HOL4PRS: Proof Recommendation System for the HOL4 Theorem Prover

Nour Dekhil(✉), Adnan Rashid, and Sofiène Tahar

Department of Electrical and Computer Engineering
Concordia University, Montreal, QC, Canada
`{n_dekhil,rashid,tahar}@ece.concordia.ca`

Abstract. Interactive theorem provers have emerged as powerful tools for formal verification, aiding in the rigorous verification of mathematical proofs and software correctness. However, the process of constructing and manipulating proofs within these systems can be complex and labor-intensive, often requiring significant expertise and time investment. In this work, we explore the integration of deep learning techniques to assist users of interactive theorem provers by recommending proof steps, aiming to enhance their productivity and efficiency. We develop a tailored tool designed to assist users of the HOL4 theorem prover, providing expert recommendations on the best tactics to employ based on the current state of a proof using a transformer-based model.

1 Introduction

Interactive theorem proving (ITP) is a widely used formal method, which is based on developing a computer-based mathematical model of a system using higher-order logic and rigorously analyzing the properties of the underlying system using deductive reasoning. It has been widely used for developing mathematical proofs, verification of hardware, and software systems. Despite its significance, ITP faces various challenges, such as laborious proof writing, managing extensive repositories and efficiently navigating various tactics for verification tasks. Integrating deep learning assistance into ITP tools offers an opportunity to streamline the interactive proof process by providing personalized guidance and recommending optimal proof steps based on the current state of the proof.

HOL4 [1] is one of the widely used higher-order-logic (HOL) interactive theorem provers. However, the process of writing formal proofs and navigating the extensive repositories of theorems within HOL4 requires a solid understanding of logic, mathematical reasoning, and the proof development process, which is quite laborious and time-consuming. In the recent past, various artificial intelligence based efforts have been done for assisting users towards the automation of the HOL4 proof development process. For instance, tools like TacticToe [2] and TacticZero [3] have made significant contributions in aiding proof search using techniques such as reinforcement learning and k-Nearest Neighbors (k-NN), respectively. Despite these advancements, TacticToe's reliance on supervised learning may constrain its efficiency when faced with new complex theorems that require advanced reasoning. Similarly, the agent trained in TacticZero may struggle to adapt to new complex theorems, potentially leading to a decrease in performance. Deep learning techniques, such as transformer-based models have a potential to overcome the above-mentioned challenges with ITP by capturing semantic relationships and patterns in proofs. To our knowledge, these techniques have not been explored within HOL4.

In this paper we present a new tool HOL4PRS (HOL4 Proof Recommendation System) that employs advanced deep learning models, such as BERT, RoBERTa, and T5, to assist the theorem proving tasks within the HOL4 theorem prover. These advanced models have demonstrated exceptional capabilities in natural language processing, understanding intricate patterns in text data, and generating contextually relevant outputs. By integrating these sophisticated models, HOL4PRS offers users highly accurate guidance, as demonstrated by evaluation results (presented in Section 4) showing accuracies of 77.3%, 89.8%, and 93.7% for the top 3, 7, and 10 proof step recommendations, respectively. The current version of HOL4PRS is available at [4], which can be used for experimentation and further development.

The rest of the paper is structured as follows: Section 2 presents an overview of our proposed methodology, outlining the key steps and techniques employed in our approach. In Section 3, we provide the process of the dataset creation from a number of HOL4 libraries alongside the deep learning models used in HOL4PRS. Section 4 presents the experimental results and evaluation of the performance of the tactic recommendation models. In Section 5, we discuss related work. We conclude the paper in Section 6 with hints to future work directions.

2 Proposed Methodology

Figure 1 depicts a workflow for HOL4PRS. It accepts a HOL4 proof state as an input (given in the right upper half of the figure), which contains a sequence of tactics (in our case, a minimum of three) that have been applied so far in an attempt to construct a proof within the HOL4 theorem prover. Advanced analysis techniques are then employed to dissect this proof state, identifying patterns and potential strategies for further progression. Based on extensive pretraining on diverse datasets, we generate a tailored list of recommended tactics relevant to the given proof state for a theorem. These recommendations aim to assist users in their proof process, potentially enhancing efficiency and efficacy.

