Towards a Common Framework for Autoformalization

Agnieszka Mensfelt[1], David Tena Cucala[1], Santiago Franco[1], Angeliki Koutsoukou-Argyraki[1,2], Vince Trencsenyi[1], Kostas Stathis[1]

[1]Department of Computer Science, Royal Holloway, University of London,
Egham Hill, Egham, TW200EX, UK
[2]Department of Computer Science and Technology, University of Cambridge,
William Gates Building, 15 JJ Thomson Avenue, Cambridge, CB3 0FD, UK
{agnieszka.mensfelt, david.tenacucala, santiago.francoaixela, angeliki.koutsoukouargyraki, vince.trencsenyi,
kostas.stathis}@rhul.ac.uk

Abstract

Autoformalization has emerged as a term referring to the automation of formalization in the context of the formalization of mathematics using interactive theorem provers (proof assistants). Its rapid development has been driven by progress in deep learning, especially large language models (LLMs). More recently, usage of the term has expanded beyond mathematics to describe tasks that involve translating natural language input into verifiable logical representations. At the same time, a growing body of research explores using LLMs to translate informal language into formal representations for reasoning, planning, and knowledge representation, but without explicitly referring to this process as autoformalization. As a result, despite addressing similar tasks, the largely independent development of these research areas has limited opportunities for shared methodologies, benchmarks, and theoretical frameworks that could accelerate progress. Our goal is to review—explicit or implicit—instances of what can be considered autoformalization and to propose a unified framework, encouraging cross-pollination between different fields to advance the development of next generation AI systems.

Extended version — https://arxiv.org/abs/2509.09810

Introduction

The effort to formalize reasoning—motivated by the desire to eliminate ambiguity and make it precise and systematic—has ancient roots, tracing back to Aristotle's syllogistic logic. With the advent of computers, this problem has taken on new significance as natural language must be bridged with the formal representations that computers can process. The challenge was taken up at the start of the field AI, motivating some of its foundational work in knowledge representation and language understanding. AI pioneers (McCarthy 1960; Kowalski 1974; Newell and Simon 1976) designed programming languages for formalizing knowledge using ideas from formal logic (e.g. LISP, Prolog). Simultaneously, early researchers in natural language processing (Weizenbaum 1966) and language understanding (Winograd 1972) used formal logic and inference systems to model human language (Warren and Pereira 1982). These advances allowed computers to manipulate formal representations of language, answer structured questions, and support human-computer interaction, partially bridging the gap between formal systems and natural language. Furthermore, these efforts paved the way for *semantic parsing* (Kamath and Das 2018), a field dedicated to translating natural language into formal meaning representations.

Recently, a closely related and rapidly growing trend has gained prominence: *autoformalization*. The term originated in the context of the automatic formalization of mathematics within interactive theorem provers (proof assistants) such as Isabelle/HOL, Lean, Coq, and Mizar (Wang, Kaliszyk, and Urban 2018; Szegedy 2020; Wu et al. 2022; Jiang, Li, and Jamnik 2024) following pioneering work that framed the task as automated translation between informal and formal mathematics (Kaliszyk et al. 2014; Kaliszyk, Urban, and Vyskočil 2015, 2017). More recently, the scope of autoformalization has broadened beyond mathematics, e.g., to translation into formal languages for reasoning tasks (Pan et al. 2023). Conceptually, autoformalization can be viewed as a special case of semantic parsing—one that targets formal languages with precise semantics and proof capabilities, in contrast to the broader scope of semantic parsing, which encompasses diverse programming and query languages. Today, much of the work in both fields is powered by large language models (LLMs). At the same time, however, there is a growing body of work on utilizing LLMs for translation from informal language into different formalisms without using the term *autoformalization* to describe the task being solved. This lack of a widely accepted definition that captures the full range of recent applications hinders communication and knowledge transfer across subfields of formal methods, proof engineering, and AI. To address this issue, we review instances of what can be considered autoformalization and propose a common framework for understanding it.

A common and widely accepted framework for conceptualizing autoformalization has the potential to advance the next generation of AI systems. LLMs, for all their versatility, remain prone to hallucinations—producing outputs that are logically inconsistent or factually incorrect. In response, "reasoning models" have been developed that incorporate techniques such as chain-of-thought prompting and structured intermediate steps. While these approaches improve

performance on certain benchmarks, they remain costly to train, fall short on reliably executing complex logical inference (Shojaee et al. 2025), and lack mechanisms for formal verification. By contrast, symbolic AI provides rigorous reasoning and verifiability but lacks the flexibility and capacity to handle the natural language input. Autoformalization aims to combine the strengths of both paradigms: the adaptability of LLMs for handling diverse informal inputs, together with the rigour and transparency of formal systems. Establishing a shared conceptualization of autoformalization could thus accelerate progress toward AI systems that reason reliably, verifiably, and at scale.

Review of Definitions

To propose a unified cross-disciplinary framework for autoformalization, we surveyed 81 research papers. We included papers that explicitly mention autoformalization as their main goal, but also works that aim to translate natural language into formal languages for the purposes of formal verification and automated reasoning, such as Prolog, PDDL, or OWL. We do not claim that our review is exhaustive; our main objective is to survey how the term *autoformalization* is defined and operationalized across different areas of the literature, and to analyze the range of tasks that can be interpreted as instances of autoformalization, even when the term itself is not explicitly employed. In doing so, we aim to highlight both the commonalities and the differences among these tasks.

We classified the reviewed literature into four broad subfields: (1) the formalization of mathematics using interactive theorem provers, (2) logical inference and declarative programming, (3) planning, and (4) knowledge representation. These categories were chosen to reflect distinct research communities, each with its own tradition of autoformalization, developed largely in parallel. At the same time, the boundaries between these subfields are often blurred: overlaps frequently occur (e.g., logical reasoning within knowledge representation), and additional categories may emerge as each field continues to evolve.

Formalization of Mathematics with Interactive Theorem Provers The term *autoformalization* originated in the context of formalizing mathematics in 2018 (Wang, Kaliszyk, and Urban 2018), and as a result, it is widely adopted within this domain. Since the recent emergence of LLMs, users of proof assistants have been experimenting with using LLMs to assist with the formalization of mathematics, yielding mixed but potentially promising results (Karatarakis 2024). All 32 papers reviewed from this subfield use the term *autoformalization*. Most provide only informal definitions of autoformalization, which may reflect the fact that formalization of mathematics is intuitively well understood within the community and does not require defining. These informal definitions typically describe autoformalization as the task of automatically translating mathematical content from informal language into formal languages (Wu et al. 2022; Jiang et al. 2022; Agrawal et al. 2022; Li et al. 2024; Zhou et al. 2024; Ying et al. 2024; Yang et al. 2024; Liu et al. 2025a; Lu et al. 2024; Poiroux et al. 2024b; Chan et al. 2025; Jiang 2025; Azerbayev et al. 2023; Wang et al. 2020; Yu et al. 2025; Poiroux et al. 2024a; Hu et al. 2025b; Wang et al. 2025a). Some authors also explicitly include machine-verifiability of the output as part of the definition (Szegedy 2020; Murphy et al. 2024; Liu et al. 2024b, 2025c; Patel, Saha, and Flanigan 2023; Cunningham, Bunescu, and Juedes 2023; Liu et al. 2025b; Weng et al. 2025; Lu et al. 2025; Jiang, Li, and Jamnik 2024; Gadgil et al. 2022). Only one group of authors attempted a formal definition (Zhang, Quan, and Freitas 2024; Zhang, Valentino, and Freitas 2025).

