Large Language Model for Interactive Formal Theorem Proving

Xijia Tao, 3035767762

September 2023

1 Background

In recent years, there has been significant progress in the field of deep learning, particularly in the development of large language models (LLM). These models, such as OpenAI's GPT-3, have demonstrated remarkable capabilities in natural language understanding, generation, and reasoning. They have been successfully applied to various tasks, including language translation, text summarization, and question answering.

While these language models have shown great potential, their application in the field of formal theorem proving remains relatively unexplored. Formal theorem proving involves using mathematical logic and rigorous reasoning to establish the correctness of mathematical statements or prove theorems. Traditionally, formal theorem proving has been a complex and time-consuming process, often requiring deep expertise in mathematical logic and substantial manual effort. Like games, it substantially demands planning and symbolic reasoning [2]. However, due to the extensive range of formal mathematics, any significant reasoning achievement attained within this field holds greater significance compared to similar achievements in games. In addition, these achievements could potentially have practical applications in important areas like software verification, and addressing real-world problems.

Currently, formal theorem proving is mainly performed using specialized proof assistants like Coq, Isabelle, and HOL Light [1]. These systems provide a framework for mechanically checking proofs, but they often require users to write detailed and explicit formal proofs, which can be laborious and error-prone. Moreover, the learning

Statement: Suppose n is a natural number such that n leaves a remainder of 3 when divided by 7. Show that (2n+1) leaves a remainder of 0 when divided by 7.

Subgoal-based Proof:
Step 1: Express n as $7k + 3$ for some k $\in \mathbb{N}$
Step 2: Substitute the expression for n into the expression for $2n + 1$ and simplify.
Step 3: Show that the resulting expression is divisible by 7.

Formal Sketch:

```
theorem mathd_numbertheory_370:
  fixes n :: nat
  assumes h0 : "n mod 7 = (3::nat)"
  shows "(2*n+1) mod 7 = (0::nat)"
proof -
  (* Step 1: Express n as 7k + 3 for some k ∈ ℕ. *)
  have "n = 7 * (n div 7) + (n mod 7)"
    using div_mult_mod_eq[of "n" "7"] sledgehammer
  also have "... = 7 * (n div 7) + 3"
    using h0 sledgehammer
  finally have c0: "n = 7 * (n div 7) + 3".
  (* Step 2: Substitute the expression for n into the expression for 2n + 1 and simplify. *)
  have "2*n+1 = 2*(7*(n div 7) + 3) + 1"
    using c0 sledgehammer
  also have "... = 14*(n div 7) + 7"
    sledgehammer
  also have "... = 7*(2*(n div 7) + 1)"
    sledgehammer
  finally have c1: "2*n+1 = 7*(2*(n div 7) + 1)" .
  (* Step 3: Show that the resulting expression is divisible by 7.*)
  show "(2*n+1) mod 7 = 0"
    using c1 sledgehammer
qed
```

Figure 1: Taken from [4], it shows a formal sketch of a proof in the Isabelle prover, leveraging both LLM and predefined prover tools (e.g., Sledgehammer).

curve associated with these proof assistants can be steep, making them less accessible to newcomers in the field.

By employing LLM for interactive formal theorem proving, we aim to address these challenges. The project will explore how AI techniques, combined with natural language understanding and reasoning capabilities, can provide an intuitive and user-friendly environment for theorem proving. The proposed system will potentially allow users to interact with the language model using natural language queries, enabling them to express mathematical statements, explore proof strategies, and receive real-time suggestions and feedback.

The development of such an interactive formal theorem proving system holds several advantages.

1. It has the potential to significantly reduce the time and effort required for theorem proving tasks, making it more accessible to a wider audience. This could lead to

increased productivity among mathematicians and researchers, as well as facilitate learning and collaboration in the field of formal mathematics.

2. Integrating AI into the theorem proving process can enhance the discovery and exploration of new mathematical theorems. The language model can assist users in formulating conjectures, identifying relevant theorems, and suggesting potential proof strategies, fostering creativity and innovation in mathematical research.

3. By leveraging the power of LLM, the project aims to push the boundaries of what is currently achievable in formal theorem proving. It presents an opportunity to explore the capabilities and limitations of AI in the context of rigorous mathematical reasoning, paving the way for advancements in both AI and formal mathematics.

Current approaches towards this problem show limited performance on relevant benchmarks. For example, the miniF2F benchmark [6] consists of a few hundred problem statements drawn from various high-school-level mathematics competitions and the International Mathematical Olympiad. Additionally, it includes material from high school and undergraduate mathematics courses. To date, the state-of-the-art model achieves 51.2% accuracy [5] on this benchmark, indicating that there is still large room for improvement. Furthermore, many existing methods like GPT-f [3] can only handle a maximum of around a hundred proof steps for proving a theorem. But more steps are required for more complex theorems. Hence, augmenting LLM to improve reasoning over long distances could yield substantial advantages.

2 Objective

The objective of this research-based project is to investigate and explore the potential of LLM in the domain of interactive formal theorem proving. The project aims to address the following research objectives:

1. **Analysis of existing formal theorem proving techniques** Conduct a comprehensive review of traditional formal theorem proving techniques, including automated theorem provers, proof assistants, and interactive proof systems. Analyze their strengths, limitations, and areas where LLM can potentially enhance the overall theorem proving process.