We initiate the development of HOL4PRS by constructing a dataset tailored to our objectives. For this purpose, we selected 6 HOL4 libraries (as given in the left upper half), through which we iterated the proof scripts (`sml` files) to extract the complete proofs of theorems and lemmas. As the process of achieving a complete proof involves various states, we identified these proof states and their relevant subsequent steps for further progression. At this stage, specific preprocessing tailored to the model comes into play, involving tasks, such as tokenizing the files, specifying the numerized vocabulary to the model, and splitting the available data into testing and training sets. With the dataset prepared for training, we fine-tune the selected models, considering hyperparameter tuning, a process that resulted in multiple trained model instances. We conclude the experiments by evaluating the trained models and selecting the one with the highest accuracy.

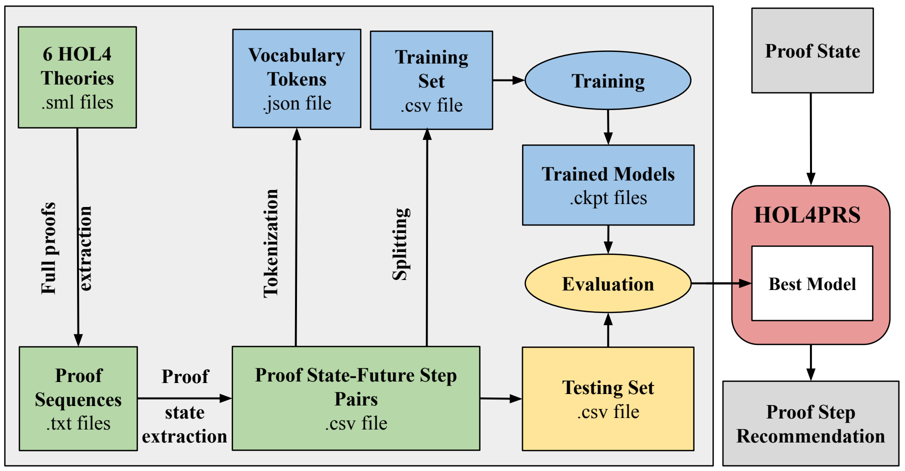

Fig. 1. Workflow Illustration for the HOL4PRS Tool.

We believe that the proposed workflow of HOL4PRS is generic enough to be applicable to a variety of other provers such as HOL Light, Coq, PVS, etc. We provide more details about these blocks of the HOL4PRS workflow in the subsequent sections.

3 Dataset and Deep Learning Models

In this work, we constructed our own HOL4 dataset from five HOL4 theories[1] developed at the Hardware Verification Group (HVG) of Concordia University, enabling us to gather a diverse corpus of proofs. The sixth theory is about real arithmetic and is available in HOL4[2]. Here, we briefly present the dataset construction process and the data prepossessing involved in HOL4PRS.

We initiated the dataset construction process by gathering all proof scripts (`sml` file) from the selected libraries. Next, for each `sml` file, we systematically list every available lemma and theorem. Subsequently, a script was executed to extract the proof steps (tactics) employed to verify each lemma and theorem. This procedure resulted in a file containing a list of proof sequences detailing all

[1] Dataset 1: https://hvg.ece.concordia.ca/projects/prob-it/pr9
Dataset 2: https://hvg.ece.concordia.ca/projects/prob-it/wsn
Dataset 3: https://hvg.ece.concordia.ca/projects/prob-it/pr10
Dataset 4: https://hvg.ece.concordia.ca/projects/prob-it/pr5
Dataset 5: https://hvg.ece.concordia.ca/projects/prob-it/pr7
[2] Dataset 6: https://dl.acm.org/doi/10.1145/3341105.3373917

proof steps employed to prove all theorems and lemmas available in the relevant library.

We aim to improve theorem proving efficiency by predicting the next proof step (tactic) based on a current proof state, which consists of a sequence of tactics already utilized in the proof. Employing language models, we approach this task as a multi-label classification problem. This classification approach acknowledges the inherent complexity and uncertainty in theorem proving tasks, where multiple tactics can lead to a successful proof. To prepare the dataset for the multi-label classification, we restructure the initial dataset composed of the complete proof sequence by generating pairs of current proof states and future tactics. Here, a proof state is an instance, where a certain number of tactics have been applied in the proof process. In our case, for each sequence of n tactics in the initial dataset that are required to prove a theorem, we created $n-4$ instances, considering sequences with a minimum history of three tactics. This enables recording all possible proof states of varying lengths and associating each state with its subsequent tactic. Additionally, instances with similar tactic histories but different future tactics were included, thereby facilitating multi-label classification. Table 1 summarizes details of our datasets from various HOL4 libraries including Dataset 7, which is a combination of all other six datasets.