Logical Inference and Declarative Programming In this category, we reviewed 30 papers. The target formalisms include first order logic (Pan et al. 2023; Lalwani et al. 2024; Han et al. 2022; Hahn et al. 2022; Yang, Ishay, and Lee 2023; Olausson et al. 2023; Chaturvedi and Asher 2024; Thatikonda et al. 2024; Ryu et al. 2024; Quan et al. 2024; Brunello et al. 2025; Liu 2025; Lee et al. 2025); declarative programming languages such as Prolog (Mensfelt, Stathis, and Trencsenyi 2024, 2025; Borazjanizadeh and Piantadosi 2024) and answer set programming (ASP) (Li et al. 2025; Coppolillo et al. 2024; Borroto Santana, Kareem, and Ricca 2024; Ishay, Yang, and Lee 2023), and temporal logics such as linear temporal logic (LTL) (Pnueli 1977) and computation tree logic (Clarke, Emerson, and Sistla 1986), (Chen et al. 2023; Pan, Chou, and Berenson 2023; William et al. 2024; Liu et al. 2024a; Mavrogiannis, Mavrogiannis, and Aloimonos 2024; Xu, Feng, and Miao 2024; Li and Tian 2025; Soroco et al. 2025; Wang et al. 2025c,b). Only 8 reviewed papers explicitly used the term *autoformalization*; all of them do so informally, typically referring to it as as the translation of informal or natural language into formal representations (Pan et al. 2023; Lalwani et al. 2024; Mensfelt, Stathis, and Trencsenyi 2024, 2025; Soroco et al. 2025; Olausson et al. 2023; Lee et al. 2025; Quan et al. 2024). Among papers that do not use the term *autoformalization*, only one formally defines the corresponding task (Liu et al. 2024a), while all others describe it informally as translation from natural language into a specific formalism. Notably, even though these papers do not use the term *autoformalization*, their definitions are consistent with those that do.

Planning The Planning Domain Definition Language (PDDL) is the de facto standard formalism within the AI planning community (Ghalab et al. 1998), encompassing various extensions ranging from basic STRIPS to first-order, probabilistic, and temporal logics (Fox and Long 2003). We included 16 papers in this category. None of them use the term *autoformalization*. One of the works provides a formal definition of the corresponding task (Hu et al. 2025a), defining it as finding a mapping function generating a world model from a natural language description. The rest of the works define the task informally (Oates et al. 2024; Liu et al. 2023; Aghzal et al. 2025; Smirnov et al. 2024; Xie et al. 2023; Mahdavi et al. 2024; Huang and Zhang 2024; Hu et al. 2025a; Lin et al. 2023; Oswald et al. 2024; Silver et al. 2024; Guan et al. 2023; Sikes et al. 2025; de la Rosa et al. 2024). In some cases the task may encompass generating both a planning domain and planning problem files (Smirnov et al.

2024), while others only require producing only parts of a planning problem or domain e.g., plan goals in (Xie et al. 2023). A related but separate line of research involves using LLMs as planners themselves, generating plans from PDDL inputs. This direction falls outside this paper's scope.

Knowledge Representation We surveyed 8 works in this category. Several of them (Mateiu and Groza 2023; Doumanas et al. 2024; Caufield et al. 2024; Saeedizade and Blomqvist 2024) focus on translating natural language descriptions of real-world domains into ontologies using the Web Ontology Language (OWL) (OWL Working Group 2012), a formal language used to define and share structured, machine-readable knowledge in terms of individuals, concepts, and relationships between them. Two other works (Eells et al. 2024; Abolhasani and Pan 2024) produce ontologies expressed in RDF or RDFS (Brickley and Guha 2014). Others produce ontologies expressed in ad-hoc languages (Tang et al. 2023; Lippolis et al. 2025), which seem to be expressible in OWL or RDFS. None of the reviewed works in this category employs the term *autoformalization*.

Summary As observed, definitions across domains exhibit notable similarities, even when the task is not explicitly labelled as *autoformalization*. At the core, these tasks involve the automatic translation of natural language into a target language that supports logical inference and automated reasoning.

Proposed Definition

Building on the shared foundations presented in the previous section, we propose a general definition of autoformalization designed to capture its essential components across the works we reviewed, while abstracting from implementation- or domain-specific details.

Definition 1. *Formalization* from informal language L_i to formal reasoning language L_f with respect to a semantic equivalence criterion E is the transformation of an expression in a domain-specific subset of L_i into a well-formed and valid expression in L_f that is semantically equivalent according to E. *Autoformalization* is formalization performed automatically by a computing system.

Our definition involves four parameters: the informal language L_i, a domain-specific subset of L_i, the formal language L_f, and the semantic equivalence criterion E. These are treated as primitives without formal definition. However, we proceed to clarify their meanings, provide examples, and explain our rationale for including them in the definition.

Informal language An informal language L_i is a collection of meaningful expressions, such as the set of all grammatically well-formed and semantically coherent texts in English. This abstraction deliberately ignores more fine-grained aspects, such as lexical categories or compositional syntax. Membership in informal languages is generally neither decidable nor computable, and their boundaries may be vague or context-dependent. Sometimes we consider *semi-formal* languages: informal languages (like plain English) mixed with elements of a formal language. For example, the mathematical language of proofs is typically semi-formal.

Our definitions refers to a domain-specific subset of L_i because, in a given autoformalization setting, we typically do not seek to formalize all possible expressions in L_i, but only a specific subset of interest that share a common conceptual framework or subject matter. For example, within plain English, we may consider: descriptions of two-player games, specifications of procedural planning settings, and natural language descriptions of real-world systems in terms of their objects, properties, and relations (i.e. informal ontologies), among many other examples.

Formal reasoning language The formal reasoning language L_f is an enumerable set of expressions typically specified by a grammar or formation rules, and is accompanied by a formal semantics that assigns precise and unambiguous meaning to each well-formed expression. It is also associated with a reasoning apparatus—usually a set of inference or transformation rules—that enables the derivation of new expressions from existing ones while preserving semantic properties. Examples of L_f include: languages of proof assistants, i.e., interactive theorem provers (e.g., Lean, Isabelle, Mizar, Rocq); fully formal logics (e.g., propositional logic, FOL, LTL); logic programming and declarative languages (e.g., Answer Set Programming, Prolog); planning and CSP languages (e.g., PDDL, MiniZinc); knowledge representation languages and formalisms (e.g., OWL, RDF, Situation Calculus, Event Calculus).

