled this problem for TP, the other areas seem to fall short. This creates room for reproducibility issues between research groups and hinders a more precise comparison between results and established benchmarks that could be provided.

Developing strong baselines might increase comparability between different methods and reduce the entry hurdle for new researchers due to publicly available data sets. The main challenge is to create data sets that can be used for different formalisms. This would allow smaller communities to still benefit from the available data, which might be presented in a more popular formalism while furthering research in their area.

An orientation on how such a unified benchmark environment could look would be OpenAIGym (Brockman et al. 2016), a Toolkit for RL to foster research by enhancing reproducibility.

Investigate the Benefits of Data Mining for FM In the age of big data, we were expecting more studies that apply data mining techniques, be it for feature engineering or performance analytical approaches. This was not the case. While it might very well be that the kind of problems solvable by data mining are irrelevant to the FM communities, it is also possible that potential benefits are not well-known.

We propose to research whether and how data mining approaches can benefit FM. First, this would highlight potential benefits to the community. Second, due to the investigative nature of data mining, this might strengthen our insights into our tools and best practices. Consequently, this might further the quality of other AI applications in FM due to new knowledge.

Application of LLMs and Generative AI While LLMs, at this point, are established tools that have found their way into daily applications, we have found little use in our corpus of studies. This can be due to their relatively high employment costs. Another reason could be that respective studies are still published and invisible to our systematic mapping study. Nevertheless, we see strong potential for generative AI approaches, especially in synthesis and auto-formalization. However, simple test cases or documentation generation for existing models seems a promising starting point for building trust.

We envision substantial benefits from applying generative AI during FM development and highlight the need for more research. One can easily envision tool chains of automatic generation, checking, and documentation of formal models, logical formulas, or proofs with reproducible and provable results. However, we also see the need for more exploration regarding how generative AI can be helpful besides text production. For instance, how could generative AI be leveraged in proof mining? What stages in the formal development process

can benefit from generated inputs, and what types could and should be generated? Generative AI for FM may be thought of further than pure text and code generation and more toward reasoning support.

Increase the Potential of AI in Model Checking Another underrepresented topic is model checking. Anecdotally, we know model checking had much EA applied in the past, while it was a seemingly unpopular application area for AI in our five-year corpus. However, we see significant potential in revisiting the research with a modern lens. Especially in the context of bounded model checking (BMC) or statistical model checking (SMC) and faulty state prediction. BMC cuts off parts of a model's state space for performance and computability. SMC checks states until a certain confidence in the absence of faulty states is reached. Both approaches do not fully traverse the state space and can only indicate the absence of faults.

AI-enforced model checking could complement BMC, SMC, and model checking by predicting potential faulty states or needed search depths, learning search policies via RL to reach faulty states more targeted, or predicting equivalence classes of states for search space reduction. For example, after estimating a defective state or a search depth, the model checker could be applied in a very targeted manner to confirm the predicted state.

7 Threats to Validity

A study of this type and extent can only be conducted with the possibility of encountering threats that threaten validity. Zhou et al. (2020) serves as a baseline for this discussion, as their contribution meticulously lists the possible threats to validity and how they may be addressed.

Internal Validity A potential threat to the conclusion could be our selection of articles and our data source. To ensure the quality of our data, we carefully drafted our IC/EC, and in cases where we found them not yet sufficient, we enhanced them. Two experienced researchers, i.e., the first two authors, carefully read the publications.

Our data source was four famous known databases containing the relevant literature (Dyba et al. 2007). Due to the extensive snowballing, we made sure that the potential for blind spots, i.e., publications that are only listed by one engine or none at all, is minimized. This also holds for the case of WoS, where different result sizes are obtained. The divergence between the two results was marginal, so we are confident that excessive snowballing alleviates it. The snowballing also alleviated some of our rigor by applying IC/EC early in our search process. While we reduced our initial tally of 1492 publications to only 89, a reduction of two orders of magnitude, snowballing helped us recapture significant contributions into our database again.

Construct Validity Another threat is the selection and use of search terms. We used PICO as Kitchenham et al. (2007) suggest to select the appropriate search term. Nevertheless, we know it is impossible to cover all types of FM, AI, and all their acronyms in one query due to the technical limitations of the search engines and the practical

feasibility of thoroughly skimming the amount of papers found this way. Therefore, the aim was to keep the initial set small and conduct an exhaustive snowballing procedure to include all relevant publications.

The amount of snowballing results was unsurprisingly large, as snowballing does not suffer from a compromise for a search query. This is a limitation already pointed out by Wohlin (2014).

Conclusion Validity While we followed Kitchenham et al. (2007) guidelines for a successful mapping study to minimize the threat to the conclusion of this study, one remaining potential threat is whether or not to include grey literature, as we could have missed crucial contributions that change the outcome of this study. Grey literature was not considered for two main reasons. First, we did not encounter large quantities of grey literature during our snowballing. Thus, We are positive that we can represent current research directions without considering them. Second, we only did a soft skim of the content, and therefore, we can not offer a quality and feasibility assessment of the included contributions. As we aim to provide the reader with only high-quality studies, we decided to require official publications.

8 Related Work

While the lack of available literature partially motivated this work, there are noteworthy related works. One of the first contributions one finds when investigating the topic is that of Amrani et al. (2018). Here, the authors came to a similar conclusion as we did about the complicated nature of the investigated landscape of available literature. Therefore, they also resorted to a complex search string. The evaluation and focus, however, are different. The authors limit themselves to an excerpt of initial results and only conduct backward snowballing, leaving room for whether the corpus is complete. Additionally, the research questions are more specifically tailored to individual FM topics, while our work aims to give a general overview.

Wang et al. (2020) aims to give a general taxonomy of learning-based FM. One core contribution is that they divide the topic into two general directions: the learning of formal specifications and learning for formal verification. They proceeded to structure the literature found by individuals under this taxonomy. Compared to their work, our work aims to give a general introduction to the cross-section of the two fields, while the authors dive into the specifics of learning algorithms for formal verification.

Solvers and Solving Large quantities of work exist to evaluate the use of AI to enable solvers. A general overview was given by Ganesh et al. (2023). In the context of SAT, multiple works exist. A general overview with comments on the use of AI was presented by Kilani et al. (2013). Explicit SAT solving was subject to two studies (Holden 2021; Guo et al. 2023). Configuration and benchmarking of SAT solvers were investigated in three contributions (Granmo and Bouhmala 2010; Hoos et al. 2021; Fuchs et al. 2023).

We found many contributions that targeted the general constraint satisfaction problem during our search. However, we excluded these results as they are only loosely connected with FM. Amadini et al. (2013) conducted a general evaluation of portfolio approaches for

CSP problems. Popescu et al. (2022) gives an overview of machine learning techniques for CSP problems.

Proving Multiple works aim to provide an overview of the topic within the math and automated reasoning community. Urban and Vyskočil (2013) give an overview of AI-supported automated theorem proving. Blanchette et al. (2014) surveyed ML-supported axiom selection, and two years later, Blanchette et al. (2016) surveyed the rising interest in automatizing reasoning over proofs. England (2018) published a survey on ML for symbolic reasoning. Tran et al. (2022) gave a short overview of the emerging field of NLP for premise selection.

Program Analysis For general program analysis, there are works for runtime prediction by Hutter et al. (2014). Kumazawa et al. (2020) published a survey on applying swarm intelligence (a flavor of EAs) for software verification. Bennaceur and Meinke (2018) published on which ML techniques may be suitable for software analysis.

Model/specification Generation Brunello et al. (2019) and Buzhinsky (2019) published surveys on creating temporal logic from natural language, while Fuggitti and Chakraborti (2023) and Szegedy (2020) discuss the topic of auto-formalization of natural language requirements.

Others Less prominent were the topics of model repair, where Barriga et al. (2022) published a longer article about the state of the art. Haltermann and Wehrheim (2022) gives an overview of invariant generation. Pan and Mishra (2022) surveyed hardware vulnerability analysis via ML, a field to which we found no contributions. For model checking, Besbas et al. (2022) gave a brief literature review of four pages.

9 Conclusion

In this systematic mapping study, we present the results of our work on assessing the quantity of research in applying artificial intelligence to formal methods. Overall, we found 457, from which we selected 189 for a closer investigation. Concluding from this investigation, we see the current trend is yet to mature, as many contributions are making some practical applications. However, only a few studies aim to create theoretical groundwork, benchmarks, or case studies. Furthermore, we see a focus on theorem proving and SAT solving. Model checking and model synthesis are underrepresented compared to this. Most work uses neural nets, reinforcement learning, or a combination of multiple approaches. Noteworthy is the predominant degradation of AI to a support function within existing formal methods.

10 Supplementary information

The supplementary material provides an overview of the sorted, investigated primary studies. It consists of tables, the basis for the figures presented in this work.

Appendix A Investigated Studies

Table 1 Investigated primary studies sorted by their FM application

FM technique	Application	Articles	Count
SAT	SAT solving	Amizadeh et al. (2019); Atkari et al. (2020); Fournier et al. (2022); Li and Si (2022); Nejati et al. (2020); Selsam et al. (2019); Xu and Lieberherr (2022); Yolcu and Poczos (2019); Zhang et al. (2022); Sun et al. (2020)	10
	Predict solvability	Cameron et al. (2020); Chang et al. (2022); Selsam and Bjørner (2019); Xu et al. (2021); Zhang et al. (2020)	5
	MaxSat	Berden et al. (2022); Chang et al. (2020); Framil et al. (2022); Kumar et al. (2023); Lassouaoui et al. (2019); Marino (2021); Nurcahyadi et al. (2022); Sadeg et al. (2021); Zheng et al. (2022)	9
	SAT parameter selection	Beskyd and Surynek (2022)	1
	Branching heuristics	Kurin et al. (2020); Lorenz and Nickerl (2020)	2
	Algorithm/solver selection	Fournier et al. (2022); Fuchs et al. (2023); Lederman et al. (2020); Nejati et al. (2020); Richter et al. (2020); Sadeg et al. (2021); Sadreddin et al. (2022); Wang et al. (2019)	8
	Instance selection	Dalla et al. (2021); Danisovszky et al. (2020); Farooque and Keni (2023); Han (2020); Hireche and Drias (2019); Hireche et al. (2020)	6
	Dependency analysis	Yan et al. (2023)	1
	Meta analysis/performance prediction	Fu et al. (2020); Huang et al. (2022); Leventi-Peetz et al. (2020); Ozolins et al. (2022)	4
	Generation	Garzón et al. (2022); You et al. (2019)	2
	Model counting	Li and Si (2022)	1
SMT	Quantifier instantiation	Jakubův et al. (2023); Janota et al. (2022)	2
	Solver selection/scheduling	Blanchette et al. (2019); Dunkelau et al. (2019); Hůla et al. (2021); Leeson et al. (2023); Pimpalkhare et al. (2021); Pimpalkhare (2021); Scott et al. (2021, 2023)	8
	Quality assesment	Dunkelau et al. (2020); Scott et al. (2021); Yao et al. (2021)	3
TP	Heuristic selection	Holden and Korovin (2019); Nagashima (2019)	2
	Premise selection	Bártek and Suda (2020, 2021); Chvalovský et al. (2019, 2021); Chvalovskỳ et al. (2023); Crouse et al. (2021); Ferreira and Freitas (2020); Firoiu et al. (2021); Goertzel et al. (2019); Goertzel and Urban (2019); Goertzel et al. (2021, 2022); Han et al. (2021); Huang et al. (2019); Jakubuv and Urban (2019); Jakubův et al. (2020); Jiang et al. (2022); Liu et al. (2020, 2022); Mangla et al. (2022); Morris et al. (2022); Nawaz et al. (2021); Olsák et al. (2020); Piotrowski and Urban (2020); Shminke (2022); Suda (2021); Tworkowski et al. (2022); Wei (2022); Welleck et al. (2022); Zhang et al. (2023); Zombori et al. (2021)	31

Table 1 (continued)