Table 1. Summary of the used Datasets

	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5	Dataset 6	**Dataset 7**
Distinct Tactics	115	132	26	44	32	89	162
Proofs	1,873	2,475	153	295	61	279	5,136
Proof States	43,167	57,602	2,973	7,371	1,784	3,259	116,156

Having access to rich datasets from various HOL4 libraries, we experiment with three different transformer-based models [5]. Based on most recent work [6] on text classification and related work [7] that used T5, we choose BERT, RoBERTa, and T5. **BERT** (Bidirectional Encoder Representations from Transformers) [8] utilizes bidirectional transformers to capture contextual information from words in a sentence, aided by positional encoding. **RoBERTa** (Robustly optimized BERT approach) [9] extends BERT's architecture with dynamic masking during pre-training and longer training periods on larger data batches for improved performance. **T5** (Text-To-Text Transfer Transformer) [10], built upon the Transformer architecture, features self-attention mechanisms enabling precise weighing of word importance, facilitating seamless transfer learning by pre-training on extensive text corpora.

4 Experimental Results

We implement the selected models using the PyTorch Lightning [11] library. To optimize the training, we conducted experiments with varying batch sizes, learning rates, and weight decay parameters, aiming to minimize the training loss function. Each model undergoes training for 10 epochs and we preserve the model instance yielding the highest testing accuracy. Moreover, the dataset partitioning allocated 90% for training and 10% for testing. These experiments were executed on GPUs provided by the Digital Research Alliance of Canada[3].

Given that there are multiple potential future steps at every proof state, we recommend more than one proof step. To assess the accuracy of the recommendations, we employ the n-correctness rate as an evaluation metric, which represents the probability that a correct tactic from the testing dataset is included among the top-n recommended tactics. Here, n denotes the number of recommended tactics considered for evaluation against the correct tactic.

Table 2 presents the evaluation results for tactic recommendation across six datasets using our selected models. Notably, the table illustrates that the higher the value of n tactics considered, the higher the n-correctness rate, which is intuitive. However, for an efficient recommendation tool, it is preferable to provide fewer but more confident recommendations. Interestingly, for the same value of n and model, the correctness varies across different datasets, influenced by the quality and patterns of proofs in the dataset. Therefore, a dataset with more proofs containing repetitive patterns and a narrower space of tactics leads to better model performance. For instance, consider Datasets 3 and 6, which have a comparable number of proof states. However, Dataset 3 features a smaller number of distinct tactics. Upon comparing their results across the three models, we notice that dataset 3 exhibits higher accuracy.

To provide the models with a broader view of proof styles, we merged the six different theories of proof into a single dataset and conducted a new round of training. Our findings indicate that RoBERTa consistently outperformed other models in most scenarios. RoBERTa demonstrated exceptional adaptability and generalization despite potential variations in the nature of proofs from different theories. In comparison to related works, where accuracies range from 50%-70% for the top 3-5 recommendations in [7], 87% for the top 3 in [12], and 54.3% for the top 10 in [13], our model exhibits a notable improvement. Achieving accuracies of 77.3%, 89.88%, and 93.7% for the top 3, 7, and 10 recommendations, respectively. Our model surpasses most related work, marking a significant impression in recommending correct tactics for proof generation in HOL4.