The choice of formal language L_f in a given autoformalization setting is guided by two considerations: first, it must be capable of representing all relevant information in the domain-specific subset of L_i; second, it must support reasoning in a way that allows properties of interest to be verified, both in terms of computational complexity and availability of appropriate verification tools. For example, if our goal is to formalise descriptions of planning scenarios, L_f might be a planning modeling language such as STRIPS.

Semantic equivalence criterion The semantic equivalence criterion E specifies what it means for the obtained formal expression from L_f to preserve the intended meaning of the informal input from L_i. While the goal is for both expressions to 'mean the same thing,' formal languages are usually less expressive than informal languages and cannot capture every aspect of the original meaning. Hence, in practice, E is designed to ensure that the formalization preserves the aspects of meaning that are most relevant for the task at hand. For example, in an interactive theorem prover, E may require that the formalization contains sufficient detail to ensure that formalisations of correct proofs (expressed informally in L_i) can be verified by a given tool or reasoning algorithm.

Specifying the semantic equivalence criterion can be challenging, as it requires defining conditions related to the meaning of informal language, which is inherently ambiguous. To address this, we introduce a related concept, the *validation criterion* V, which serves as a computable approximation of E. This distinction is important: E defines what it means for an informal and formal expression to "mean the same thing" as a formal expression, whereas V provides a practical means of verifying whether a given pair of infor-

mal and formal expressions satisfies this equivalence.

We conclude this section with several observations that further clarify the scope of the definition.

Remark 1. An informal expression in L_i may be formalized in multiple distinct ways into L_f, differing in their choice of names, syntactic structure, or level of abstraction. Conversely, two distinct expressions in L_i may also be formalized into the same expression in L_f if they convey the same information relevant to the problem at hand.

Remark 2. Systems for autoformalization need not be deterministic: the same informal input may be mapped to different formal outputs in different runs. These outputs are typically "equivalent" or "isomorphic" in some sense within L_f. In practice, however, autoformalization systems aim to produce syntactically or structurally minimal outputs, promoting conciseness and efficiency

Remark 3. The expressions from L_i and L_f in our proposed definition can be quite large; for example, we may formalize a full mathematical proof into a full proof expressed in Rocq's language. Many applications, however, require equivalence to hold at a finer granularity. For example, we may require that each sentence in a mathematical proof should be formalized as a statement in Rocq's language that faithfully captures its meaning. In the definition we propose, a fine-grained correspondence such as this can be subsumed within the semantic equivalence criterion E.

Case Studies

We next revisit examples of autoformalization from the reviewed literature and show how they fit in our framework.

Formalization of Mathematics with Interactive Theorem Provers The following example comes from a seminal paper on autoformalization with LLMs (Wu et al. 2022). Here, the informal language L_i is the natural mathematical language (plain English extended with mathematical symbols) and the relevant subset includes the class of mathematical problems in the 1987 International Mathematical Olympiad. The target language L_f is the proof language *Isar* of the theorem prover Isabelle/HOL (Nipkow, Wenzel, and Paulson 2002), which includes keywords, mathematical symbols, and logical operators, and has a formally defined syntax. Isabelle/HOL provides a higher-order logic theorem proving environment, realised by various automatic tools and a number of external automatic theorem provers and satisfiability-modulo-theories solvers which can be called by the Sledgehammer tool featured by Isabelle/HOL (Blanchette et al. 2025)). An input expression from L_i to be formalized was:

> *Prove that there is no function f from the set of non-negative integers into itself such that $f(f(n)) = n + 1987$ for every n.*

The corresponding formalization in Isabelle/HOL produced by the Codex LLM tool by (Wu et al. 2022) was:

```
theorem
fixes f :: "nat \<Rightarrow> nat"
assumes "\<forall> n. f (f n) = n + 1987"
shows False
```

The statement `shows False` indicates that the assumption of the function's existence leads to a contradiction. The semantic equivalence criterion E in this case requires that all important mathematical details from the original theorem statement are captured in the formal statement so as to allow for correctness verification. Two validation criteria V were used here to evaluate E. The first is BLEU scores (Papineni et al. 2002), which measure surface-level similarity between the autoformalized statement and the ground truth (i.e. a manual formalization deemed correct by human experts), but not direct semantic equivalence. The second criterion was manual evaluation of selected examples, which is reliable but costly and difficult to scale. This highlights a fundamental challenge in autoformalization: automating the evaluation of semantic correspondence.

First-Order Logic (FOL) The next example is from (Pan et al. 2023), who explicitly use the term *autoformalization* and state that their goal is to extend autoformalization into the domain of logical reasoning. Their informal language L_i is simply natural language, and they foucs on the subset of descriptions of and questions about real-word objects and shapes. The target formal system L_f is *Prover9*, which is based on FOL syntax and includes a special signature of predicates describing geometric objects and properties (e.g., `Square`, `FourSided`, `Shape`). The reasoning apparatus for this language is instantiated by Prover9; it is a sound calculus based on resolution and paramodulation. Consider the following example in their paper, from benchmark FOLIO:

> ***Context:*** *All squares have four sides. All four-sided things are shapes.*
> ***Question:*** *Based on the above information, is the following statement true, false, or uncertain? All squares are shapes.*
> ***Options:*** *A) True B) False C) Uncertain*

This informal input is translated into the following expression, consisting of three different sets of FOL formulas:

> **Predicates:**
> `Square(x)` — x is a square
> `FourSided(x)` — x has four sides
> `Shape(x)` — x is a shape
> **Premises:**
> $\forall x$(`Square(x)` $\rightarrow$ `FourSided(x)`)
> $\forall x$(`FourSided(x)` $\rightarrow$ `Shape(x)`)
> **Conclusion:**
> $\forall x$(`Square(x)` $\rightarrow$ `Shape(x)`)

Prover9 then verifies that the conclusion logically follows from the premises via transitive implication. The semantic equivalence criterion E relies on all relevant properties of the objects described in the question being captured by the logical formulas, without introducing spurious properties. The validation criterion V used to measure semantic equivalence consists in comparing the answer (A, B, or C) derived by Prover9 with the ground truth correct answer. This approach enables automated evaluation but it does not guarantee semantic equivalence.