FM technique	Application	Articles	Count
	Proof search	Abdelaziz et al. (2023); Baghdasaryan and Bolibekyan (2021); Bansal et al. (2019); Blaauwbroek et al. (2020); Färber et al. (2020); Gauthier et al. (2020); Goertzel (2020); Lample et al. (2022); Mo et al. (2020); Nawaz et al. (2021); Paliwal et al. (2020); Piepenbrock et al. (2021, 2022); Piotrowski and Urban (2020); Rawson and Reger (2019); Sanchez-Stern et al. (2020); Urban and Jakubův (2020); Wang et al. (2023); Wu et al. (2021); Zombori et al. (2019, 2020, 2021)	22
	Proof mining	Jiang et al. (2021); Nawaz et al. (2020)	2
	Proof synthesis	Aygün et al. (2022); First et al. (2020); First and Brun (2022); Gauthier (2020); Glorot et al. (2019); Han et al. (2022); Kommrusch et al. (2023); Laurent and Platzer (2022); Palermo et al. (2022); Poesia and Goodman (2023); Qian (2021); Rawson and Reger (2021); Sanchez-Stern et al. (2023); Wang and Deng (2020); Wu et al. (2020, 2022a, 2022b); Yang and Deng (2019); Zhang et al. (2023)	19
	Symbolic	Poesia et al. (2021)	1
	Formula synthesis	Brown and Gauthier (2020)	1
	Formula classification	PurgaŁ et al. (2021); Suda (2021)	2
	Tactic prediction	Nawaz et al. (2019); Wu et al. (2020, 2021); Zhang et al. (2019, 2021)	5
	Lemma name generation	Nie et al. (2020)	1
	Portfolio solving	Nikolić et al. (2019)	1
	Position prediction	Huang et al. (2019)	1
	Symbol guessing	Olsák et al. (2020)	1
MC		Cao et al. (2022); Dunkelau and Baldus (2021); Gross et al. (2022); Hu et al. (2023); Jaeger et al. (2020); Kumazawa et al. (2019, 2021, 2023); Luo et al. (2023); Ma et al. (2019); Pira (2022); Rafe et al. (2019); Tsutomu Kumazawa and Kambayashi (2022); Wang et al. (2020, 2021); Zhu et al. (2019); Zhu (2022); Zhu and Wu (2022)	18
Synthesis	Invariant learning	Kobayashi et al. (2021); Liu et al. (2023); Wang and Wang (2023)	3
	Loop invariant learning	Laurent and Platzer (2021); Xu et al. (2022); Yao et al. (2020); Yu et al. (2023)	4
	Model repair	Cai et al. (2019a, 2019b, 2022)	3
	Models/Specification	Abate et al. (2022); Bordg et al. (2022); Cherukuri et al. (2022); He et al. (2022); Hu et al. (2022); Koscinski et al. (2021); Nayak et al. (2022); Shigyo and Katayama (2020)	8
	Annotations	Puccetti et al. (2021)	1
Other	Algorithm selection	Richter and Wehrheim (2020); Wang et al. (2021)	2
	Termination anlysis	Giacobbe et al. (2022)	1
	Program analysis	Allamanis (2022); Chen et al. (2019); Kumazawa et al. (2022)	3
	LTL satisfiability	Luo et al. (2022)	1
	General verification	Cao et al. (2020); Si et al. (2020)	2
Unique total			189

Table 2 Datasets

FM	Articles	Count
SAT	You et al. (2019)	1
SMT	Dunkelau et al. (2019)	1
TP	Bansal et al. (2019); Blaauwbroek et al. (2020); Goertzel et al. (2022); Han et al. (2022); Jiang et al. (2021); Nagashima (2020); Piotrowski and Urban (2020); PurgaŁ et al. (2021); Reichel et al. (2023); Shminke (2022); Urban and Jakubův (2020); Wang and Deng (2020); Wu et al. (2020, 2020); Yang and Deng (2019)	15
MC	Wang et al. (2021)	1
Synthesis	Bordg et al. (2022); Cai et al. (2019a); He et al. (2022)	3
Total		21

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10664-025-10729-8.

Author Contributions The first author conceived the idea of conducting the systematic mapping study. The first and second authors conducted the search process outlined in Section 4 together. The application of IC/EC was done independently by the first and second authors. Disagreements in this filtering process were solved by the third author. The first and second authors were responsible for the manuscript's writing. The third author reviewed the manuscript, provided feedback, and supervised the project. The public repository containing the dataset of found and investigated studies was prepared by the second author.

Funding Open Access funding enabled and organized by Projekt DEAL. The authors have no relevant financial or non-financial interests to disclose.

Data Availibility Statement The primary studies found in this systematic mapping study are publicly available at https://github.com/hhu-stups/ai4fm-studies.

Declarations

Conflicts of Interest The authors have no competing interests to declare that are relevant to the content of this article.

Ethical Approval Not applicable.

Informed Consent Not applicable.

Clinical Trial Number Not applicable.

References

Abate A, Edwards A, Giacobbe M (2022) Neural abstractions. In: Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K, Oh A (eds.) Advances in neural information processing systems 35 (NeurIPS 2022), vol 35, pp 26432–26447. Curran Associates, Inc., New Orleans, LA, USA. https://proceedings.neurips.cc/paper_files/paper/2022/file/a922b7121007768f78f770c404415375-Paper-Conference.pdf

Abdelaziz I, Crouse M, Makni B, Austel V, Cornelio C, Ikbal S, Kapanipathi P, Makondo N, Srinivas K, Witbrock M, Fokoue A (2023) Learning to guide a saturation-based theorem prover. IEEE Trans Pattern Anal Mach Intell 45(1):738–751. https://doi.org/10.1109/TPAMI.2022.3140382

Abran A, Moore JW, Bourque P, Dupuis R (eds) (2004) Software engineering body of knowledge vol 25. IEEE Computer Society, Los Alamitos, CA, USA. https://www.computer.org/education/bodies-of-knowledge/software-engineering

Allamanis M (2022) In: Wu L, Cui P, Pei J, Zhao L (eds) Graph neural networks in program analysis, pp 483–497. Springer, Singapore. https://doi.org/10.1007/978-981-16-6054-2_22

Amadini R, Gabbrielli M, Mauro J (2013) An empirical evaluation of portfolios approaches for solving CSPs. In: Gomes C, Sellmann M (eds.) Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, pp 316–324. Springer, Yorktown Heights, NY, USA.https://doi.org/10.1007/978-3-642-38171-3_21

Amizadeh S, Matusevych S, Weimer M (2019) Learning to solve Circuit-SAT: An unsupervised differentiable approach. In: International conference on learning representations. OpenReview.net, New Orleans, LA, USA. https://openreview.net/forum?id=BJxgz2R9t7

Amrani M, Lúcio L, Bibal A (2018) ML + FV= ♡? a survey on the application of machine learning to formal verification. arXiv:1806.03600 [cs.SE]

Atkari A, Dhargalkar N, Angne H (2020) Employing machine learning models to solve uniform random 3-SAT. In: Jain LC, Tsihrintzis GA, Balas VE, Sharma DK (eds.) Data Communication and Networks. Advances in Intelligent Systems and Computing, vol. 1049, pp 255–264. Springer, New Delhi, India. https://doi.org/10.1007/978-981-15-0132-6_17

Aygün E, Anand A, Orseau L, Glorot X, Mcaleer SM, Firoiu V, Zhang LM, Precup, D, Mourad S (2022) Proving theorems using incremental learning and hindsight experience replay. In: Chaudhuri K, Jegelka S, Song L, Szepesvari C, Niu G, Sabato S (eds) Proceedings of the 39th international conference on machine learning. Proceedings of machine learning research, vol 162, pp 1198–1210. PMLR, Baltimore, MA, USA. https://proceedings.mlr.press/v162/aygun22a.html

Baghdasaryan A, Bolibekyan H (2021) On recurrent neural network based theorem prover for first order minimal logic. J Univ Comput Sci 27(11):1193–1202. https://doi.org/10.3897/jucs.76563

Baier C, Katoen J-P (2008) Principles of Model Checking. MIT press, Cambridge, MA, USA. https://mitpress.mit.edu/9780262026499/principles-of-model-checking/

Bansal K, Loos S, Rabe M, Szegedy C, Wilcox S (2019) HOList: An environment for machine learning of higher order logic theorem proving. In: Chaudhuri K, Salakhutdinov R (eds) Proceedings of the 36th international conference on machine learning. proceedings of machine learning research, vol 97, pp 454–463. PMLR, Long Beach, CA, USA. https://proceedings.mlr.press/v97/bansal19a.html

Barenkamp M, Rebstadt J, Thomas O (2020) Applications of AI in classical software engineering. AI Perspectives 2. https://doi.org/10.1186/s42467-020-00005-4

Barrett C, Tinelli C (2018) Satisfiability Modulo Theories. In: Clarke EM, Henzinger TA, Veith H, Bloem R (eds) Handbook of Model Checking, pp 305–343. Springer, Cham. https://doi.org/10.1007/978-3-319-10575-8_11

Barriga A, Rutle A, Heldal R (2022) AI-powered model repair: an experience report–lessons learned, challenges, and opportunities. Softw Syst Model 21(3):1135–1157. https://doi.org/10.1007/s10270-022-00983-5

Bártek F, Suda M (2020) Learning precedences from simple symbol features. In: Fontaine P, Korovin K, Kotsireas IS, Rümmer P, Tourret S (eds.) PAAR+SC-Square 2020 Practical Aspects of Automated Reasoning and Satisfiability Checking and Symbolic Computation Workshop 2020. CEUR Workshop Proceedings, vol. 2752, pp 21–33. CEUR-WS.org, Virtual Event. https://ceur-ws.org/Vol-2752/paper2.pdf

Bártek F, Suda M (2021) Neural precedence recommender. In: Platzer A, Sutcliffe G (eds.) Automated Deduction – CADE 28. Lecture Notes in Computer Science, vol. 12699, pp 525–542. Springer, Virtual Event. https://doi.org/10.1007/978-3-030-79876-5_30

Bennaceur A, Meinke K (2018) In: Bennaceur A, Hähnle R, Meinke K (eds.) Machine Learning for Software Analysis: Models, Methods, and Applications. Lecture Notes in Computer Science, vol. 11026, pp 3–49. Springer, Dagstuhl Castle, Germany. https://doi.org/10.1007/978-3-319-96562-8_1

Berden S, Kumar M, Kolb S, Guns T (2022) Learning MAX-SAT models from examples using genetic algorithms and knowledge compilation. In: 28th International conference on principles and practice of constraint programming. Leibniz International Proceedings in Informatics, LIPIcs, vol. 235, pp 1–17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Haifa, Israel. https://doi.org/10.4230/LIPIcs.CP.2022.8

Bertot Y, Castéran P (2013) Interactive Theorem Proving and Program Development, 1st edn Texts in Theoretical Computer Science. An EATCS Series. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-07964-5

Besbas A, Belaiche L, Slatnia S, Kahloul L, Khalgui M (2022) Machine learning solutions to model checking: A brief literature review. In: 2022 International Symposium on iNnovative Informatics of Biskra (ISNIB), pp 1–4. IEEE, Biskra, Algeria. https://doi.org/10.1109/ISNIB57382.2022.10076160

Beskyd F, Surynek P (2022) Parameter setting in SAT solver using machine learning techniques. In: Rocha A, Steels L, VandenHerik J (eds.) Proceedings of the 14th International Conference on Agentes and Artifical Inteligence. ICAART, vol. 2, pp 586–597. SciTePress, Virtual Event. https://doi.org/10.5220/0010910200003116

Biere A, Heule M, Maaren H (eds) (2015) Handbook of Satisfiability, 2nd edn. Frontiers in artificial intelligence and applications, vol 336. IOS press, Amsterdam. https://ebooks.iospress.nl/volume/handbook-of-satisfiability-second-edition

Blaauwbroek L, Urban J, Geuvers H (2020) Tactic learning and proving for the Coq proof assistant. In: Albert E, Kovacs L (eds) 23rd International conference on logic for programming, artificial intelligence and reasoning. EPiC series in computing, vol 73, pp 138–150. EasyChair, Virtual Event. https://doi.org/10.29007/wg1q

Blanchette JC, El Ouraoui D, Fontaine P, Kaliszyk C (2019) Machine learning for instance selection in SMT solving. In: 4th Conference on artificial intelligence and theorem proving. AITP Conference, Obergurgl, Austria. https://aitp-conference.org/2019/abstract/paper%2017.pdf

Blanchette JC, Kühlwein D, Geschke S, Loewe B, Schlicht P (2014) A survey of axiom selection as a machine learning problem. In: Stefan Geschke, P.S. Benedikt Loewe (ed.) Infinity, Computability, and Metamathematics: Festschrift Celebrating the 60th Birthdays of Peter Koepke and Philip Welch, pp. 1–15. College Publications, London, UK. https://hdl.handle.net/2066/132733