Table 2. Performance Evaluation of Tactic Recommendation Models

	T5			**BERT**			**RoBERTa**		
Datasets	**Top 3**	**Top 7**	**Top 10**	**Top 3**	**Top 7**	**Top 10**	**Top 3**	**Top 7**	**Top 10**
Dateset 1	51.3%	68.7%	76.4%	52.7%	71.9%	79.9%	54.5%	73.6%	93.7%
Dateset 2	60.4%	75.5%	80.5%	60.5%	78.9%	86.3%	59.7%	79.5%	85.8%
Dateset 3	69.8%	93.4%	95.4%	76.1%	93.9%	97%	78.4%	94.4%	97.5%
Dateset 4	77.3%	95.3%	97.2%	87.3%	97.0%	98.5%	89.5%	97.8%	98.8%
Dateset 5	76.6%	97.6%	98.2%	76.6%	97.6%	98.2%	76.6%	97.6%	97.6%
Dateset 6	39.9%	55.2%	61.9%	45.1%	65.4%	72.7%	43.4%	64.3%	73.8%
Dataset 7	**72.9%**	**85.6%**	**87.8%**	**75.4%**	**88.7%**	**92.3%**	**77.3%**	**89.8%**	**93.7%**

[3] https://alliancecan.ca

5 Related Work

In recent years, significant research efforts have been done to employ artificial intelligence to streamline the process of verification using HOL theorem provers. The objective of aiding practitioners in writing proofs in theorem provers has been approached through various avenues, including proof step generation, premise selection, and proof search.

Proof Step Generation involves providing guidance on the most suitable proof steps or tactics to use within a specific proof context. For instance, Gauthier et al. [2] developed a tool TacticToe that uses k-NN to predict the most suitable HOL4 tactics for each proof situation, achieving an overall success rate of 66.4% on 7,164 theorems of the HOL4 standard library. In a somehow similar work for HOL4, Wu et al. [3] developed a tool, called TacticZero, that uses reinforcement learning for predicting most suitable tactics in the proof process of a theorem from scratch, the authors used a dataset of 1,342 theorem for training and testing. In [12], Blaauwbroek et al. introduced a tactic proof search framework for the Coq theorem prover that leverages machine learning to identify a correct tactic 23.4% of the time for a given proof state. Similarly, Luan et al. [13] utilized the Long Short-term Memory (LSTM) model to automate the proof process in Coq by predicting the next tactics, and achieved an impressive 87% accuracy in predicting the top 3 tactics. More recently, Yeh et al. [7] developed CoProver, a system based on transformers, enabling learning from historical proof steps to predict the next steps in the PVS theorem prover, with top 3/5 recommended proof steps yielding accuracies of 50%/70% on the validation set.

Premise Selection focuses on identifying useful theorems that are crucial for successful proof progression. For example, Alemi et al. [14] proposed a tool to predict the most relevant premises in the E Prover using Convolutional Neural Networks (CNN), achieving success rates ranging from 78.4% to 80.9%. Similarly, Kaliszyk et al. [15] employed K-NN and Naive Bayes to select the most relevant premises (theorems) likely to aid in proving mathematical conjectures within the Flyspeck project in the HOL Light theorem prover. The authors presented the HOL(y)Hammer system that has been able to achieve a success rate of 39% in proving 14185 theorems in the same project. This work was followed by an adaptation of the HOL(y)Hammer system for HOL4 [16], providing premise selection and automated reasoning capabilities to enhance the proof process in HOL4 using k-NN.

Proof Search aims to learn from existing proofs to generate potential proof paths for given theorems. For instance, First et al. [17] utilized beam search to automatically generate complete Coq proof scripts for proving the target theorem. The authors have been able to successfully prove 12.9% of the testing theorems. Also the tool TacticToe [2] combines Monte Carlo tree search and supervised learning algorithms to guide proof search, resulting in a high success rate for proving HOL4 theorems. Finally, the tool TacticZero [3] uses a reinforcement learning environment to guide an agent in predicting complete proofs by navigating through a series of tactic applications and proof steps towards the final theorem.

In this paper, we used more recent deep learning models to generate proof step recommendations for HOL4 proofs. Our research builds on the significant efforts made in related work by investigating the potential of transformer-based models, which are renowned for their capability to discern patterns in contextual data. We trained these models on project-specific proof scripts authored by a single individuals, as these scripts often exhibit higher levels of repetitiveness and distinct patterns. This approach contrasts with prior studies on HOL4, which have not explored the use of transformers or application-specific proofs, typically relying on the HOL4 standard library.