Logic Programming The next example involves formalizing strategic interactions that can be modeled as bima-

trix games in order to reason about them formally in Prolog (Mensfelt, Stathis, and Trencsenyi 2025). The informal language L_i is once again natural language, and the domain-specific subset we focus on is the set of aforementioned strategic interactions. The formal language L_f is Prolog, which has a well-defined semantics and a reasoning apparatus that relies on logical inference—instantiated by SWI-Prolog in this case. We consider the following example:

> *A couple is deciding how to spend their evening together. One prefers to go to the opera and the other prefers to go to a football game, but both prefer to be together rather than apart. If they both choose the opera, the opera lover gets a payoff of 2 units of happiness, and the football lover gets 1 unit. If they both choose the football game, the football lover gets 2 units of happiness, and the opera lover gets 1 unit. If they choose different events, they both get 0 units.*

The snippet below, which is part of the formalization output, shows the payoff matrix for the game and the subset of predicates defining the initial state:

```
...
payoffBOS(opera, opera, 2, 1).
payoffBOS(football, football, 1, 2).
payoffBOS(opera, football, 0, 0).
payoffBOS(football, opera, 0, 0).

initially(player(p1), s0).
initially(player(p2), s0).
initially(role(p1, opera_lover), s0).
initially(role(p2, football_lover), s0).
...
```

The semantic equivalence criterion E requires that the Prolog program correctly predicts the payoffs for each possible choice of the agents in the game (e.g. player 1 selecting opera and player 2 selecting football should result in payoffs of (0,0) when we query the generated formalization). The verification V of the semantic equivalence criterion is automated by querying all possible combinations of actions and comparing the resulting payoffs to ground-truth payoffs for each pair of actions. This method is automated and ensures practical semantic equivalence. However, it applies only to a narrow class of problems—bimatrix games.

Planning We next chose an example used in (Oswald et al. 2024), covering the domain of planning. The informal language is natural language; the relevant subset of expressions are those describing—fully or partially—real-world domains where planning is relevant, including the available actions and their results. The formal language L_f is PDDL, a FOL-based formal language for representing classical planning problems, where states are described using propositional atoms, actions define state transitions, and the goal is to find a sequence of actions that transforms the initial state into one satisfying goal conditions. The formal reasoning apparatus for PDDL requires planners like (Helmert 2006) that interpret a PDDL domain description and solve planning problems. The paper uses the planner K* planner (Lee, Katz, and Sohrabi 2023). Consider the following informal action description:

> *The action, "FLY-AIRPLANE" will fly an airplane from one airport to another. After the action, the airplane will be in the new location.*

The output is a PDDL representation of the described action, where the given action's preconditions and effects are formalized to model the planning domain. Note that this is aided by adding a list of allowed predicates to the informal action description prompt:

```
(:action FLY-AIRPLANE
  :parameters (?airplane - airplane
  ?loc-from  - airport ?loc-to - airport)
  :precondition (at ?airplane ?loc-from)
  :effect (and (not (at ?airplane
  ?loc-from)) (at ?airplane ?loc-to)) )
```

Semantic equivalence is determined by both the behavior of actions and their logical consequences. Since it is generally impractical to check this directly, we instead rely on a validation criterion. The validation criterion checks whether the formalized action behaves equivalently to a known ground-truth formalization via the *heuristic domain equivalence* test, where representative planning problems are solved in two domains that differ only in the action being tested: one with the ground-truth action and one with the reconstructed action. The plans from each domain are then cross-validated in the other, and if all plans succeed, the reconstructed action is deemed semantically equivalent. This procedure defines a relaxed notion of equivalence that has proved very effective in practice.

Knowledge Representation The following example, by (Mateiu and Groza 2023), involves translating texts in informal natural language L_i describing real-world domain knowledge into ontologies expressed in the Web Ontology Language (OWL), which therefore acts as formal language L_f. OWL is a formalism based on first-order logic used to model knowledge about a domain in terms of named individuals, classes, and roles—equivalent to FOL constants, unary predicates, and binary predicates, respectively. Furthermore, OWL statments are restricted to use specific logicla connectives that corresponds to specific fragments of guarded, 2-variable FOL. It is designed to achieve a good balance between expressive power and computational complexity.

To illustrate the approach of (Mateiu and Groza 2023), we recapitulate the output of their system on the sentence *Anna and Lana are each other's sisters*:

```
Declaration(ObjectProperty(:has_sister))
     Declaration(NamedIndividual(:Anna))
     Declaration(NamedIndividual(:Lana))
   ObjPropAssert(:has_sister :Anna :Lana)
   ObjPropAssert(:has_sister :Lana :Anna)
```

The first statement identifies the role (binary predicate) `has_sister` and named individuals `Anna` and `Lana`. Then it generates two additional assertions to represent the symmetry implied in the sentence.

In this setting, the semantic equivalence criterion E requires that all relational information in the informal description be correctly captured in the OWL ontology. In practice, this cannot be formally measured, so the paper establishes a validation criterion V based on ontology engineers

manually verifying that the ontology encodes all relevant knowledge. In general, however, such manual V can be supplemented with an automated component: assuming the informal knowledge is consistent—as is usually the case—automated reasoning tools can check consistency of the formalization. Furthermore, when the expected consequences and undesirable outcomes of the knowledge base are known and have admissible formalizations, reasoners can be used to verify formally that the ontology both entails all intended consequences and avoiding unintended ones.

Challenges and Open Problems

While there is a growing body of work aimed at improving autoformalization (Patel, Saha, and Flanigan 2023; Zhang, Quan, and Freitas 2024; Li et al. 2024; Tarrach et al. 2024; Chan et al. 2025; Jiang, Li, and Jamnik 2024; Lu et al. 2025; Poiroux et al. 2024b; Liu et al. 2025b), several fundamental challenges still offer exciting research opportunities that will shape the field's future development. We next provide a selective overview of these challenges and opportunities.

Semantic Verification The biggest challenge in autoformalization remains verifying equivalence of the informal input and its formalization. Current approaches rely mostly on human supervision, either through direct assessment or comparison with reference solutions; sometimes enhanced with automated checks such as consistency checking of the extracted formalism. Fully formal and automated verification of semantic equivalence, however, appears impossible even in principle, since natural language and its intended meaning are inherently informal. An open question is whether it is possible to train intelligent systems to acquire the dual competence that humans possess: understanding both formal and informal domains, and being able to assess correspondence between them. Such verification capabilities are in principle distinct from the current practice of autoformalization, but they could be integrated into it rather than applied post-hoc.

Target Formal Languages Selecting an appropriate target formal language for a given task remains a significant challenge. The language must be sufficiently expressive to capture all relevant semantic content of the informal text, yet there is an inherent tension between expressive power and computational complexity of verification, which can render automated reasoning difficult or even undecidable. The choice of language often depends on the intended use case: who will interact with the formalized text, and what computational tasks must be supported? For example, if the goal is interactive theorem proving, the availability and sophistication of verification tools, as well as the contents of formal proof libraries, are critical. Conversely, if human mathematicians will work with the output, considerations such as readability and adherence to established community conventions become crucial. Given these constraints, a promising direction for future research is the design of new formal languages specifically tailored for autoformalization, balancing expressivity, verifiability, and usability.