Blanchette JC, Greenaway D, Kaliszyk C, Kühlwein D, Urban J (2016) A learning-based fact selector for Isabelle/HOL. J Autom Reason 57:219–244. https://doi.org/10.1007/s10817-016-9362-8

Bordg A, Stathopoulos YA, Paulson LC (2022) A parallel corpus of natural language and isabelle artefacts. In: 7th Conference on artificial intelligence and theorem proving. AITP Conference, Aussois and online, France. https://aitp-conference.org/2022/abstract/AITP_2022_paper_8.pdf

Breiman L (2001) Random forests. Mach Learn 45(1):5–32. https://doi.org/10.1023/a:1010933404324

Breiman L, Friedman J, Stone CJ, Olshen RA (1984) Classification and Regression Trees. Taylor & Francis New York NY. https://doi.org/10.1201/9781315139470

Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J, Zaremba W (2016) OpenAI Gym. arXiv:1606.01540 [cs.LG]

Brown CE, Gauthier T (2020) Self-learned formula synthesis in set theory. In: 5th Conference on artificial intelligence and theorem proving. AITP Conference, Aussois, France. https://aitp-conference.org/2020/abstract/paper_1.pdf

Brown CE (2013) Reducing higher-order theorem proving to a sequence of SAT problems. J Automated Reason 51:57–77. https://doi.org/10.1007/s10817-013-9283-8

Brunello, A., Montanari A, Reynolds M (2019) Synthesis of LTL Formulas from Natural Language Texts: State of the Art and Research Directions. In: Gamper J, Pinchinat S, Sciavicco G (eds.) 26th International Symposium on Temporal Representation and Reasoning (TIME 2019). Leibniz International Proceedings in Informatics (LIPIcs), vol. 147, pp 1–19. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Málaga, Spain. https://doi.org/10.4230/LIPIcs.TIME.2019.17

Buzhinsky I (2019) Formalization of natural language requirements into temporal logics: a survey. In: 2019 IEEE 17th International conference on industrial informatics (INDIN), pp 400–406. IEEE, Helsinki, Finland. https://doi.org/10.1109/INDIN41052.2019.8972130

Cai C-H, Sun J, Dobbie G (2019a) Automatic B-model repair using model checking and machine learning. Autom Softw Eng 26(3):653–704. https://doi.org/10.1007/s10515-019-00264-4

Cai C-H, Sun J, Dobbie G (2022) B model quality assessments on automated reachability repair with ISO/IEC 25010. Sci Comput Program 214:102732. https://doi.org/10.1016/j.scico.2021.102732

Cai C-H, Sun J, Dobbie G, Lee SU-J (2019b) Achieving abstract machine reachability with learning-based model fulfilment. In: 2019 26th Asia-Pacific software engineering conference (APSEC), Putrajaya, Malaysia, pp 260–267. https://doi.org/10.1109/APSEC48747.2019.00043

Cameron C, Chen R, Hartford J, Leyton-Brown K (2020) Predicting propositional satisfiability via end-to-end learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp 3324–3331. AAAI Press, New York, NY, USA.https://doi.org/10.1609/aaai.v34i04.5733

Cao W, Wu Y, Wang Q, Zhang J, Zhang X, Qiu M (2022) A novel RVFL-based algorithm selection approach for software model checking. In: Memmi G, Yang B, Kong L, Zhang T, Qiu M (eds) Knowledge science, engineering and management. Lecture notes in computer science, vol 13370, pp 414–425. Springer, Singapore. https://doi.org/10.1007/978-3-031-10989-8_33

Cao W, Xie Z, Zhou X, Xu Z, Zhou C, Theodoropoulos G, Wang Q (2020) A learning framework for intelligent selection of software verification algorithms. J Artif Intell 2(4):177–187. https://doi.org/10.32604/jai.2020.014829

Chang K-H (2023) Artificial intelligence and information processing: A systematic literature review. Mathematics 11(11). https://doi.org/10.3390/math11112420

Chang W, Zhang H, Luo J (2022) Predicting propositional satisfiability based on graph attention networks. Int J Computat Intell Syst 15(1):84. https://doi.org/10.1007/s44196-022-00139-9

Chang Y, Wang X, Wang J, Wu Y, Yang L, Zhu K, Chen H, Yi X, Wang C, Wang Y, Ye W, Zhang Y, Chang Y, Yu PS, Yang Q, Xie X (2024) A survey on evaluation of large language models. ACM Trans Intell Syst Technol 15(3):1–45. https://doi.org/10.1145/3641289

Chang O, Flokas L, Lipson H, Spranger M (2020) Assessing SATNet's ability to solve the symbol grounding problem. In: Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin H (eds.) Advances in Neural Information Processing Systems, vol. 33, pp 1428–1439. Curran Associates, Inc., Virtual Event. https://proceedings.neurips.cc/paper_files/paper/2020/file/0ff8033cf9437c213ee13937b1c4c455-Paper.pdf

Chen J, Wei J, Feng Y, Bastani O, Dillig I (2019) Relational verification using reinforcement learning. Proceedings of the ACM on programming languages 3(OOPSLA), 1–30. https://doi.org/10.1145/3360567

Cherukuri H, Ferrari A, Spoletini P (2022) Towards explainable formal methods: From LTL to natural language with neural machine translation. In: Gervasi V, Vogelsang A (eds.) International working conference on requirements engineering: foundation for software quality. Lecture notes in computer science, vol 13216, pp 79–86. Springer, Birmingham, UK. https://doi.org/10.1007/978-3-030-98464-9_7

Chowdhary KR (2020) Natural Language Processing. Fundamentals of Artificial Intelligence, pp 603–649. Springer, New Delhi. https://doi.org/10.1007/978-81-322-3972-7_19

Chvalovský K, Jakubův J, Olšák M, Urban J (2021) Learning theorem proving components. In: Anupam Das SN (ed.) Automated Reasoning with Analytic Tableaux and Related Methods. Lecture Notes in Computer Science, vol. 12842, pp 266–278. Springer, Birmingham, UK. https://doi.org/10.1007/978-3-030-86059-2_16

Chvalovský K, Jakubův J, Suda M, Urban J (2019) ENIGMA-NG: Efficient neural and gradient-boosted inference guidance for E. In: Fontaine P (ed.) International Conference on Automated Deduction. Lecture Notes in Computer Science, vol. 11716, pp 197–215. Springer, Natal, Brazil. https://doi.org/10.1007/978-3-030-29436-6_12

Chvalovský K, Korovin K, Piepenbrock J, Urban J (2023) Guiding an instantiation prover with graph neural networks. In: Piskac R, Voronkov A (eds.) Proceedings of 24th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning. EPiC Series in Computing, vol. 94, pp 112–123. EasyChair, Manizales, Colombia. https://doi.org/10.29007/tp23

Clarke EM, Henzinger TA, Veith H, Bloem R et al (2018) Handbook of Model Checking. Springer Cham. https://doi.org/10.1007/978-3-319-10575-8

Clarke EM, Wing JM (1996) Formal methods: State of the art and future directions. ACM Comput Surv 28(4):626–643. https://doi.org/10.1145/242223.242257

Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13(1):21–27. https://doi.org/10.1109/TIT.1967.1053964

Crouse M, Abdelaziz I, Makni B, Whitehead S, Cornelio C, Kapanipathi P, Srinivas K, Thost V, Witbrock M, Fokoue A (2021) A deep reinforcement learning approach to first-order logic theorem proving. In: Proceedings of the AAAI conference on artificial intelligence, vol. 35, pp 6279–6287. AAAI Press, Virtual Event. https://doi.org/10.1609/aaai.v35i7.16780

Dalla M, Visentin A, O'Sullivan B (2021) Automated SAT problem feature extraction using convolutional autoencoders. In: 2021 IEEE 33rd International conference on tools with artificial intelligence (ICTAI), pp 232–239. IEEE, Washington, DC, USA. https://doi.org/10.1109/ictai52525.2021.00039

Danisovszky M, Yang ZG, Kusper G (2020) Classification of SAT problem instances by machine learning methods. In: Kovásznai G, István F, Tómács T (eds.) Proceedings of the 11th International Conference on Applied Informatics. CEUR Workshop Proceedings, vol. 2650, pp 94–104. CEUR-WS.org, Eger, Hungary. https://ceur-ws.org/Vol-2650/paper11.pdf

Dunkelau J, Baldus L (2021) Ranking model checking backends for automated selection via classification and regression learning. In: Monica DD, Pozzato GL, Scala E (eds) OVERLAY'21: Workshop on artificial intelligence and fOrmal VERification, Logic, Automata, and sYnthesis, 22nd September, 2021, Padova, Italy. CEUR Workshop Proceedings, vol 2987, pp 77–82. CEUR Workshop Proceedings, Padua, Italy. https://ceur-ws.org/Vol-2987/paper14.pdf

Dunkelau J, Krings S, Schmidt J (2019) Automated backend selection for ProB using deep learning. In: Julia M, Badger KYR (ed.) NASA Formal Methods Symposium. Lecture Notes in Computer Science, vol. 11460, pp 130–147. Springer, Houston, TX, USA. https://doi.org/10.1007/978-3-030-20652-9_9

Dunkelau J, Schmidt J, Leuschel M (2020) Analysing ProB's constraint solving backends. In: Alexander Raschke FH Dominique Méry (ed.) Rigorous State-Based Methods: 7th International Conference, ABZ 2020. Lecture Notes in Computer Science, vol. 12071, pp 107–123. Springer, Ulm, Germany. https://doi.org/10.1007/978-3-030-48077-6_8

Dyba T, Dingsoyr T, Hanssen GK (2007) Applying systematic reviews to diverse study types: An experience report. In: First international symposium on empirical software engineering and measurement (ESEM 2007), pp 225–234. IEEE, Madrid, Spain. https://doi.org/10.1109/ESEM.2007.59

Elahi M, Afolaranmi SO, Martinez Lastra JL, Perez Garcia JA (2023) A comprehensive literature review of the applications of AI techniques through the lifecycle of industrial equipment. Discover Artif Intell 3. https://doi.org/10.1007/s44163-023-00089-x

England M (2018) Machine learning for mathematical software. In: Davenport JH, Kauers M, Labahn G, Urban J (eds.) Mathematical Software – ICMS 2018. Lecture Notes in Computer Science, vol. 10931, pp 165–174. Springer, South Bend, IN, USA. https://doi.org/10.1007/978-3-319-96418-8_20

Färber M, Kaliszyk C, Urban J (2020) Machine learning guidance for connection tableaux. J Autom Reason 65(2):287–320. https://doi.org/10.1007/s10817-020-09576-7

Farooque M, Keni A (2023) A brief analogy of NNs and literal selection procedures within SAT solvers. In: International conference on life sciences, engineering and technology. International Society for Technology, Education, and Science (ISTES), Denver, CO, USA. https://scholarsphere.psu.edu/resources/d1c83077-8e2f-40e7-94fb-4026b828b7b9

Ferreira D, Freitas A (2020) Premise selection in natural language mathematical texts. In: Jurafsky D, Chai J, Schluter N, Tetreault J (eds.) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp 7365–7374. Association for Computational Linguistics, Virtual Event. https://doi.org/10.18653/v1/2020.acl-main.657

Firoiu V, Aygun E, Anand A, Ahmed Z, Glorot X, Orseau L, Zhang L, Precup D, Mourad S (2021) Training a first-order theorem prover from synthetic data. In: 1st Mathematical reasoning in general artificial intelligence workshop, ICLR 2021. Math-AI Organizers, Virtual Event. https://mathai-iclr.github.io/papers/papers/MATHAI_18_paper.pdf

First E, Brun Y (2022) Diversity-driven automated formal verification. In: Proceedings of the 44th international conference on software engineering. ICSE '22, pp 749–761. Association for Computing Machinery, Pittsburgh, PA, USA. https://doi.org/10.1145/3510003.3510138

First E, Brun Y, Guha A (2020) TacTok: Semantics-aware proof synthesis. Proceed ACM Program Languages 4(OOPSLA):1–31. https://doi.org/10.1145/3428299

