X: A Modular Environment for Driving Insights in Automated Interactive Theorem-Proving

Anonymous submission

Abstract

Recent interest in Artificial Intelligence for Theorem Proving (AITP) has given rise to a plethora of benchmarks and methodologies, particularly in the area of Interactive Theorem Proving (ITP). Research in the area is fragmented, with a diverse set of approaches being spread across several ITP systems. This presents a significant challenge to the comparison of methods, which are often complex and difficult to replicate. To address this, we present X.. By separating the learning approach from the data and ITP environment, X allows for the fair and efficient comparison of approaches between systems. X is designed to seamlessly incorporate new systems and benchmarks, currently integrating the HOList, HOLStep, MIZAR40, LeanStep, LeanGym and TacticZero benchmarks. We demonstrate the utility of X through a study of embedding architectures in the context of the above systems. These experiments lay the foundations for future work in evaluation of performance of formula embedding, while simultaneously demonstrating the capability of the framework. We complement these with a qualitative analysis of embeddings within and between systems, illustrating that improved performance was associated with more semantically aware embeddings. By streamlining the implementation and comparison of Machine Learning algorithms in the ITP context, we anticipate X will be a springboard for future research.

Introduction

Interactive Theorem Proving (ITP) is a central paradigm of formal verification. With a human providing high level proof guidance, ITP systems have been used to write verified compilers (cite), formalise mathematical conjectures (cite) and develop provably correct microkernels (cite). As proficiency in both the formal system and application domain is needed, large verification projects require significant human resources and expertise. This restricts their scalability and widespread adoption, with for example xx taking xx amount of years. Advancements in Artificial Intelligence for Interactive Theorem Proving (AI-ITP) have shown potential in automating and assisting human ITP guidance, but there remain many challenges in the area. With the current state of the art achieving 42% accuracy on the (highly difficult) miniF2F-curriculum benchmark (cite), there is still much progress to be made.

To that end, it is important to efficiently and fairly compare approaches, which is difficult as results are fragmented across multiple ITP systems. Current results are evaluated either with labelled datasets extracted from proof data, as in HOLStep, LeanStep and MIZAR40, or through direct interaction with a proving environment. Environments for agent interaction have been shown to be vital in improving performance (cite), as they allow for the generation of new data through automated proof attempts. Such environments include HOList (Bansal et al. 2019a) for HOL Light, CoqGym for Coq (cite) LeanGym for Lean (cite), and INT as an AI-ITP specific (in)equality proving system. These provide a broad set of tasks for benchmarking, however as they are all isolated to a single system it is difficult to compare approaches between them. This is further complicated by the variety and complexity of algorithms used, which can vary over several axes. For example, TacticZero (Wu et al. 2021) uses a seq2seq autoencoder for expressions, and learns through online Reinforcement Learning (RL) with a custom goal selection algorithm. (Bansal et al. 2019a) instead use Breadth First Search (BFS) for goal selection, with offline learning over human and synthetic proof logs.

Despite this, many fundamental components are common across AI-ITP systems. We leverage these to develop a modular, unified framework for accelerating research in the area. XXX brings together several environments, datasets and models from AI-ITP, with a central interface for running experiments and sharing components between systems. Being fully open source, the addition of new benchmarks and algorithms is facilitated by a modular and decoupled design. We combine this with automatic logging, checkpointing and configuration management to allow for rapid testing and prototyping of ideas with minimal overhead.

A large focus of this paper is using xx to perform a comparison of embedding architectures in AI-ITP. ITP expressions are either treated as a natural language sequence, or as a directed graph derived from their abstract syntax tree representation. It has been argued that a graph representation is more appropriate, with a large body of work in AI-ITP using Graph Neural Network (GNN) as the embedding model to achieve strong results in several tasks(Wang et al.; Paliwal et al. 2019; Bansal et al. 2019b). However, Transformer models applied to the sequence representation have also demonstrated strong performance through leveraging large models pre-trained on NLP datasets (cite). Both architectures have fundamental limitations, as GNNs suffer from

poor integration of global information and Transformers are trained without the structural information in graphs. The INT environment, which we include, was used to compare these approaches in a synthetic setting, however we extend this to a comparison across a suite of tasks in various ITPs. We also compare the recent SAT (cite) architecture which combines both approaches, and show ... Additionally, we provide a qualitative analysis supporting the intuition that better performing models generate more semantically aware embeddings.

Related Work

The very recent LeanDojo (cite) is similar to this work, in that they provide an open source environment which is replicable and with reasonable resource requirements. As with other current frameworks in AI-ITP, it is focused on a single proving system. Our approach is complementary to this, and builds upon the extensive work done developing these benchmarks by creating a unified cross platform framework for interfacing with them.

Similar unified frameworks have been across areas such as NLP, CV, GNN (graph benchmarks), which have provided strong value to their respective areas

Closely related to our work are MWPToolkit and LImA? for Math Word Problems (MWP), another area of AI for Mathematical Reasoning. These similarly bring together datasets and approaches across the area into a single framework.