Scalability and Integration with Mathematical Practice Current autoformalization systems perform well on curated benchmarks but often struggle to scale to real-world use cases. For instance, they remain limited in supporting actual mathematical research, where proofs in published papers are dominated by incomplete arguments, informal reasoning, and proof sketches. Contextual understanding and gap-filling continue to be major challenges. This gap in scalability also reflects uncertainty about how autoformalization should fit into research workflows. Without clear guidance on whether it is intended to support proof development, literature review, verification, or hypothesis testing, there is a risk of creating technically impressive systems that remain peripheral to practical applications.

Interactive Formalization Many of the current autoformalization approaches treat the task as a one-shot translation problem. This is difficult to square with the inherent ambiguity and context-dependence of natural or mathematical language. A more promising direction involves systems that can engage in dialogue with users, asking clarifying questions about ambiguous terms, implicit assumptions, or intended interpretations. Early feedback can help speed-up the system by enabling earlier backtracking. The system should recognise when clarification is needed and formulate meaningful questions to the user, which then would be incorporated to iteratively refine the formalization.

Cross-Domain Transfer and Generalization Current autoformalization approaches are often domain-specific, with systems designed separately for areas such as mathematical reasoning, planning, or logic. Developing general methods that can transfer reasoning patterns and formalization strategies across these different areas remains a challenge. Our present analysis of the notion of autoformalization shows that the underlying formal structures and reasoning patterns in these areas often share commonalities that could be exploited for more general autoformalization systems.

Conclusions

In practice, we judge successful formalization by its fruits: Does it help us solve problems? Does it support verification? Does it reveal new connections? Does it eliminate disputes? Does it enable computation? Formalization is less about focussing on mere translation and more about creating tools that extend our reasoning capabilities in useful ways.

One of the most promising applications is enhancing the reasoning capabilities of LLMs and developing robust, general-purpose reasoners. Achieving this through autoformalization presents many challenges. A common framework could accelerate progress by facilitating knowledge exchange and enabling transfer learning.

As we have demonstrated, translating problems into different formalisms for logical inference and automated reasoning shares similarities and common challenges. Based on the reviewed papers, we propose a preliminary definition that encompasses various types of formalisms. This definition is not intended to be definitive; rather, our aim is to initiate a cross-disciplinary dialogue on how best to conceptualize autoformalization, articulate its desired properties, and promote effective interdisciplinary collaboration.

Acknowledgments

This work was supported by a Leverhulme Trust International Professorship Grant (LIP-2022-001).

References

Abolhasani, M. S.; and Pan, R. 2024. Leveraging LLM for Automated Ontology Extraction and Knowledge Graph Generation. *CoRR*, abs/2412.00608.

Aghzal, M.; Plaku, E.; Stein, G. J.; and Yao, Z. 2025. A Survey on Large Language Models for Automated Planning.

Agrawal, A.; Gadgil, S.; Goyal, N.; Narayanan, A.; and Tadipatri, A. 2022. Towards a Mathematics Formalisation Assistant using Large Language Models. ArXiv:2211.07524.

Azerbayev, Z.; Piotrowski, B.; Schoelkopf, H.; Ayers, E. W.; Radev, D.; and Avigad, J. 2023. ProofNet: Autoformalizing and Formally Proving Undergraduate-Level Mathematics.

Blanchette, J.; Desharnais, M.; Paulson, L. C.; and Bartl, L. 2025. *Hammering Away A User's Guide to Sledgehammer for Isabelle/HOL*. https://isabelle.in.tum.de/dist/Isabelle2025/doc/sledgehammer.pdf: tutorial.

Borazjanizadeh, N.; and Piantadosi, S. T. 2024. Reliable Reasoning Beyond Natural Language. *arXiv preprint arXiv:2407.11373*.

Borroto Santana, M.; Kareem, I.; and Ricca, F. 2024. Towards Automatic Composition of ASP Programs from Natural Language Specifications. *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*.

Brickley, D.; and Guha, R. V. 2014. RDF Schema 1.1. https://www.w3.org/TR/rdf-schema/. W3C Recommendation.

Brunello, A.; Ferrarese, R.; Geatti, L.; Marzano, E.; Montanari, A.; and Saccomanno, N. 2025. Evaluating LLMs Capabilities at Natural Language to Logic Translation: A Preliminary Investigation. In *Proceedings of the Logic and Engineering of Natural Language Semantics (LENLS) Workshop 2025*.

Caufield, J. H.; Hegde, H.; Emonet, V.; Harris, N. L.; Joachimiak, M. P.; Matentzoglu, N.; Kim, H.; Moxon, S. A. T.; Reese, J. T.; Haendel, M. A.; Robinson, P. N.; and Mungall, C. J. 2024. Structured Prompt Interrogation and Recursive Extraction of Semantics (SPIRES): a method for populating knowledge bases using zero-shot learning. *Bioinform.*, 40(3).

Chan, W.; Souliman, M.; Nordhagen, J.; Miranda, B.; Obbad, E.; and Koyejo, K. F. S. 2025. Lean-ing on Quality: How High-Quality Data Beats Diverse Multilingual Data in AutoFormalization.

Chaturvedi, A.; and Asher, N. 2024. Learning Semantic Structure through First-Order-Logic Translation. *arXiv preprint arXiv:2410.03203*.

Chen, Y.; Gandhi, R.; Zhang, Y.; and Fan, C. 2023. NL2TL: Transforming Natural Languages to Temporal Logics using Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 15880–15903.

Clarke, E. M.; Emerson, E. A.; and Sistla, A. P. 1986. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2): 244–263.

Coppolillo, E.; Calimeri, F.; Manco, G.; Perri, S.; and Ricca, F. 2024. LLASP: Fine-tuning Large Language Models for Answer Set Programming. *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning*.

Cunningham, G.; Bunescu, R. C.; and Juedes, D. 2023. Towards Autoformalization of Mathematics and Code Correctness: Experiments with Elementary Proofs.

de la Rosa, T.; Gopalakrishnan, S.; Pozanco, A.; Zeng, Z.; and Borrajo, D. 2024. TRIP-PAL: Travel Planning with Guarantees by Combining Large Language Models and Automated Planners.

Doumanas, D.; Soularidis, A.; Kotis, K.; and Vouros, G. A. 2024. Integrating LLMs in the Engineering of a SAR Ontology. In Maglogiannis, I.; Iliadis, L. S.; MacIntyre, J.; Avlonitis, M.; and Papaleonidas, A., eds., *Artificial Intelligence Applications and Innovations - 20th IFIP WG 12.5 International Conference, AIAI 2024, Corfu, Greece, June 27-30, 2024, Proceedings, Part IV*, volume 714 of *IFIP Advances in Information and Communication Technology*, 360–374. Springer.