Fournier T, Lallouet A, Cropsal T, Glorian G, Papadopoulos A, Petitet A, Perez G, Sekar S, Suijlen W (2022) A deep reinforcement learning heuristic for SAT based on antagonist graph neural networks. In: 2022 IEEE 34th International conference on tools with artificial intelligence (ICTAI), pp 1218–1222. IEEE, Macao, China. https://doi.org/10.1109/ICTAI56018.2022.00185

Framil M, Cabalar P, Santos J (2022) A MaxSAT solver based on differential evolution (preliminary report). In: Marreiros G, Martins B, Paiva A, Ribeiro B, Sardinha A (eds.) Progress in Artificial Intelligence. Lecture Notes in Computer Science, vol. 13566, pp 676–687. Springer, Lisbon, Portugal. https://doi.org/10.1007/978-3-031-16474-3_55

Fu H, Xu Y, Chen S, Liu J (2020) Improving WalkSAT for random 3-SAT problems. J Univ Comput Sci 26(2):220–243. https://doi.org/10.3897/jucs.2020.013

Fuchs T, Bach J, Iser M (2023) Active learning for SAT solver benchmarking. In: Sankaranarayanan S, Sharygina N (eds.) Tools and Algorithms for the Construction and Analysis of Systems. Lecture Notes in Computer Science, vol. 13993, pp. 407–425. Springer, Paris, France. https://doi.org/10.1007/978-3-031-30823-9_21

Fuggitti F, Chakraborti T (2023) NL2LTL–a python package for converting natural language (NL) instructions to linear temporal logic (LTL) formulas. In: AAAI-23 Special Programs, IAAI-23, EAAI-23, Student Papers and Demonstrations, vol. 37. AAAI Press, Washington, DC, USA. https://doi.org/10.1609/aaai.v37i13.27068

Gallier JH (2015) Logic for Computer Science: Foundations of Automatic Theorem Proving, 2nd edn. Courier Dover Publications, USA. https://store.doverpublications.com/products/9780486780825

Ganesh V, Seshia SA, Jha S (2023) Machine learning and logic: a new frontier in artificial intelligence. Formal Methods Syst Design 60:1–26. https://doi.org/10.1007/s10703-023-00430-1

Garzón I, Mesejo P, Giráldez-Cru J (2022) On the performance of deep generative models of realistic SAT instances. In: Meel KS, Strichman O (eds.) 25th International Conference on Theory and Applications of Satisfiability Testing (SAT 2022). Leibniz International Proceedings in Informatics (LIPIcs), vol. 236, pp 1–19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Haifa, Israel. https://doi.org/10.4230/LIPIcs.SAT.2022.3

Gauthier T (2020) Deep reinforcement learning for synthesizing functions in higher-order logic. In: Albert E, Kovacs L (eds) 3rd International conference on logic for programming, artificial intelligence and reasoning. EPiC Series in Computing, vol 73, pp 230–248. EasyChair, Virtual Event. https://doi.org/10.29007/7jmg

Gauthier T, Kaliszyk C, Urban J, Kumar R, Norrish M (2020) TacticToe: Learning to prove with tactics. J Autom Reason 65(2):257–286. https://doi.org/10.1007/s10817-020-09580-x

Giacobbe M, Kroening D, Parsert J (2022) Neural termination analysis. In: Roychoudhury A, Cadar C, Kim M (eds) Proceedings of the 30th ACM joint european software engineering conference and symposium on the foundations of software engineering. ESEC/FSE 2022, pp 633–645. Association for Computing Machinery, Singapore, Singapore. https://doi.org/10.1145/3540250.3549120

Glorot X, Anand A, Aygun E, Mourad S, Kohli P, Precup D (2019) Learning representations of logical formulae using graph neural networks. In: Workshop on graph representation learning at 33rd neural information processing systems. Graph Representation Learning Committee, Vancouver, Canada. https://grlearning.github.io/papers/58.pdf

Goertzel ZA (2020) Make E smart again (short paper). In: Peltier N, Sofronie-Stokkermans V (eds) Automated reasoning: 10th international joint conference, IJCAR 2020. Lecture Notes in Computer Science, vol 12167, pp 408–415. Springer, Paris, France. https://doi.org/10.1007/978-3-030-51054-1_26

Goertzel ZA, Chvalovský K, Jakubův J, Olšák M, Urban J (2021) Fast and slow Enigmas and parental guidance. In: Konev B, Reger G (eds.) International Symposium on Frontiers of Combining Systems. Lecture Notes in Computer Science, vol. 12941, pp 173–191. Springer, Birmingham, UK. https://doi.org/10.1007/978-3-030-86205-3_10

Goertzel ZA, Jakubův J, Kaliszyk C, Olšák M, Piepenbrock J, Urban J (2022) The Isabelle ENIGMA. In: Andronick J, Moura L (eds.) 13th International Conference on Interactive Theorem Proving (ITP 2022). Leibniz International Proceedings in Informatics (LIPIcs), vol. 237, pp 1–21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Haifa, Israel. https://doi.org/10.4230/LIPIcs.ITP.2022.16

Goertzel Z, Jakubův J, Urban J (2019) ENIGMAWatch: ProofWatch meets ENIGMA. In: Cerrito S, Popescu A (eds.) Automated Reasoning with Analytic Tableaux and Related Methods. Lecture Notes in Computer Science, vol. 11714, pp 374–388. Springer, London, UK. https://doi.org/10.1007/978-3-030-29026-9_21

Goertzel Z, Urban J (2019) Usefulness of lemmas via graph neural networks. In: 4th Conference on artificial intelligence and theorem proving. AITP Conference, Obergurgl, Austria. https://aitp-conference.org/2019/abstract/AITP_2019_paper_32.pdf

Goldberg EI, Prasad MR, Brayton RK (2001) Using SAT for combinational equivalence checking. In: Proceedings Design, Automation and Test in Europe. Conference and Exhibition 2001, pp. 114–121. IEEE, Munich, Germany. https://doi.org/10.1109/DATE.2001.915010

Goodfellow I, Bengio Y, Courville A (2016) Deep Learning. MIT Press, Cambridge, MA. http://www.deeplearningbook.org

Granmo O-C, Bouhmala N (2010) Using learning automata to enhance local-search based SAT solvers with learning capability. In: Zhang Y (ed.) Application of Machine Learning. IntechOpen, London, UK. https://doi.org/10.5772/8610

Gross D, Jansen N, Junges S, Pérez GA (2022) COOL-MC: A comprehensive tool for reinforcement learning and model checking. In: Dong W, Talpin J-P (eds) International symposium on dependable software engineering: Theories, tools, and applications. lecture notes in computer science, vol 13649, pp 41–49. Springer, Beijing, China. https://doi.org/10.1007/978-3-031-21213-0_3

Guo W, Zhen H-L, Li X, Luo W, Yuan M, Jin Y, Yan J (2023) Machine learning methods in solving the boolean satisfiability problem. Mach Intell Res 20:640–655. https://doi.org/10.1007/s11633-022-1396-2

Haltermann J, Wehrheim H (2022) Machine learning based invariant generation: A framework and reproducibility study. In: 2022 IEEE Conference on software testing, verification and validation (ICST), pp 12–23. IEEE, Valencia, Spain. https://doi.org/10.1109/ICST53961.2022.00012

Han JM (2020) Learning cubing heuristics for SAT from DRAT proofs. In: 5th Conference on artificial intelligence and theorem proving. AITP Conference, Aussois, France. https://aitp-conference.org/2020/abstract/paper_23.pdf

Han JM, Rute J, Wu Y, Ayers E, Polu S (2022) Proof artifact co-training for theorem proving with language models. In: International conference on learning representations. OpenReview.net, Virtual Event. https://openreview.net/forum?id=rpxJc9j04U

Han JM, Xu T, Polu S, Neelakantan A, Radford A (2021) Contrastive finetuning of generative language models for informal premise selection. In: 6th Conference on artificial intelligence and theorem proving. AITP Conference, Virtual Event. https://aitp-conference.org/2021/abstract/paper_21.pdf

He J, Bartocci E, Ničković D, Isakovic H, Grosu R (2022) DeepSTL: From english requirements to signal temporal logic. In: Proceedings of the 44th international conference on software engineering. ICSE '22, pp 610–622. Association for Computing Machinery, Pittsburgh, Pennsylvania. https://doi.org/10.1145/3510003.3510171

Hireche C, Drias H (2019) Multidimensional appropriate clustering and DBSCAN for SAT solving. Data Technol Appl 53(1):85–107. https://doi.org/10.1108/dta-07-2018-0068

Hireche C, Drias H, Moulai H (2020) Grid based clustering for satisfiability solving. Appl Soft Comput J 88:106069. https://doi.org/10.1016/j.asoc.2020.106069

Holden EK, Korovin K (2019) SMAC and XGBoost your theorem prover. In: 4th Conference on artificial intelligence and theorem proving, Obergurgl, Austria. https://aitp-conference.org/2019/abstract/paper%2031.pdf

Holden SB (2021) Machine learning for automated theorem proving: Learning to solve SAT and QSAT. Found Trends®Mach Learn 14(6):807–989. https://doi.org/10.1561/2200000081

Hoos HH, Hutter F, Leyton-Brown K (2021) Automated configuration and selection of SAT solvers. In: Biere A, Heule M, Maaren H, Walsh T (eds) Handbook of Satisfiability. IOS Press, Amsterdam, NL, pp 481–507

Hu M, Zhang M, Mallet F, Fu X, Chen M (2022) Accelerating reinforcement learning-based CCSL specification synthesis using curiosity-driven exploration. IEEE Trans Comput 75(5):1431–1446. https://doi.org/10.1109/TC.2022.3197956

Huang D, Dhariwal P, Song D, Sutskever I (2019) GamePad: A learning environment for theorem proving. In: International conference on learning representations. OpenReview.net, New Orleans, LA, USA. https://openreview.net/forum?id=r1xwKoR9Y7

Huang J, Zhen H-L, Wang N, Mao H, Yuan M, Huang Y (2022) Neural fault analysis for SAT-based ATPG. In: 2022 IEEE International test conference (ITC), pp 36–45. IEEE, Anaheim, CA, USA. https://doi.org/10.1109/ITC50671.2022.00010

Hůla J, Mojžíšek D, Janota M (2021) Graph neural networks for scheduling of SMT solvers. In: 2021 IEEE 33rd international conference on tools with artificial intelligence (ICTAI), pp 447–451. IEEE, Washington, DC, USA.https://doi.org/10.1109/ICTAI52525.2021.00072

Hutson M (2018) Artificial intelligence faces reproducibility crisis. Science 359(6377):725–726. https://doi.org/10.1126/science.359.6377.725

Hutter F, Xu L, Hoos HH, Leyton-Brown K (2014) Algorithm runtime prediction: Methods & evaluation. Artif Intell 206:79–111. https://doi.org/10.1016/j.artint.2013.10.003

Hu G, Zhang W, Zhang H (2023) NeuroPDR: Integrating neural networks in the PDR algorithm for hardware model checking. In: 2023 ACM/IEEE 5th workshop on machine learning for CAD (MLCAD). MLCAD, pp 1–6. IEEE, Snowbird, UT, USA. https://doi.org/10.1109/MLCAD58807.2023.10299875

Jaeger M, Larsen KG, Tibo A (2020) From statistical model checking to run-time monitoring using a bayesian network approach. In: Deshmukh J, Ničković D (eds) Runtime verification. Lecture notes in computer science, vol 12399, pp 517–535. Springer, Los Angeles, CA, USA. https://doi.org/10.1007/978-3-030-60508-7_30

Jakubův J, Chvalovský K, Olšák M, Piotrowski B, Suda M, Urban J (2020) ENIGMA anonymous: Symbol-independent inference guiding machine (system description). In: Peltier N, Sofronie-Stokkermans V (eds.) Automated Reasoning. Lecture Notes in Computer Science, vol. 12167, pp 448–463. Springer, Paris, France. https://doi.org/10.1007/978-3-030-51054-1_29

Jakubův J, Janota M, Piotrowski B, Piepenbrock J, Reynolds A (2023) Selecting quantifiers for instantiation in SMT. In: Graham-Lengrand S, Preiner M (eds.) Proceedings of the 21st International Workshop on Satisfiability Modulo Theories (SMT 2023). CEUR Workshop Proceedings, vol. 3439. CEUR-WS.org, Rome, Italy. https://ceur-ws.org/Vol-3429/short10.pdf