By incorporating transformer models, our objective is to enhance the precision and effectiveness of recommending proof steps for the HOL4 theorem prover. The selection of transformer-based models was influenced by their ability to effectively capture long-range dependencies in sequences, which is crucial for understanding the context and patterns in sequences of tactics used in theorem proving. We conducted experiments with models such as BERT, RoBERTa, and T5 to determine the most suitable model for our datasets. After evaluation, RoBERTa demonstrated strong performance in predicting the next tactic in the Proof Recommendation System for the HOL4 Theorem Prover, achieving correctness rates ranging from 64.3% to 97.8% across various datasets.

6 Conclusion

HOL4PRS introduces a novel tool tailored to enhance the HOL4 theorem proving through the integration of a RoBERTa model. Specifically, the tool operates by receiving a sequence of tactics (minimum three in our case) already employed in the proof process, which we refer to as the current proof state, and providing recommendations for the optimal next proof step. The RoBERTa model employed in this context builds upon the Transformer architecture which is renowned for its advanced contextual understanding and proficient text generation capabilities. Future work could focus on expanding HOL4PRS to include more HOL4 theories and enhancing its interfacing with HOL4. Furthermore, investigating the HOL4PRS potential to autonomously generate complete proofs could streamline the theorem proving process.

References

1. HOL4. `https://hol-theorem-prover.org/`, 2024.
2. Thibault Gauthier, Cezary Kaliszyk, Josef Urban, Ramana Kumar, and Michael Norrish. Tactictoe: Learning to Prove with Tactics. *Journal of Automated Reasoning*, 65:257–286, 2021.
3. Minchao Wu, Michael Norrish, Christian J. Walder, and Amir Dezfouli. Tacticzero: Learning to Prove Theorems from Scratch with Deep Reinforcement Learning. *Advances in Neural Information Processing Systems*, 34:9330–9342, 2021.
4. HOL4PRS: Proof Recommendation System for the HOL4 Theorem Prover. `https://github.com/DkNour/HOL4PRS-Proof-Recommendation-System-for-the-HOL4-Theorem-Prover.git`.

5. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. *Advances in neural information processing systems*, 30:6000–6010, 2017.
6. Victor Kwaku Agbesi, Wenyu Chen, Sophyani Banaamwini Yussif, Md Altab Hossin, Chiagoziem C. Ukwuoma, Noble A. Kuadey, Colin Collinson Agbesi, Nagwan Abdel Samee, Mona M. Jamjoom, and Mugahed A. Al-antari. Pre-trained Transformer-based Models for Text Classification Using Low-resourced Ewe Language. *Systems*, 12(1), 2024.
7. Eric Yeh, Briland Hitaj, Sam Owre, Maena Quemener, and Natarajan Shankar. CoProver: A Recommender System for Proof Construction. In *Intelligent Computer Mathematics*, volume 14101 of *LNAI*, pages 237–251. Springer, 2023.
8. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *North American Chapter of the Association for Computational Linguistics*, pages 4171–4186. Association for Computational Linguistics, 2019.
9. Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692, 2019.
10. Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. volume 21, pages 1–67, 2019.
11. PyTorch Lightning. `https://lightning.ai/docs/pytorch/`, 2024.
12. Lasse Blaauwbroek, Josef Urban, and Herman Geuvers. Tactic Learning and Proving for the Coq Proof Assistant. *arXiv preprint arXiv:2003.09140*, 2020.
13. Xiaokun Luan, Xiyue Zhang, and Meng Sun. Using LSTM to Predict Tactics in Coq. In *Software Engineering and Knowledge Engineering*, pages 132–137, 2021.
14. Alexander A. Alemi, François Chollet, Niklas Een, Geoffrey Irving, Christian Szegedy, and Josef Urban. Deepmath - Deep Sequence Models for Premise Selection. In *International Conference on Neural Information Processing Systems*, page 2243–2251. Curran Associates Inc., 2016.
15. Cezary Kaliszyk and Josef Urban. Learning-Assisted Automated Reasoning with Flyspeck. *Journal of Automated Reasoning*, 53:173–213, 2012.
16. Thibault Gauthier and Cezary Kaliszyk. Premise Selection and External Provers for HOL4. In *Certified Programs and Proofs*, page 49–57. ACM, 2015.
17. Emily First, Yuriy Brun, and Arjun Guha. Tactok: Semantics-Aware Proof Synthesis. *Proceedings of the ACM on Programming Languages.*, 4:1–31, 2020.