Eells, A.; Dave, B.; Hitzler, P.; and Shimizu, C. 2024. Commonsense Ontology Micropatterns. In Besold, T. R.; d'Avila Garcez, A.; Jiménez-Ruiz, E.; Confalonieri, R.; Madhyastha, P.; and Wagner, B., eds., *Neural-Symbolic Learning and Reasoning - 18th International Conference, NeSy 2024, Barcelona, Spain, September 9-12, 2024, Proceedings, Part II*, volume 14980 of *Lecture Notes in Computer Science*, 51–59. Springer.

Fox, M.; and Long, D. 2003. PDDL2. 1: An extension to PDDL for expressing temporal planning domains. *Journal of artificial intelligence research*, 20: 61–124.

Gadgil, S.; Tadipatri, A. R.; Agrawal, A.; Narayanan, A.; and Goyal, N. 2022. Towards automating formalisation of theorem statements using large language models. In *36th Conference on Neural Information Processing Systems (NeurIPS 2022) Workshop on MATH-AI*.

Ghalab, M.; Howe, A.; Knoblock, C.; McDermott, I. D.; Ram, A.; Veloso, M.; Weld, D.; Sri, D. W.; Barrett, A.; Christianson, D.; et al. 1998. Pddl—the planning domain definition language. *IPC-98 Technical Report*.

Guan, L.; Valmeekam, K.; Sreedharan, S.; and Kambhampati, S. 2023. Leveraging pre-trained large language models to construct and utilize world models for model-based task planning. *Advances in Neural Information Processing Systems*, 36: 79081–79094.

Hahn, C.; Schmitt, F.; Tillman, J. J.; Metzger, N.; Siber, J.; and Finkbeiner, B. 2022. Formal Specifications from Natural Language.

Han, S.; Schoelkopf, H.; Zhao, Y.; Qi, Z.; Riddell, M.; Zhou, W.; Coady, J.; Peng, D.; Qiao, Y.; Benson, L.; Sun, L.; Wardle-Solano, A.; Szabo, H.; Zubova, E.; Burtell, M.; Fan, J.; Liu, Y.; Wong, B.; Sailor, M.; Ni, A.; Nan, L.; Kasai, J.;

Yu, T.; Zhang, R.; Fabbri, A. R.; Kryscinski, W.; Yavuz, S.; Liu, Y.; Lin, X. V.; Joty, S.; Zhou, Y.; Xiong, C.; Ying, R.; Cohan, A.; and Radev, D. 2022. FOLIO: Natural Language Reasoning with First-Order Logic.

Helmert, M. 2006. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26: 191–246.

Hu, M.; Chen, T.; Zou, Y.; Lei, Y.; Chen, Q.; Li, M.; Mu, Y.; Zhang, H.; Shao, W.; and Luo, P. 2025a. Text2World: Benchmarking Large Language Models for Symbolic World Model Generation.

Hu, X.; Zhou, Q.; Grechuk, B.; and Tyukin, I. Y. 2025b. StepProof: Step-by-step verification of natural language mathematical proofs. *arXiv preprint arXiv:2506.10558*.

Huang, C.; and Zhang, L. 2024. On the Limit of Language Models as Planning Formalizers.

Ishay, A.; Yang, Z.; and Lee, J. 2023. Leveraging Large Language Models to Generate Answer Set Programs. *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*.

Jiang, A. Q.; Li, W.; and Jamnik, M. 2024. Multi-language diversity benefits autoformalization. *Advances in Neural Information Processing Systems*, 37: 83600–83626.

Jiang, A. Q.; Welleck, S.; Zhou, J. P.; Li, W.; Liu, J.; Jamnik, M.; Lacroix, T.; Wu, Y.; and Lample, G. 2022. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. *arXiv preprint arXiv:2210.12283*.

Jiang, Q. 2025. Language models for verifiable mathematical automation: Interaction, integration, and autoformalization.

Kaliszyk, C.; Urban, J.; and Vyskočil, J. 2015. Learning to parse on aligned corpora (rough diamond). In *Interactive Theorem Proving: 6th International Conference, ITP 2015, Nanjing, China, August 24-27, 2015, Proceedings 6*, 227–233. Springer.

Kaliszyk, C.; Urban, J.; and Vyskočil, J. 2017. Automating formalization by statistical and semantic parsing of mathematics. In *International Conference on Interactive Theorem Proving*, 12–27. Springer.

Kaliszyk, C.; Urban, J.; Vyskočil, J.; and Geuvers, H. 2014. Developing corpus-based translation methods between informal and formal mathematics: Project description. In *International Conference on Intelligent Computer Mathematics*, 435–439. Springer.

Kamath, A.; and Das, R. 2018. A Survey on Semantic Parsing. *arXiv preprint arXiv:1812.00978*.

Karatarakis, M. 2024. Leveraging Large Language Models for Autoformalizing Theorems: A Case Study. In *9th Conference on Artificial Intelligence and Theorem Proving (AITP)*.

Kowalski, R. 1974. Predicate logic as a programming language. In *IFIP Congress*, 569–574.

Lalwani, A.; Kim, T.; Chopra, L.; Hahn, C.; Jin, Z.; and Sachan, M. 2024. Autoformalizing Natural Language to First-Order Logic: A Case Study in Logical Fallacy Detection.

Lee, J.; Katz, M.; and Sohrabi, S. 2023. On k* search for top-k planning. In *Proceedings of the International Symposium on Combinatorial Search*, volume 16, 38–46.

Lee, J.; Lee, D.; Choi, C.; Im, Y.; Wi, J.; Heo, K.; Oh, S.; Lee, S.; and Shin, I. 2025. Safeguarding Mobile GUI Agent via Logic-based Action Verification.

Li, J.; and Tian, M. T. 2025. Automatic Generation of Safety-compliant Linear Temporal Logic via Large Language Model: A Self-supervised Framework. *arXiv preprint arXiv:2503.15840*.

Li, N.; Liu, P.; Liu, Z.; Dai, T.; Jiang, Y.; and Xia, S.-T. 2025. Logic-of-Thought: Empowering Large Language Models with Logic Programs for Solving Puzzles in Natural Language. *arXiv preprint arXiv:2505.16114*.

Li, Z.; Wu, Y.; Li, Z.; Wei, X.; Zhang, X.; Yang, F.; and Ma, X. 2024. Autoformalize Mathematical Statements by Symbolic Equivalence and Semantic Consistency.

Lin, K.; Agia, C.; Migimatsu, T.; Pavone, M.; and Bohg, J. 2023. Text2Motion: from natural language instructions to feasible plans. *Autonomous Robots*, 47(8): 1345–1365.

Lippolis, A. S.; Saeedizade, M. J.; Keskisärkkä, R.; Zuppiroli, S.; Ceriani, M.; Gangemi, A.; Blomqvist, E.; and Nuzzolese, A. G. 2025. Ontology Generation Using Large Language Models. In Curry, E.; Acosta, M.; Poveda-Villalón, M.; van Erp, M.; Ojo, A. K.; Hose, K.; Shimizu, C.; and Lisena, P., eds., *The Semantic Web - 22nd European Semantic Web Conference, ESWC 2025, Portoroz, Slovenia, June 1-5, 2025, Proceedings, Part I*, volume 15718 of *Lecture Notes in Computer Science*, 321–341. Springer.