Jakubuv J, Urban J (2019) Hammering Mizar by learning clause guidance. In: Harrison J, O'Leary J, Tolmach A (eds.) 10th International Conference on Interactive Theorem Proving (ITP 2019). Leibniz International Proceedings in Informatics (LIPIcs), vol. 141, pp 1–8. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Portland, OR, USA. https://doi.org/10.4230/LIPIcs.ITP.2019.34

Janota M, Piepenbrock J, Piotrowski B (2022) Towards learning quantifier instantiation in SMT. In: Meel KS, Strichman O (eds.) 25th International Conference on Theory and Applications of Satisfiability Testing (SAT 2022). Leibniz International Proceedings in Informatics (LIPIcs), vol. 236, pp 1–18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Haifa, Israel. https://doi.org/10.4230/LIPIcs.SAT.2022.7

Jiang AQ, Li W, Han JM, Wu Y (2021) LISA: Language models of ISAbelle proofs. In: 6th Conference on artificial intelligence and theorem proving. AITP Conference, Assouis and online, France. https://aitp-conference.org/2021/abstract/paper_17.pdf

Jiang AQ, Li W, Tworkowski S, Czechowski K, Odrzygóźdź T, Mił oś P, Wu Y, Jamnik M (2022) Thor: Wielding hammers to integrate language models and automated theorem provers. In: Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K, Oh A (eds.) Advances in Neural Information Processing Systems 35 (NeurIPS 2022), vol. 35, pp 8360–8373. Curran Associates, Inc., New Orleans, LA, USA. https://proceedings.neurips.cc/paper_files/paper/2022/file/377c25312668e48f2e531e2f2c422483-Paper-Conference.pdf

Jiang X, Xie M, Ma L, Dong L, Li D (2023) International publication trends in the application of artificial intelligence in ophthalmology research: an updated bibliometric analysis. Ann Translational Med 11(5). https://doi.org/10.21037/atm-22-3773

Jordan MI (1997) Serial order: A parallel distributed processing approach. In: Donahoe JW, Dorsel VP (eds) Neural-Network Models of Cognition. Advances in Psychology, vol. 121, pp 471–495. North-Holland, Amsterdam. https://doi.org/10.1016/S0166-4115(97)80111-2

Kaelbling LP, Littman ML, Moore AW (1996) Reinforcement learning: A survey. J Artif Intell Res 4:237–285. https://doi.org/10.1613/jair.301

Kecman V (2005) Support Vector Machines – An Introduction. In: Wang, L. (ed.) Support Vector Machines: Theory and Applications, pp 1–47. Springer, Berlin, Heidelberg. https://doi.org/10.1007/10984697_1

Khurana D, Koli A, Khatter K, Singh S (2023) Natural language processing: state of the art, current trends and challenges. Multimed Tools Appl 82:3713–3744. https://doi.org/10.1007/s11042-022-13428-4

Kilani Y, Bsoul M, Alsarhan A, Al-Khasawneh A (2013) A survey of the satisfiability-problems solving algorithms. Int J Adv Intell Paradigms 5(3):233–256. https://doi.org/10.1504/IJAIP.2013.056447

Kitchenham BA, Budgen D, Brereton OP (2010) The value of mapping studies – a participant-observer case study. In: Proceedings of the 14th international conference on evaluation and assessment in software engineering. EASE'10, pp 25–33. BCS Learning & Development Ltd., Swindon, UK. https://doi.org/10.14236/ewic/EASE2010.4

Kitchenham B, Charters S et al (2007) Guidelines for performing systematic literature reviews in software engineering. Technical report, Keele University. https://legacyfileshare.elsevier.com/promis_misc/525444systematicreviewsguide.pdf

Kobayashi N, Sekiyama T, Sato I, Unno H (2021) Toward neural-network-guided program synthesis and verification. In: Drăgoi C, Mukherjee S, Namjoshi K (eds) Static analysis. Lecture notes in computer science, vol 12931, pp 236–260. Springer, Chicago, IL, USA. https://doi.org/10.1007/978-3-030-88806-0_12

Kommrusch S, Monperrus M, Pouchet L-N (2023) Self-supervised learning to prove equivalence between straight-line programs via rewrite rules. IEEE Trans Softw Eng 49(7):3771–3792. https://doi.org/10.1109/TSE.2023.3271065

Koscinski V, Gambardella C, Gerstner E, Zappavigna M, Cassetti J, Mirakhorli M (2021) A natural language processing technique for formalization of systems requirement specifications. In: Yue T, Mirakhorli M (eds) 2021 IEEE 29th international requirements engineering conference workshops (REW). REW, pp 350–356. IEEE, Virtual Event. https://doi.org/10.1109/REW53955.2021.00062

Kumar M, Kolb S, Teso S, De Raedt L (2023) Learning MAX-SAT from contextual examples for combinatorial optimisation. Artif Intell 314:103794. https://doi.org/10.1016/j.artint.2022.103794

Kumazawa T, Takada K, Takimoto M, Kambayashi Y (2019) Ant colony optimization based model checking extended by smell-like pheromone with hop counts. Swarm Evolutionary Comput 44:511–521. https://doi.org/10.1016/j.swevo.2018.06.002

Kumazawa T, Takimoto M, Kambayashi Y (2020) A survey on the applications of swarm intelligence to software verification. In: Tan Y (ed.) Handbook of Research on Fireworks Algorithms and Swarm Intelligence, pp 376–398. IGI Global, Hershey, PA. https://doi.org/10.4018/978-1-7998-1659-1.ch017

Kumazawa T, Takimoto M, Kambayashi Y (2021) Exploration strategies for model checking with ant colony OptimizationQD. In: Nguyen NT, Iliadis L, Maglogiannis I, Trawiński B (eds) Computational collective intelligence, pp 264–276. Springer, Rhodes, Greece. https://doi.org/10.1007/978-3-030-88081-1_20

Kumazawa T, Takimoto M, Kambayashi Y (2022) A safety checking algorithm with multi-swarm particle swarm optimization. In: Fieldsend JE (ed) Proceedings of the genetic and evolutionary computation conference companion. GECCO '22, pp 786–789. Association for Computing Machinery, Boston, MA, USA. https://doi.org/10.1145/3520304.3528918

Kumazawa T, Takimoto M, Kodama Y, Kambayashi Y (2023) Enhancing safety checking coverage with multi-swarm particle swarm optimization. In: Mathieu P, Dignum F, Novais P, Prieta F (eds) Advances in practical applications of agents, multi-agent systems, and cognitive mimetics. The PAAMS Collection. Lecture Notes in Computer Science, vol 13955, pp 137–148. Springer, Guimarães, Portugal. https://doi.org/10.1007/978-3-031-37616-0_12

Kurin V, Godil S, Whiteson S, Catanzaro B (2020) Can Q-learning with graph networks learn a generalizable branching heuristic for a SAT solver? In: Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin H (eds.) Advances in Neural Information Processing Systems, vol. 33, pp 9608–9621. Curran Associates, Inc., Virtual Event. https://proceedings.neurips.cc/paper_files/paper/2020/file/6d70cb65d15211726dcce4c0e971e21c-Paper.pdf

Lample G, Lacroix T, Lachaux M-A, Rodriguez A, Hayat A, Lavril T, Ebner G, Martinet X (2022) HyperTree proof search for neural theorem proving. In: Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K, Oh A (eds) Advances in neural information processing systems 35 (NeurIPS 2022), vol 35, pp 26337–26349. Curran Associates, Inc., New Orleans, LA, USA. https://proceedings.neurips.cc/paper_files/paper/2022/file/a8901c5e85fb8e1823bbf0f755053672-Paper-Conference.pdf

Lassouaoui M, Boughaci D, Benhamou B (2019) A multilevel synergy Thompson sampling hyper-heuristic for solving Max-SAT. Intell Decision Technol 13(2):193–210. https://doi.org/10.3233/idt-180036

Laurent J, Platzer A (2021) Designing a theorem prover for reinforcement learning and neural guidance. In: 6th Conference on artificial intelligence and theorem proving. AITP Conference, Aussois and online, France. https://aitp-conference.org/2021/abstract/paper_8.pdf

Laurent J, Platzer A (2022) Learning to find proofs and theorems by learning to refine search strategies: The case of loop invariant synthesis. In: Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K, Oh A (eds) Advances in neural information processing systems, vol 35, pp 4843–4856. Curran Associates, Inc., New Orleans, LA, USA. https://proceedings.neurips.cc/paper_files/paper/2022/file/1f14ac136d55c34a18a04ce3db083599-Paper-Conference.pdf

LeCun Y, Kavukcuoglu K, Farabet C (2010) Convolutional networks and applications in vision. In: Proceedings of 2010 IEEE International symposium on circuits and systems, pp 253–256. IEEE, Paris, France. https://doi.org/10.1109/ISCAS.2010.5537907

Lederman G, Rabe M, Seshia S, Lee EA (2020) Learning heuristics for quantified Boolean formulas through reinforcement learning. In: International conference on learning representations. OpenReview.net, Addis Ababa, Ethiopia. https://openreview.net/forum?id=BJluxREKDB

Leeson W, Dwyer MB, Filieri A (2023) Sibyl: Improving software engineering tools with SMT selection. In: 2023 IEEE/ACM 45th International conference on software engineering (ICSE), pp 2185–2197. IEEE, Melbourne, Australia. https://doi.org/10.1109/ICSE48619.2023.00184

Leventi-Peetz AM, Peetz J-V, Rohde M (2020) ML supported predictions for SAT solvers performance. In: Arai K, Bhatia R, Kapoor S (eds.) Proceedings of the Future Technologies Conference (FTC) 2019. Advances in Intelligent Systems and Computing, vol. 1069, pp 64–78. Springer, San Francisco, CA, USA. https://doi.org/10.1007/978-3-030-32520-6_7

Li Z, Si X (2022) NSNet: A general neural probabilistic framework for satisfiability problems. In: Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K, Oh A (eds.) Advances in Neural Information Processing Systems 35 (NeurIPS 2022), vol. 35, pp 25573–25585. Curran Associates, Inc., New Orleans, LA, USA. https://proceedings.neurips.cc/paper_files/paper/2022/file/a40462acc6959034c6aa6dfb8e696415-Paper-Conference.pdf

Liu J, Chen Y, Tan B, Dillig I, Feng Y (2023) Learning contract invariants using reinforcement learning. In: Proceedings of the 37th IEEE/ACM international conference on automated software engineering. ASE '22. Association for computing machinery, Rochester, MI, USA. https://doi.org/10.1145/3551349.3556962

Liu Q, Wu Z, Wang Z, Sutcliffe G (2020) Evaluation of axiom selection techniques. In: Boris Konev AS Claudia Schon (ed.) PAAR+SC-Square 2020 Practical Aspects of Automated Reasoning and Satisfiability Checking and Symbolic Computation Workshop 2020. CEUR Workshop Proceedings, vol. 2752, pp 63–75. CEUR-WS.org, Virtual Event. https://ceur-ws.org/Vol-2752/paper5.pdf

Liu Q, Xu Y, He X (2022) Attention recurrent cross-graph neural network for selecting premises. Int J Mach Learn Cybernet 13:1301–1315. https://doi.org/10.1007/s13042-021-01448-9

Lorenz J-H, Nickerl J (2020) The potential of restarts for ProbSAT. In: Moreno-Díaz R, Pichler F, Quesada-Arencibia A (eds.) Computer Aided Systems Theory – EUROCAST 2019. Lecture Notes in Computer Science, vol. 12013, pp 352–360. Springer, Las Palmas de Gran Canaria, Spain. https://doi.org/10.1007/978-3-030-45093-9_43

Lu Y (2019) Artificial intelligence: a survey on evolution, models, applications and future trends. J Manag Anal 6:1–29. https://doi.org/10.1080/23270012.2019.1570365

Luo W, Wan H, Du J, Li X, Fu Y, Ye R, Zhang D (2022) Teaching LTLf satisfiability checking to neural networks. In: Raedt LD (ed) Proceedings of the thirty-first international joint conference on artificial intelligence (IJCAI-22), pp 3292–3298. International Joint Conferences on Artificial Intelligence Organization, Vienna, Austria. https://doi.org/10.24963/ijcai.2022/457