2. **Exploration of LLM capabilities**: Investigate the capabilities and limitations of understanding and generating formal mathematical statements, logical rea-

soning, and proof structures. Identify the challenges specific to integrating the models into the theorem proving context and propose potential solutions.

3. **Design and implementation of an interactive framework**: Develop a prototype framework that integrates LLM into the formal theorem proving process, allowing users to interact with the system using natural language queries and obtain coherent and accurate responses.

4. **Identification of challenges and future directions**: Identify the challenges, limitations, and potential areas for improvement in the integration of LLM into formal theorem proving. Propose future research directions to address these challenges, such as refining the training process, fine-tuning methodologies, or exploring alternative language models.

5. **Documentation of findings**: Document the research methodology, experimental setup, implementation details, and evaluation results in a comprehensive research report.

By accomplishing these research objectives, this project will contribute to the existing body of knowledge in formal theorem proving and demonstrate the potential of LLM in this domain. The outcomes of this project will provide insights, guidelines, and directions for future research efforts in integrating language models into formal theorem proving systems, paving the way for more advanced and efficient approaches in this field.

3 Methodology

1. **Data Collection and Annotation**: The first step is to gather a comprehensive dataset of formal mathematical statements, proofs, and related annotations. This dataset will serve as the training data for LLM. The collection process may involve accessing existing theorem proving libraries or databases, consulting mathematical literature, and collaborating with domain experts to ensure the dataset's quality and diversity.

2. **Preprocessing and Dataset Preparation**: Once the dataset is collected, it needs to be preprocessed and transformed into a suitable format for training the language model. This may involve cleaning the data, converting mathematical symbols into textual representations, and organizing the data into appropriate input-output pairs that align with the interactive theorem proving framework.

3. **Development of Theoretical Framework**: With an understanding the limitations of existing methods, we propose potential solutions by modifying or establishing theories for modeling the reasoning process behind theorem proving.

Knowledge from symbolic regression might be needed as an insight for designing such theoretical framework. We can also consider employ techniques from reinforcement learning to design learning objectives.

4. **Model Training and Fine-tuning**: The next step is to train our model using the prepared dataset. This involves employing techniques such as unsupervised learning and transfer learning to optimize the model's performance on formal theorem proving tasks. Fine-tuning the model on the specific dataset may be necessary to improve its understanding of mathematical concepts and reasoning. Given limited compute, we might need to skip the training process and perform in-context learning instead.

5. **Evaluation and Analysis**: Once the model is developed, rigorous evaluation is conducted to assess its effectiveness and efficiency. This evaluation involves comparing the performance of the framework with traditional theorem proving techniques, both in terms of proof generation time and accuracy. User feedback and domain expert input are collected to gauge the usability and usefulness of the framework.

   By following this project methodology, the implementation of LLM for interactive formal theorem proving will be systematically carried out, ensuring a well-structured and coherent approach to harnessing the capabilities of LLM for enhancing the theorem proving process.

4 Schedule and Milestones

Below we outline a list of expected outcomes at different time stages of our project. An estimated number of learning hours is given at the end of each outcome. We assume that approximately 50 hours will be spent on the project every month.

1. **October 2023** (done)
   - survey sub-areas in AI for science
   - survey existing methods in formal theorem proving
   - submit deliverables of Phase 1, including the detailed project plan (i.e., this document) and project web page
2. **December 2023**
   - achieve objective 1 (analysis of existing formal theorem proving techniques) - 20 hours

- achieve objective 2 (exploration of LLM capabilities) - 20 hours
- partially achieve objective 3 (design of an interactive framework) - 60 hours

3. **January 2024**
    - partially achieve objective 3 (implementation of an interactive framework) - 30 hours
    - verify our method on relevant benchmarks - 10 hours
    - give first presentation - 10 hours
    - submit deliverables of Phase 2, including the preliminary implementation and detailed interim project report - 10 hours
4. **March 2024**
    - achieve objective 3 by improving upon our initial design and implementation - 50 hours
    - conduct extensive evaluation and analysis of our method - 30 hours
5. **Early April 2024**
    - achieve objective 4 (identification of challenges and future directions) - 10 hours
    - achieve objective 5 (documentation of findings) - 40 hours
6. **Late April 2024**
    - submit deliverables of Phase 3, including the finalized tested implementation and final report - 20 hours
    - attend project exhibition - 20 hours

References

[1] John Harrison, Josef Urban, and Freek Wiedijk. History of interactive theorem proving. In *Computational Logic*, 2014.

[2] Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. Formal mathematics statement curriculum learning. *arXiv preprint arXiv:2202.01344*, 2022.

[3] Stanislas Polu and Ilya Sutskever. Generative language modeling for automated theorem proving. *arXiv preprint arXiv:2009.03393*, 2020.

[4] Xueliang Zhao, Wenda Li, and Lingpeng Kong. Decomposing the enigma: Subgoal-based demonstration learning for formal theorem proving. *arXiv preprint arXiv:2305.16366*, 2023.

[5] Chuanyang Zheng, Haiming Wang, Enze Xie, Zhengying Liu, Jiankai Sun, Huajian Xin, Jianhao Shen, Zhenguo Li, and Yu Li. Lyra: Orchestrating dual correction in automated theorem proving. *arXiv preprint arXiv:2309.15806*, 2023.

[6] Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. Minif2f: a cross-system benchmark for formal olympiad-level mathematics. *arXiv preprint arXiv:2109.00110*, 2021.