Liu, B.; Jiang, Y.; Zhang, X.; Liu, Q.; Zhang, S.; Biswas, J.; and Stone, P. 2023. LLM+P: Empowering Large Language Models with Optimal Planning Proficiency.

Liu, C.; Yuan, Y.; Yin, Y.; Xu, Y.; Xu, X.; Chen, Z.; Wang, Y.; Shang, L.; Liu, Q.; and Zhang, M. 2025a. Safe: Enhancing Mathematical Reasoning in Large Language Models via Retrospective Step-aware Formal Verification. *arXiv preprint arXiv:2506.04592*.

Liu, J. 2025. Few-Shot Natural Language to First-Order Logic Translation via In-Context Learning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 10939–10960.

Liu, J. X.; Shah, A.; Konidaris, G.; Tellex, S.; and Paulius, D. 2024a. Lang2LTL-2: Grounding Spatiotemporal Navigation Commands Using Large Language and Vision-Language Models. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2325–2332. ISSN: 2153-0866.

Liu, Q.; Zheng, X.; Lu, X.; Cao, Q.; and Yan, J. 2024b. Rethinking and Improving Autoformalization: Towards a Faithful Metric and a Dependency Retrieval-based Approach. In *Proceedings of The Thirteenth International Conference on Learning Representations*.

Liu, Q.; Zheng, X.; Lu, X.; Cao, Q.; and Yan, J. 2025b. Rethinking and Improving Autoformalization: Towards a

Faithful Metric and a Dependency Retrieval-based Approach. In *The Thirteenth International Conference on Learning Representations (ICLR)*. Spotlight paper.

Liu, X.; Bao, K.; Zhang, J.; Liu, Y.; Chen, Y.; Liu, Y.; Jiao, Y.; and Luo, T. 2025c. ATLAS: Autoformalizing Theorems through Lifting, Augmentation, and Synthesis of Data.

Lu, J.; Wan, Y.; Huang, Y.; Xiong, J.; Liu, Z.; and Guo, Z. 2025. FormalAlign: Automated Alignment Evaluation for Autoformalization. In *The Thirteenth International Conference on Learning Representations (ICLR)*. Preprint accepted to ICLR 2025.

Lu, J.; Wan, Y.; Liu, Z.; Huang, Y.; Xiong, J.; Liu, C.; Shen, J.; Jin, H.; Zhang, J.; Wang, H.; Yang, Z.; Tang, J.; and Guo, Z. 2024. Process-Driven Autoformalization in Lean 4.

Mahdavi, S.; Aoki, R.; Tang, K.; and Cao, Y. 2024. Leveraging Environment Interaction for Automated PDDL Translation and Planning with Large Language Models.

Mateiu, P.; and Groza, A. 2023. Ontology engineering with Large Language Models. In *25th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2023, Nancy, France, September 11-14, 2023*, 226–229. IEEE.

Mavrogiannis, A.; Mavrogiannis, C.; and Aloimonos, Y. 2024. Cook2LTL: Translating Cooking Recipes to LTL Formulae using Large Language Models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 17679–17686.

McCarthy, J. 1960. Programs with Common Sense. Technical report, USA.

Mensfelt, A.; Stathis, K.; and Trencsenyi, V. 2024. Autoformalization of Game Descriptions using Large Language Models.

Mensfelt, A.; Stathis, K.; and Trencsenyi, V. 2025. Generative Agents for Multi-Agent Autoformalization of Interaction Scenarios. Accepted at European Conference on Artificial Intelligence (ECAI) 2025.

Murphy, L.; Yang, K.; Sun, J.; Li, Z.; Anandkumar, A.; and Si, X. 2024. Autoformalizing Euclidean Geometry.

Newell, A.; and Simon, H. A. 1976. Computer science as empirical inquiry: Symbols and search. *Communications of the ACM*, 19(3): 113–126.

Nipkow, T.; Wenzel, M.; and Paulson, L. C. 2002. *Isabelle/HOL: a proof assistant for higher-order logic*. Springer.

Oates, T.; Alford, R.; Johnson, S.; and Hall, C. 2024. Using Large Language Models to Extract Planning Knowledge from Unstructured Text. In *Proceedings of the ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Olausson, T.; Gu, A.; Lipkin, B.; Zhang, C.; Solar-Lezama, A.; Tenenbaum, J.; and Levy, R. 2023. LINC: A Neurosymbolic Approach for Logical Reasoning by Combining Language Models with First-Order Logic Provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 5153–5176. Singapore: Association for Computational Linguistics.

Oswald, J.; Srinivas, K.; Kokel, H.; Lee, J.; Katz, M.; and Sohrabi, S. 2024. Large Language Models as Planning Domain Generators. *Proceedings of the International Conference on Automated Planning and Scheduling*, 34: 423–431.

OWL Working Group. 2012. OWL 2 Web Ontology Language Document Overview (Second Edition). W3c recommendation, World Wide Web Consortium (W3C). Accessed: 2025-07-28.

Pan, J.; Chou, G.; and Berenson, D. 2023. Data-Efficient Learning of Natural Language to Linear Temporal Logic Translators for Robot Task Specification. *Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA)*.

Pan, L.; Albalak, A.; Wang, X.; and Wang, W. Y. 2023. Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311–318.

Patel, N.; Saha, R.; and Flanigan, J. 2023. A New Approach Towards Autoformalization.

Pnueli, A. 1977. The temporal logic of programs. *18th Annual Symposium on Foundations of Computer Science*, 46–57.

Poiroux, A.; Weiss, G.; Kunčak, V.; and Bosselut, A. 2024a. Improving Autoformalization using Type Checking. *arXiv preprint arXiv:2406.07222*.

Poiroux, A.; Weiss, G.; Kunčak, V.; and Bosselut, A. 2024b. Improving Autoformalization using Type Checking.

Quan, X.; Valentino, M.; Dennis, L. A.; and Freitas, A. 2024. Verification and Refinement of Natural Language Explanations through LLM-Symbolic Theorem Proving.

Ryu, H.; Kim, G.; Lee, H. S.; and Yang, E. 2024. Divide and Translate: Compositional First-Order Logic Translation and Verification for Complex Logical Reasoning.

Saeedizade, M. J.; and Blomqvist, E. 2024. Navigating Ontology Development with Large Language Models. In Meroño-Peñuela, A.; Dimou, A.; Troncy, R.; Hartig, O.; Acosta, M.; Alam, M.; Paulheim, H.; and Lisena, P., eds., *The Semantic Web - 21st International Conference, ESWC 2024, Hersonissos, Crete, Greece, May 26-30, 2024, Proceedings, Part I*, volume 14664 of *Lecture Notes in Computer Science*, 143–161. Springer.