Luo W, Wan H, Zhang D, Du J, Su H (2023) Checking LTL satisfiability via end-to-end learning. In: Proceedings of the 37th IEEE/ACM international conference on automated software engineering. ASE '22. Association for Computing Machinery, Rochester, MI, USA. https://doi.org/10.1145/3551349.3561163

Ma Y, Cao Z, Liu Y (2019) Genetic algorithm-based assume-guarantee reasoning for stochastic model checking. In: Song Y, Acharya S, Nguyen N (eds) IEEE/ACIS 17th International conference on software engineering research, mangement and applications (SERA). SERA, pp 124–127. IEEE, Honolulu, Hawaii, USA. https://doi.org/10.1109/SERA.2019.8886798

Mangla C, Holden SB, Paulson LC (2022) Bayesian ranking for strategy scheduling in automated theorem provers. In: Blanchette J, Kovács L, Pattinson D (eds.) International Joint Conference on Automated Reasoning. Lecture Notes in Computer Science, vol. 13385, pp 559–577. Springer, Haifa, Israel. https://doi.org/10.1007/978-3-031-10769-6_33

Marino R (2021) Learning from survey propagation: a neural network for MAX-E-3-SAT. Mach Learn Sci Technol 2(3):035032. https://doi.org/10.1088/2632-2153/ac0496

Mitchell M (1998) An Introduction to Genetic Algorithms. MIT press, Cambridge, MA. https://direct.mit.edu/books/monograph/4675/An-Introduction-to-Genetic-Algorithms

Mitchell TM (1997) Machine Learning. McGraw-Hill Professional, New York, NY, USA. https://www.cs.cmu.edu/~tom/mlbook.html

Montgomery DC, Peck EA, Vining GG (2021) Introduction to Linear Regression Analysis, 6th edn. Wiley series in probability and statistics. John Wiley & Sons, New Jersey. https://www.wiley-vch.de/en/areas-interest/mathematics-statistics/statistics-16st/regression-analysis-16st4/introduction-to-linear-regression-analysis-978-1-119-57872-7

Morris M, Minervini P, Blunsom P (2022) Learning proof path selection policies in neural theorem proving. In: Garcez EJ-R (ed.) Proceedings of the 16th International Workshop on Neural-Symbolic Learning and Reasoning (NeSy). CEUR Workshop Proceedings, vol. 3212. CEUR-WS.org, Siena, Italy. https://ceur-ws.org/Vol-3212/paper5.pdf

Mo G, Xiong Y, Huang W, Ma L (2020) Automated theorem proving via interacting with proof assistants by dynamic strategies. In: Guerrero J (ed) 2020 6th International conference on big data computing and communications (BIGCOM), pp 71–75. IEEE, Deqing, China. https://doi.org/10.1109/bigcom51056.2020.00016

Nagashima Y (2019) Towards evolutionary theorem proving for isabelle/HOL. In: López-Ibáñez M (ed.) Proceedings of the Genetic and Evolutionary Computation Conference Companion. GECCO '19, pp 419–420. Association for Computing Machinery, Prague, Czech Republic. https://doi.org/10.1145/3319619.3321921

Nagashima Y (2020) Simple dataset for proof method recommendation in Isabelle/HOL. In: Benzmüller C, Miller B (eds) Intelligent computer mathematics. Lecture notes in computer science, vol 12236, pp 297–302. Springer, Bertinoro, Italy. https://doi.org/10.1007/978-3-030-53518-6_21

Nawaz MS, Fournier-Viger P, Zhang J (2020) Proof learning in PVS with utility pattern mining. IEEE Access 8:119806–119818. https://doi.org/10.1109/ACCESS.2020.3004199

Nawaz MS, Nawaz MZ, Hasan O, Fournier-Viger P, Sun M (2021) An evolutionary/heuristic-based proof searching framework for interactive theorem prover. Appl Soft Comput 104:107200. https://doi.org/10.1016/j.asoc.2021.107200

Nawaz MS, Nawaz MZ, Hasan O, Fournier-Viger P, Sun M (2021) Proof searching and prediction in HOL4 with evolutionary/heuristic and deep learning techniques. Appl Intell 51:1580–1601. https://doi.org/10.1007/s10489-020-01837-7

Nawaz MS, Sun M, Fournier-Viger P (2019) Proof guidance in PVS with sequential pattern mining. In: Hojjat H, Massink M (eds) International conference on fundamentals of software engineering. Lecture notes in computer science, vol 11761, pp 45–60. Springer, Tehran, Iran. https://doi.org/10.1007/978-3-030-31517-7_4

Nayak A, Timmapathini HP, Murali V, Ponnalagu K, Venkoparao VG, Post A (2022) Req2Spec: Transforming software requirements into formal specifications using natural language processing. In: Gervasi A Vincenzoand Vogelsang (ed) Requirements Engineering: Foundation for Software Quality. Lecture Notes in Computer Science, vol 13216, pp 87–95. Springer, Birmingham, UK. https://doi.org/10.1007/978-3-030-98464-9_8

Nejati S, Le Frioux L, Ganesh V (2020) A machine learning based splitting heuristic for divide-and-conquer solvers. In: Simonis H (ed.) Principles and Practice of Constraint Programming. Lecture Notes in Computer Science, vol. 12333, pp 899–916. Springer, Louvain-la-Neuve, Belgium. https://doi.org/10.1007/978-3-030-58475-7_52

Nie P, Palmskog K, Li JJ, Gligoric M (2020) Deep generation of Coq lemma names using elaborated terms. In: Peltier N, Sofronie-Stokkermans V (eds) International joint conference on automated reasoning. Lecture notes in computer science, vol 12167, pp 97–118. Springer, Paris, France. https://doi.org/10.1007/978-3-030-51054-1_6

Nikolić M, Marinković V, Kovács Z, Janičić P (2019) Portfolio theorem proving and prover runtime prediction for geometry. Ann Math Artif Intell 85:119–146. https://doi.org/10.1007/s10472-018-9598-6

Nurcahyadi T, Blum C, Manyà F (2022) Negative learning ant colony optimization for MaxSAT. Int J Computat Intell Syst 15(1). https://doi.org/10.1007/s44196-022-00120-6

Olsák M, Kaliszyk C, Urban J (2020) Property invariant embedding for automated reasoning. In: De Giacomo G, Catala A, Dilkina B, Milano M, Barro S, Bugarín A, Lang J (eds.) 24th European Conference on Artificial Intelligence. Fronties in Artificial Intelligence and Applications, vol. 325, pp 1395–1402. Santiago de Compostela, Spain. https://doi.org/10.3233/FAIA200244. https://ebooks.iospress.nl/publication/55039

Ozolins E, Freivalds K, Draguns A, Gaile E, Zakovskis R, Kozlovics S (2022) Goal-aware neural SAT solver. In: 2022 International Joint Conference on Neural Networks (IJCNN), pp 1–8. IEEE, Padua, Italy. https://doi.org/10.1109/IJCNN55064.2022.9892733

Palermo J, Ye J, Han JM (2022) Synthetic proof term data augmentation for theorem proving with language models. In: 7th Conference on artificial intelligence and theorem proving. AITP Conference, Aussois, France. https://aitp-conference.org/2022/abstract/AITP_2022_paper_5.pdf

Paliwal A, Loos S, Rabe M, Bansal K, Szegedy C (2020) Graph representations for higher-order logic and theorem proving. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 2967–2974. AAAI Press, New York, NY, USA. https://doi.org/10.1609/aaai.v34i03.5689

Pan Z, Mishra P (2022) A survey on hardware vulnerability analysis using machine learning. IEEE Access 10:49508–49527. https://doi.org/10.1109/ACCESS.2022.3173287

Petersen K, Vakkalanka S, Kuzniarz L (2015) Guidelines for conducting systematic mapping studies in software engineering: An update. Inf Softw Technol 64:1–18. https://doi.org/10.1016/j.infsof.2015.03.007

Piepenbrock J, Heskes T, Janota M, Urban J (2021) Learning equational theorem proving. In: 6th Conference on artificial intelligence and theorem proving. AITP Conference, Aussois, France. https://doi.org/10.48550/ARXIV.2102.05547. https://aitpconference.org/2021/abstract/paper_20.pdf

Piepenbrock J, Heskes T, Janota M, Urban J (2022) Guiding an automated theorem prover with neural rewriting. In: Blanchette J, Kovács L, Pattinson D (eds) International joint conference on automated reasoning. Lecture notes in computer science, vol 13385, pp 597–617. Springer, Haifa, Israel. https://doi.org/10.1007/978-3-031-10769-6

Pimpalkhare N (2021) Dynamic algorithm selection for SMT. In: Proceedings of the 35th IEEE/ACM international conference on automated software engineering. ASE '20, pp 1376–1378. Association for Computing Machinery, Virtual Event.https://doi.org/10.1145/3324884.3418922

Pimpalkhare N, Mora F, Polgreen E, Seshia SA (2021) MedleySolver: Online SMT algorithm selection. In: Chu-Min Li FM (ed.) Theory and Applications of Satisfiability Testing — SAT 2021. Lecture Notes in Computer Science, vol. 12831, pp 453–470. Springer, Barcelona, Spain. https://doi.org/10.1007/978-3-030-80223-3_31

Piotrowski B, Urban J (2020) Guiding inferences in connection tableau by recurrent neural networks. In: Benzmüller C, Miller B (eds) Intelligent computer mathematics. lecture notes in computer science, vol 12236, pp 309–314. Springer, Bertinoro, Italy. https://doi.org/10.1007/978-3-030-53518-6_23

Piotrowski B, Urban J (2020) Stateful premise selection by recurrent neural networks. In: Albert E, Kovacs L (eds.) 23rd International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR-23). EPiC Series in Computing, vol. 73, pp 409–422. EasyChair, Virtual Event. https://doi.org/10.29007/j5hd

Pira E (2022) Using knowledge discovery to propose a two-phase model checking for safety analysis of graph transformations. Softw Qual J 30(1):37–64. https://doi.org/10.1007/s11219-020-09542-x

Poesia G, Goodman ND (2023) Peano: learning formal mathematical reasoning. Philosophical Trans Royal Soc Math Phys Eng Sci 381(2251):20220044. https://doi.org/10.1098/rsta.2022.0044

Poesia G, Dong W, Goodman N (2021) Contrastive reinforcement learning of symbolic reasoning domains. In: Ranzato M, Beygelzimer A, Dauphin Y, Liang PS, Vaughan JW (eds) Advances in neural information processing systems, vol 34, pp 15946–15956. Curran Associates, Inc., Virtual Event. https://proceedings.neurips.cc/paper_files/paper/2021/file/859555c74e9afd45ab771c615c1e49a6-Paper.pdf

Popescu A, Polat-Erdeniz S, Felfernig A, Uta M, Atas M, Le V-M, Pilsl K, Enzelsberger M, Tran TNT (2022) An overview of machine learning techniques in constraint solving. J Intell Inf Syst 58:91–118. https://doi.org/10.1007/s10844-021-00666-5

Puccetti A, De Chalendar G, Gibello P-Y (2021) Combining formal and machine learning techniques for the generation of JML specifications. In: Cok DR (ed) FTfJP '21: Proceedings of the 23rd ACM international workshop on formal techniques for java-like programs. FTfJP, pp 59–64. Association for Computing Machinery, Virtual Event. https://doi.org/10.1145/3464971.3468425

Purgał S, Parsert J, Kaliszyk C (2021) A study of continuous vector representations for theorem proving. J Log Comput 31(8):2057–2083. https://doi.org/10.1093/logcom/exab006

Qian H (2021) Research on automation strategy of Coq. In: Sun X, Zhang X, Xia Z, Bertino E (eds) Advances in artificial intelligence and security. Communications in computer and information science, vol 1423, pp 656–665. Springer, Dublin, Ireland. https://doi.org/10.1007/978-3-030-78618-2_54

Rafe V, Darghayedi M, Pira E (2019) MS-ACO: a multi-stage ant colony optimization to refute complex software systems specified through graph transformation. Soft Comput 23:4531–4556. https://doi.org/10.1007/s00500-018-3444-y

Rawson M, Reger G (2019) A neurally-guided, parallel theorem prover. In: Herzig A, Popescu A (eds) Frontiers of combining systems: 12th international symposium, FroCoS 2019. Lecture notes in computer science, vol 11715, pp 40–56. Springer, London, UK. https://doi.org/10.1007/978-3-030-29007-8_3

Rawson M, Reger G (2021) lazyCoP: Lazy paramodulation meets neurally guided search. In: Das A, Negri S (eds) Automated reasoning with analytic tableaux and related methods. Lecture notes in computer science, vol 12842, pp 187–199. Springer, Birmingham, UK. https://doi.org/10.1007/978-3-030-86059-2_11

Reichel T, Henderson RW, Touchet A, Gardner A, Ringer T (2023) Proof repair infrastructure for supervised models: Building a large proof repair dataset. In: Naumowicz A, Thiemann R (eds) 14th International conference on interactive theorem proving (ITP 2023). Leibniz International Proceedings in Informatics (LIPIcs), vol 268, pp 1–20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Bialystok, Poland. https://doi.org/10.4230/LIPIcs.ITP.2023.26

Richter C, Hüllermeier E, Jakobs M-C, Wehrheim H (2020) Algorithm selection for software validation based on graph kernels. Autom Softw Eng 27(1):153–186. https://doi.org/10.1007/s10515-020-00270-x

Richter C, Wehrheim H (2020) Attend and represent: a novel view on algorithm selection for software verification. In: Proceedings of the 35th IEEE/ACM international conference on automated software engineering. Association for computing machinery, virtual event. https://doi.org/10.1145/3324884.3416633

Rosenblatt F (1962) Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington, DC

Sadeg S, Hamdad L, Kada O, Benatchba K, Habbas Z (2021) Meta-learning to select the best metaheuristic for the MaxSAT problem. In: Chikhi S, Amine A, Chaoui A, Saidouni DE, Kholladi MK (eds.) International Symposium on Modelling and Implementation of Complex Systems. Lecture Notes in Networks and Systems, vol. 156, pp 122–135. Springer, Batna, Algeria. https://doi.org/10.1007/978-3-030-58861-8_9

Sadreddin A, Mouhoub M, Sadaoui S (2022) Portfolio selection for SAT instances. In: 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp 2962–2967. IEEE, Prague, Czech Republic. https://doi.org/10.1109/SMC53654.2022.9945151

Sanchez-Stern A, Alhessi Y, Saul L, Lerner S (2020) Generating correctness proofs with neural networks. In: Sen K, Naik M (eds) Proceedings of the 4th ACM SIGPLAN international workshop on machine learning and programming languages. MAPL '20. Association for computing machinery, New York, NY, USA. https://doi.org/10.1145/3394450.3397466

Sanchez-Stern A, First E, Zhou T, Kaufman Z, Brun Y, Ringer T (2023) Passport: Improving automated formal verification using identifiers. ACM Trans Program Languages Systems 45(2). https://doi.org/10.1145/3593374

Sarker IH (2021) Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. SN Comput Sci 2. https://doi.org/10.1007/s42979-021-00815-1

Scott J, Niemetz A, Preiner M, Nejati S, Ganesh V (2021) MachSMT: A machine learning-based algorithm selector for SMT solvers. In: Groote JF, Larsen KG (eds.) International Conference on Tools and Algorithms for the Construction and Analysis of Systems. Lecture Notes in Computer Science, vol. 12652, pp 303–325. Springer, Luxembourg City, Luxembourg. https://doi.org/10.1007/978-3-030-72013-1_16

Scott J, Niemetz A, Preiner M, Nejati S, Ganesh V (2023) Algorithm selection for SMT. Int J Softw Tools Technol Transfer 25:219–239. https://doi.org/10.1007/s10009-023-00696-0

Scott J, Sudula T, Rehman H, Mora F, Ganesh V (2021) BanditFuzz: Fuzzing SMT solvers with multi-agent reinforcement learning. In: Huisman M, Păsăreanu C, Zhan N (eds.) International Symposium on Formal Methods. Lecture Notes in Computer Science, vol. 13047, pp 103–121. Springer, Virtual Event. https://doi.org/10.1007/978-3-030-90870-6_6

Sejnowski TJ (2018) The Deep Learning Revolution. MIT press, Cambridge, MA. https://mitpress.mit.edu/9780262038034/the-deep-learning-revolution/

Selsam D, Bjørner N (2019) Guiding high-performance SAT solvers with Unsat-Core predictions. In: Janota M, Lynce I (eds.) Theory and Applications of Satisfiability Testing – SAT 2019. Lecture Notes in Computer Science, vol. 11628, pp 336–353. Springer, Lisbon, Portugal. https://doi.org/10.1007/978-3-030-24258-9_24

Selsam D, Lamm M, Bünz B, Liang P, Moura L, Dill DL (2019) Learning a SAT solver from single-bit supervision. In: International Conference on Learning Representations. OpenReview.net, New Orleans, LA, USA. https://openreview.net/forum?id=HJMC_iA5tm

Shafiq S, Mashkoor A, Mayr-Dorn C, Egyed A (2021) A literature review of using machine learning in software development life cycle stages. IEEE Access 9:140896–140920. https://doi.org/10.1109/ACCESS.2021.3119746

Shigyo Y, Katayama T (2020) Proposal of an approach to generate VDM++ specifications from natural language specification by machine learning. In: IEEE Global Conference on consumer electronics (GCCE), pp 292–296. IEEE, Kobe, Japan. https://doi.org/10.1109/GCCE50665.2020.9292047

Shminke B (2022) gym-saturation: an OpenAI Gym environment for saturation provers. J Open Source Softw 7(71):3849. https://doi.org/10.21105/joss.03849

Shtrichman O (2000) Tuning SAT checkers for bounded model checking. In: Emerson EA, Sistla AP (eds.) Computer Aided Verification, pp. 480–494. Springer, Berlin, Heidelberg. https://doi.org/10.1007/10722167_36

Simmons AB, Chappell SG (1988) Artificial intelligence-definition and practice. IEEE J Oceanic Eng 13(2):14–42. https://doi.org/10.1109/48.551

Si X, Naik A, Dai H, Naik M, Song L (2020) Code2Inv: A deep learning framework for program verification. In: Lahiri SK, Wang C (eds) International conference on computer aided verification. Lecture notes in computer science, vol 12225, pp 151–164. Springer, Los Angeles, CA, USA. https://doi.org/10.1007/978-3-030-53291-8_9

Suda M (2021) Improving ENIGMA-style clause selection while learning from history. In: Platzer A, Sutcliffe G (eds.) Automated Deduction – CADE 28. Lecture Notes in Computer Science, vol. 12699, pp 543–561. Springer, Virtual Event. https://doi.org/10.1007/978-3-030-79876-5_31

Suda M (2021) Vampire with a brain is a good ITP hammer. In: Konev G Borisand Reger (ed) Frontiers of combining systems. Lecture notes in computer science, vol 12941, pp 192–209. Springer, Birmingham, UK. https://doi.org/10.1007/978-3-030-86205-3_11

Sun L, Gerault D, Benamira A, Peyrin T (2020) NeuroGIFT: Using a machine learning based SAT solver for cryptanalysis. In: Dolev S, Kolesnikov V, Lodha S, Weiss G (eds.) International Symposium on Cyber Security Cryptography and Machine Learning. Lecture Notes in Computer Science, vol. 12161, pp 62–84. Springer, Be'er Sheva, Israel. https://doi.org/10.1007/978-3-030-49785-9_5

Sutton RS, Barto AG (2018) Reinforcement Learning: An Introduction, 2nd edn. MIT Press, Cambridge, MA. http://incompleteideas.net/book/the-book-2nd.html

Szegedy C (2020) A promising path towards autoformalization and general artificial intelligence. In: Benzmüller C, Miller B (eds.) Intelligent Computer Mathematics. Lecture Notes in Computer Science, vol. 12236, pp 3–20. Springer, Bertinoro, Italy. https://doi.org/10.1007/978-3-030-53518-6_1

Tipping ME (2004) Bayesian Inference: An Introduction to Principles and Practice in Machine Learning. In: Bousquet O, Luxburg U, Rätsch G (eds) Advanced Lectures on Machine Learning, pp 41–62. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-28650-9_3

Tran THH, Martinc M, Doucet A, Pollak S (2022) IJS at TextGraphs-16 natural language premise selection task: Will contextual information improve natural language premise selection? In: Ustalov D, Gao Y, Panchenko A, Valentino M, Thayaparan M, Nguyen TH, Penn G, Ramesh A, Jana A (eds.) Proceedings of TextGraphs-16: Graph-based Methods for Natural Language Processing, pp 114–118. Association for Computational Linguistics, Gyeongju, Republic of Korea. https://aclanthology.org/2022.textgraphs-1.12

Tsutomu Kumazawa MT, Kambayashi Y (2022) Exploration strategies for balancing efficiency and comprehensibility in model checking with ant colony optimization. J Inf Telecommun 6(3):341–359. https://doi.org/10.1080/24751839.2022.2047470

Tworkowski S, Mikula M, Odrzygóźdź T, Czechowski K, Antoniak S, Jiang AQ, Szegedy C, Kuciński L, Milós P, Wu Y (2022) Formal premise selection with language models. In: 7th Conference on artificial intelligence and theorem proving. AITP Conference, Aussois, France. https://aitp-conference.org/2022/abstract/AITP_2022_paper_32.pdf

Urban J (2003) MPTP 0.1 - system description. Electron Notes Theoretical Comput Sci 86:147–152. https://doi.org/10.1016/S1571-0661(04)80659-5

Urban J, Jakubův J (2020) First neural conjecturing datasets and experiments. In: Benzmüller C, Miller B (eds) Intelligent computer mathematics. lecture notes in computer science, vol 12236, pp 315–323. Springer, Bertinoro, Italy. https://doi.org/10.1007/978-3-030-53518-6_24

Urban J, Vyskočil J (2013) Theorem Proving in Large Formal Mathematics as an Emerging AI Field. In: Bonacina MP, Stickel ME (eds.) Automated Reasoning and Mathematics. Lecture Notes in Computer Science, vol. 7788, pp 240–257. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-36675-8_13

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Lu, Polosukhin I (2017) Attention is all you need. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R (eds) Advances in Neural Information Processing Systems, vol. 30, pp. 5998–6008. Curran Associates, Inc., Long Beach, CA, USA. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Wang F, Cao Z, Tan L, Zong H (2020) Survey on learning-based formal methods: Taxonomy, applications and possible future directions. IEEE Access 8:108561–108578. https://doi.org/10.1109/ACCESS.2020.3000907

Wang Q, Cao W, Jiang J, Zhao Y, Ming Z (2021) NNRW-based algorithm selection for software model checking. In: Cao J, Vong CM, Miche Y, Lendasse A (eds) Proceedings of ELM2019. Proceedings in adaptation, learning and optimization, vol 14, pp 11–21. Springer, Yangzhou, China. https://doi.org/10.1007/978-3-030-58989-9_2

Wang M, Deng J (2020) Learning to prove theorems by learning to generate theorems. In: Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin H (eds) Advances in neural information processing systems, vol 33, pp 18146–18157. Curran Associates, Inc., Virtual Event. https://proceedings.neurips.cc/paper_files/paper/2020/file/d2a27e83d429f0dcae6b937cf440aeb1-Paper.pdf

Wang Q, Jiang J, Zhao Y, Cao W, Wang C, Li S (2021) Algorithm selection for software verification based on adversarial LSTM. In: 2021 7th IEEE Intl conference on big data security on cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS). IEEE, New York, NY, USA. https://doi.org/10.1109/bigdatasecurityhpscids52275.2021.00026

Wang Y, Roohi N, West M, Viswanathan M, Dullerud GE (2020) Statistically model checking PCTL specifications on markov decision processes via reinforcement learning. In: 2020 59th IEEE conference on decision and control (CDC), pp 1392–1397. IEEE, Jeju, South Korea. https://doi.org/10.1109/CDC42340.2020.9303982

Wang J, Wang C (2023) Learning to synthesize relational invariants. In: Proceedings of the 37th IEEE/ACM international conference on automated software engineering. ASE '22. Association for computing machinery, Rochester, MI, USA. https://doi.org/10.1145/3551349.3556942

Wang W, Wang K, Zhang M, Khurshid S (2019) Learning to optimize the Alloy Analyzer. In: 2019 12th IEEE Conference on software testing, validation and verification (ICST), pp 228–239. IEEE, Xi'an, China. https://doi.org/10.1109/ICST.2019.00031