Shojaee, P.; Mirzadeh, I.; Alizadeh, K.; Horton, M.; Bengio, S.; and Farajtabar, M. 2025. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. *arXiv preprint arXiv:2506.06941*.

Sikes, K.; Fine-Morris, M.; Sreedharan, S.; Smith, L. N.; and Roberts, M. 2025. Creating PDDL Models from Javascript using LLMs: Preliminary Results. In *AAAI 2025 Workshop LM4Plan*.

Silver, T.; Dan, S.; Srinivas, K.; Tenenbaum, J. B.; Kaelbling, L.; and Katz, M. 2024. Generalized Planning in PDDL Domains with Pretrained Large Language Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(18): 20256–20264.

Smirnov, P.; Joublin, F.; Ceravola, A.; and Gienger, M. 2024. Generating consistent PDDL domains with Large Language Models.

Soroco, M.; Song, J.; Xia, M.; Emond, K.; Sun, W.; and Chen, W. 2025. PDE-Controller: LLMs for Autoformalization and Reasoning of PDEs. *arXiv preprint arXiv:2502.00963*.

Szegedy, C. 2020. A promising path towards autoformalization and general artificial intelligence. In *Intelligent Computer Mathematics: 13th International Conference, CICM 2020, Bertinoro, Italy, July 26–31, 2020, Proceedings 13*, 3–20. Springer.

Tang, Y.; da Costa, A. A. B.; Zhang, X.; Irvine, P.; Khastgir, S.; and Jennings, P. A. 2023. Domain Knowledge Distillation from Large Language Model: An Empirical Study in the Autonomous Driving Domain. In *26th IEEE International Conference on Intelligent Transportation Systems, ITSC 2023, Bilbao, Spain, September 24-28, 2023*, 3893–3900. IEEE.

Tarrach, G.; Jiang, A. Q.; Raggi, D.; Li, W.; and Jamnik, M. 2024. More details, please: Improving autoformalization with more detailed proofs. In *AI for Math Workshop@ ICML 2024*.

Thatikonda, R. K.; Han, J.; Buntine, W.; and Shareghi, E. 2024. Strategies for Improving NL-to-FOL Translation with LLMs: Data Generation, Incremental Fine-Tuning, and Verification.

Wang, H.; Unsal, M.; Lin, X.; Baksys, M.; Liu, J.; Santos, M. D.; Sung, F.; Vinyes, M.; Ying, Z.; Zhu, Z.; et al. 2025a. Kimina-prover preview: Towards large formal reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.11354*.

Wang, J.; Sundarsingh, D. S.; Deshmukh, J. V.; and Kantaros, Y. 2025b. ConformalNL2LTL: Translating Natural Language Instructions into Temporal Logic Formulas with Conformal Correctness Guarantees. ArXiv:2504.21022.

Wang, Q.; Brown, C.; Kaliszyk, C.; and Urban, J. 2020. Exploration of neural machine translation in autoformalization of mathematics in Mizar. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, 85–98.

Wang, Q.; Kaliszyk, C.; and Urban, J. 2018. First experiments with neural translation of informal to formal mathematics. In *Intelligent Computer Mathematics: 11th International Conference, CICM 2018, Hagenberg, Austria, August 13-17, 2018, Proceedings 11*, 255–270. Springer.

Wang, R.; Li, Y.; Fung, Y. R. M.; and Zhang, T. 2025c. Let's Reason Formally: Natural-Formal Hybrid Reasoning Enhances LLM's Math Capability. *arXiv preprint arXiv:2505.23703*.

Warren, D. H.; and Pereira, F. C. 1982. An Efficient Easily Adaptable System for Interpreting Natural Language Queries. *American Journal of Computational Linguistics*, 8(3-4): 110–122.

Weizenbaum, J. 1966. ELIZA — A Computer Program For the Study of Natural Language Communication Between Man and Machine. *Communications of the ACM*, 9(1): 36–45.

Weng, K.; Du, L.; Li, S.; Lu, W.; Sun, H.; Liu, H.; and Zhang, T. 2025. Autoformalization in the Era of Large Language Models: A Survey. *arXiv preprint arXiv:2505.23486*.

William, M.; Nikolaus, H.; Koenig, N.; Leyi, C.; Rothkopf, R.; Qiao, F.; and Santolucito, M. 2024. Guiding LLM Temporal Logic Generation with Explicit Separation of Data and Control. *arXiv preprint arXiv:2406.07400*.

Winograd, T. 1972. Understanding Natural Language. *Cognitive Psychology*, 3(1): 1–191.

Wu, Y.; Jiang, A. Q.; Li, W.; Rabe, M.; Staats, C.; Jamnik, M.; and Szegedy, C. 2022. Autoformalization with Large Language Models. *Advances in Neural Information Processing Systems*, 35: 32353–32368.

Xie, Y.; Yu, C.; Zhu, T.; Bai, J.; Gong, Z.; and Soh, H. 2023. Translating Natural Language to Planning Goals with Large-Language Models.

Xu, Y.; Feng, J.; and Miao, W. 2024. Learning from Failures: Translation of Natural Language Requirements into Linear Temporal Logic with Large Language Models. In *2024 IEEE 24th International Conference on Software Quality, Reliability and Security (QRS)*, 204–215. ISSN: 2693-9177.

Yang, K.; Poesia, G.; He, J.; Li, W.; Lauter, K.; Chaudhuri, S.; and Song, D. 2024. Formal mathematical reasoning: A new frontier in ai. *arXiv preprint arXiv:2412.16075*.

Yang, Z.; Ishay, A.; and Lee, J. 2023. Coupling Large Language Models with Logic Programming for Robust and General Reasoning from Text.

Ying, H.; Wu, Z.; Geng, Y.; Wang, J.; Lin, D.; and Chen, K. 2024. Lean Workbook: A large-scale Lean problem set formalized from natural language math problems.

Yu, Z.; Peng, R.; Ding, K.; Li, Y.; Peng, Z.; Liu, M.; Zhang, Y.; Yuan, Z.; Xin, H.; Huang, W.; Wen, Y.; Zhang, G.; and Liu, W. 2025. FormalMATH: Benchmarking Formal Mathematical Reasoning of Large Language Models. arXiv:2505.02735.

Zhang, L.; Quan, X.; and Freitas, A. 2024. Consistent Autoformalization for Constructing Mathematical Libraries. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Zhang, L.; Valentino, M.; and Freitas, A. 2025. Formalizing Complex Mathematical Statements with LLMs: A Study on Mathematical Definitions.

Zhou, J. P.; Staats, C.; Li, W.; Szegedy, C.; Weinberger, K. Q.; and Wu, Y. 2024. Don't Trust: Verify – Grounding LLM Quantitative Reasoning with Autoformalization.