Wang H, Yuan Y, Liu Z, Shen J, Yin Y, Xiong J, Xie E, Shi H, Li Y, Li L, Yin J, Li Z, Liang X (2023) DT-solver: Automated theorem proving with dynamic-tree sampling guided by proof-level value function. In: Proceedings of the 61st Annual meeting of the association for computational linguistics, vol 1, pp 12632–12646. Association for Computational Linguistics, Toronto, Canada. https://doi.org/10.18653/v1/2023.acl-long.706

Wei Y (2022) Applying autoencoder to automated theorem proving. In: 4th International conference on frontiers technology of information and computer (ICFTIC). IEEE, Qingdao, China. https://doi.org/10.1109/icftic57696.2022.10075262

Welleck S, Liu J, Lu X, Hajishirzi H, Choi Y (2022) NaturalProver: Grounded mathematical proof generation with language models. In: Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K, Oh A (eds.) Advances in Neural Information Processing Systems 35 (NeurIPS 2022), vol. 35, pp 4913–4927. Curran Associates, Inc., New Orleans, LA, USA. https://proceedings.neurips.cc/paper_files/paper/2022/file/1fc548a8243ad06616eee731e0572927-Paper-Conference.pdf

Wohlin C (2014) Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the 18th international conference on evaluation and assessment in software engineering. EASE '14. Association for Computing Machinery, London, UK. https://doi.org/10.1145/2601248.2601268

Woodcock J, Larsen PG, Bicarregui J, Fitzgerald J (2009) Formal methods: Practice and experience. ACM Comput Surv 41:1–36. https://doi.org/10.1145/1592434.1592436

Wu T, He S, Liu J, Sun S, Liu K, Han Q-L, Tang Y (2023) A brief overview of ChatGPT: The history, status quo and potential future development. IEEE/CAA J Autom Sinica 10(5):1122–1136. https://doi.org/10.1109/JAS.2023.123618

Wu Y, Jiang AQ, Li W, Rabe M, Staats C, Jamnik M, Szegedy C (2022a) Autoformalization with large language models. In: Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K, Oh A (eds) Advances in neural information processing systems 35 (NeurIPS 2022), vol 35, pp 32353–32368. Curran Associates, Inc., New Orleans, LA, USA. https://proceedings.neurips.cc/paper_files/paper/2022/file/d0c6bc641a56bebee9d985b937307367-Paper-Conference.pdf

Wu Y, Jiang A, Grosse R, Ba J (2020) Neural theorem proving on inequality problems. In: 5th Conference on artificial intelligence and theorem proving. AITP Conference, Obergurgl, Austria. https://aitp-conference.org/2020/abstract/paper_18.pdf

Wu Y, Jiang A, Li W, Rabe MN, Staats C, Jamnik M, Szegedy C (2022b) Autoformalization for neural theorem proving. In: 7th Conference on artificial intelligence and theorem proving. AITP Conference, Aussois, France. https://aitp-conference.org/2022/abstract/AITP_2022_paper_33.pdf

Wu M, Norrish M, Walder C, Dezfouli A (2020) Reinforcement learning for interactive theorem proving in HOL4. In: 5th Conference on artificial intelligence and theorem proving. AITP Conference, Aussois, France. https://aitp-conference.org/2020/abstract/paper_7.pdf

Wu M, Norrish M, Walder C, Dezfouli A (2021) TacticZero: Learning to prove theorems from scratch with deep reinforcement learning. In: Ranzato M, Beygelzimer A, Dauphin Y, Liang PS, Vaughan JW (eds) Advances in neural information processing systems, vol 34, pp 9330–9342. Curran Associates, Inc., Virtual Event. https://proceedings.neurips.cc/paper_files/paper/2021/file/4dea382d82666332fb564f2e711cbc71-Paper.pdf

Xu R, Chen J, He F (2022) Data-driven loop bound learning for termination analysis. In: Proceedings of the 44th international conference on software engineering. ICSE '22, pp 499–510. Association for computing machinery, Pittsburgh, PA, USA. https://doi.org/10.1145/3510003.3510220

Xu L, Hoos HHH, Leyton-Brown K (2021) Predicting satisfiability at the phase transition. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 26, pp 584–590. AAAI Press, Toronto, Canada. https://doi.org/10.1609/aaai.v26i1.8142

Xu R, Lieberherr K (2022) Towards tackling QSAT problems with deep learning and monte carlo tree search. In: Arai, K. (ed.) Intelligent Computing. Lecture Notes in Networks and Systems, vol. 507, pp 45–58. Springer, London, UK.https://doi.org/10.1007/978-3-031-10464-0_4

Yang K, Deng J (2019) Learning to prove theorems via interacting with proof assistants. In: Chaudhuri K, Salakhutdinov R (eds) Proceedings of the 36th international conference on machine learning. Proceedings of machine learning research, vol 97, pp 6984–6994. PMLR, Long Beach, CA, USA. https://proceedings.mlr.press/v97/yang19a.html

Yan Z, Li M, Shi Z, Zhang W, Chen Y-C, Zhang H (2023) The elephant in the room: Variable dependency in GNN-based SAT solving. In: First International Workshop on Deep Learning-aided Verification. OpenReview.net, Paris, France. https://openreview.net/forum?id=yOyxNRK5MOM

Yao P, Huang H, Tang W, Shi Q, Wu R, Zhang C (2021) Fuzzing SMT solvers via two-dimensional input space exploration. In: Proceedings of the 30th ACM SIGSOFT international symposium on software testing and analysis. ISSTA 2021, pp 322–335. Association for Computing Machinery, Virtual Event. https://doi.org/10.1145/3460319.3464803

Yao J, Ryan G, Wong J, Jana S, Gu R (2020) Learning nonlinear loop invariants with gated continuous logic networks. In: Donaldson AF, Torlak E (eds) Proceedings of the 41st ACM SIGPLAN conference on programming language design and implementation. PLDI 2020, pp 106–120. Association for Computing Machinery, London, UK. https://doi.org/10.1145/3385412.3385986

Yolcu E, Poczos B (2019) Learning local search heuristics for boolean satisfiability. In: Wallach H, Larochelle H, Beygelzimer A, Alché-Buc F, Fox E, Garnett R (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc., Vancouver, Canada. https://proceedings.neurips.cc/paper_files/paper/2019/file/12e59a33dea1bf0630f46edfe13d6ea2-Paper.pdf

You J, Wu H, Barrett C, Ramanujan R, Leskovec J (2019) G2SAT: Learning to generate SAT formulas. In: Wallach H, Larochelle H, Beygelzimer A, Alché-Buc F, Fox E, Garnett R (eds.) Advances in Neural Information Processing Systems 32 (NeurIPS 2019), vol. 32. Curran Associates, Inc., Vancouver, Canada. https://proceedings.neurips.cc/paper_files/paper/2019/file/daea32adcae6abcb548134fa98f139f9-Paper.pdf

Yu S, Wang T, Wang J (2023) Loop invariant inference through SMT solving enhanced reinforcement learning. In: Just R, Fraser G (eds) Proceedings of the 32nd ACM SIGSOFT international symposium on software testing and analysis. ISSTA 2023, pp 175–187. Association for Computing Machinery, Seattle, WA, USA. https://doi.org/10.1145/3597926.3598047

Zeng Z, Talupuru KR, Ciesielski M (2005) Functional test generation based on word-level SAT. J Syst Architect 51(8):488–511. https://doi.org/10.1016/j.sysarc.2004.10.006

Zhang L, Blaauwbroek L, Kaliszyk C, Urban J (2023) Learning proof transformations and its applications in interactive theorem proving. In: Sattler U, Suda M (eds) Frontiers of combining systems: 14th international symposium. lecture notes in computer science, vol 14279, pp 236–254. Springer, Prague, Czech Republic. https://doi.org/10.1007/978-3-031-43369-6_13

Zhang L, Blaauwbroek L, Piotrowski B, Černỳ P, Kaliszyk C, Urban J (2021) Online machine learning techniques for Coq: A comparison. In: Kamareddine F, Sacerdoti Coen C (eds) Intelligent computer mathematics. Lecture notes in computer science, vol 12833, pp 67–83. Springer, Timisoara, Romania. https://doi.org/10.1007/978-3-030-81097-9_5

Zhang X, Li Y, Hong W, Sun M (2019) Using recurrent neural network to predict tactics for proving component connector properties in Coq. In: 2019 International symposium on theoretical aspects of software engineering (TASE), pp 107–112. IEEE, Guilin, China. https://doi.org/10.1109/TASE.2019.00-12

Zhang W, Sun Z, Zhu Q, Li G, Cai S, Xiong Y, Zhang L (2020) NLocalSAT: boosting local search with solution prediction. In: Bessiere C (ed.) Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, pp 1177–1183. International Joint Conferences on Artificial Intelligence Organization, Yokohama, Japan. https://doi.org/10.24963/ijcai.2020/164

Zhang C, Zhang Y, Mao J, Chen W, Yue L, Bai G, Xu M (2022) Towards better generalization for neural network-based SAT solvers. In: Gama J, Li T, Yu Y, Chen E, Zheng Y, Teng F (eds.) Advances in Knowledge Discovery and Data Mining. Lecture Notes in Artificial Intelligence, vol. 13281, pp 199–210. Springer, Chengdu, China. https://doi.org/10.1007/978-3-031-05936-0_16

Zheng J, He K, Zhou J, Jin Y, Li C-M, Manyà F (2022) BandMaxSAT: A local search MaxSAT solver with multi-armed bandit. In: Raedt LD (ed.) Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22), pp. 1901–1907. International Joint Conferences on Artificial Intelligence Organization, Vienna, Austria. https://doi.org/10.24963/ijcai.2022/264

Zhou J, Cui G, Hu S, Zhang Z, Yang C, Liu Z, Wang L, Li C, Sun M (2020) Graph neural networks: A review of methods and applications. AI Open 1:57–81. https://doi.org/10.1016/j.aiopen.2021.01.001

Zhu W (2022) Approximate model checking based on deep forest. In: 2022 International conference on artificial intelligence and computer information technology (AICIT). AICIT, pp 1–4. IEEE, Yichang, China. https://doi.org/10.1109/AICIT55386.2022.9930208

Zhu W, Wu H (2022) CTL model checking based on binary classification of machine learning. Int Arab J Inf Technol 19(2):249–260. https://doi.org/10.34028/iajit/19/2/12

Zhu W, Wu H, Deng M (2019) LTL model checking based on binary classification of machine learning. IEEE Access 7:135703–135719. https://doi.org/10.1109/ACCESS.2019.2942762

Zombori Z, Csiszárik A, Michalewski H, Kaliszyk C, Urban J (2019) Curriculum learning and theorem proving. In: 4th Conference on artificial intelligence and theorem proving. AITP Conference, Obergurgl, Austria. https://aitp-conference.org/2019/abstract/paper%2013.pdf

Zombori Z, Csiszárik A, Michalewski H, Kaliszyk C, Urban J (2021) Towards finding longer proofs. In: Das A, Negri S (eds) Automated reasoning with analytic tableaux and related methods. Lecture notes in computer science, vol 12842, pp 167–186. Springer, Birmingham, UK. https://doi.org/10.1007/978-3-030-86059-2_10

Zombori Z, Urban J, Brown CE (2020) Prolog technology reinforcement learning prover. In: Peltier N, Sofronie-Stokkermans V (eds) Automated Reasoning. Lecture notes in computer science, vol 12167, pp 489–507. Springer, Paris, France. https://doi.org/10.1007/978-3-030-51054-1_33

Zombori Z, Urban J, Olšák M (2021) The role of entropy in guiding a connection prover. In: Das A, Negri S (eds) International conference on automated reasoning with analytic tableaux and related methods. lecture notes in computer science, vol 12842, pp 218–235. Springer, Birmingham, UK. https://doi.org/10.1007/978-3-030-86059-2_13

Authors and Affiliations

Sebastian Stock[1] · Jannik Dunkelau[2] · Atif Mashkoor[1]

✉ Jannik Dunkelau
jannik.dunkelau@hhu.de

Sebastian Stock
sebastian.stock@jku.at

Atif Mashkoor
atif.mashkoor@partner.jku.at

[1] Institute of Software System Engineering, Johannes Kepler University Linz, Altenbergerstraße 69, Linz 4040, Austria

[2] Heinrich Heine University Düsseldorf, Faculty of Mathematics and Natural Sciences, Universitätsstraße 1, Düsseldorf 40225, Germany