There are additional considerations in our case:

- Environment interaction makes it difficult to be fully decouple data and model. (e.g. HOL4 environment for TacticZero)
- Several axes of variation in algorithm design. Learning paradigm, proof search, embedding arch., action selection arch, whereas MWP focuses primarily on NLP pipelines

Although we focus on automating the human interaction with ITP, other approaches such as ML for Hammers (cite) have also provided promising directions for automated and assisting ITP proofs.

Background

Datasets and Benchmarks

Figure x provides a summary of current datasets and benchmarks related to AI-ITP. Fields marked as non-interactive represent labelled datasets derived from proofs in ITP systems, an A common task in the literature is predicting whether a premise was useful in the proof of a given conjecture.

HOLStep (Kaliszyk, Chollet, and Szegedy 2017), MIZAR40 (Kaliszyk and Urban 2015),.. provide datasets around this (ISARstep ..., look at INT paper intro)

LeanStep for others

Figure (Table): Benchmark vs Details (size, system, interactive) Highlight in bold what is currently included Include HOL4 premise selection HOList, LeanStep, HOLStep, MIZAR40, LeanGym, TacticZero, (CoqGym, IsarStep etc from survey)

	System	Interactive	Included	Task
HOList	HOL Light	Y	Y	a
HOL4 MDP	HOL4	Y	Y	a
LeanStep	Lean	N	Y	a
LeanGym	Lean	Y	Y	a
HOLStep	HOL Light	N	Y	a
CoqGym	Coq	Y	N	a
GamePad	MML?	Y	N	a
MIZAR40	MML	N	Y	a
IsarStep	Isar	N	N	a
INT	INT	Y	Y	a
LISA	Isabelle	Y	N	a
LeanDojo	Lean	Y	N	a

Table 1: Current Benchmarks in AI-ITP, including interactive environments and those integrated in X.

As outlined in Figure x, benchmarks consider each system in isolation, and generally focus on a small set of learning and proof search approaches. Given the many differences between ITPs, this is unsurprising, as there is significant effort required to set up these systems for automated proving.

AI-ITP

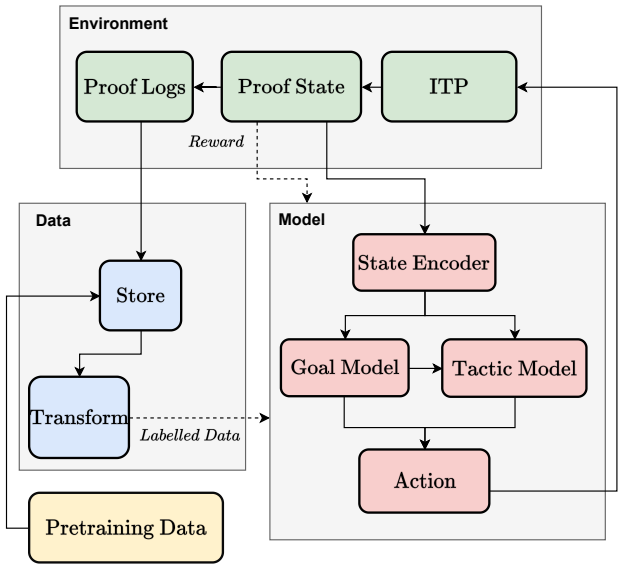

Figure 1: General Framework for AI-ITP approaches. Dashed lines represent different learning approaches, specifically Reinforcement Learning and Supervised Learning. XXX implements this through highly decoupled Data, Model and Environment modules, simplifying development by enabling the reuse of shared components.

Problem Setup The AI-ITP problem is fundamentally a sequential decision process (cite). The objective in each round is to prove an initial theorem, defined as the goal g. The initial state $(s \supseteq g) \in \mathcal{S}$ includes g along with auxiliary information such as allowed premises and axioms. The

agent π is a mapping from the state to an action $a \in \mathcal{A}$, i.e. $\pi : \mathcal{S} \rightarrow \mathcal{A}$. The environment $\mathcal{E} : \mathcal{A} \rightarrow \mathcal{S}$ is a superset of the ITP system, taking an action a to a new state $s' \in \mathcal{S}$. s' is either a terminal state confirming the proof of the original goal, or represents a new state with subgoal(s) equivalent to proving g. We assume that s' includes all previous states so that the agent has a full view of the history. $\mathcal{E}$ is effectively a wrapper over the ITP environment, and can include reward signals for reinforcement learning. The wrapper may also keep track of previous states such that s' represents a *proof tree*, representing all current paths to proving the original goal. In practice, there are many choices for the learning system π, which we outline below.

Learning Approach The learning paradigm is an important consideration in the design of an AI-ITP system, representing how the parameters of π are updated. The simplest approach is direct supervised learning over proof logs with gradient descent. As shown in 1, these logs are either collected from human proofs or collected as part of the interaction of the AI system with the ITP environment. For the HOList benchmark, (Bansal et al. 2019b) demonstrate strong results with supervised training over human and synthetic logs, improving over (Paliwal et al. 2019) which used only human proofs. Both approaches train a model to rank premises and tactics based on log information, which is then used to guide ITP interaction.

TacticZero (Wu et al. 2021) learns only using synthetic proof data. In this case, learning is done through Reinforcement Learning with Policy Gradient, with the agent replaying previously recorded proof traces. Compared to supervised learning approaches, RL has the advantage of removing biases in the labelling of data. For example, it is common to assign negative labels to premises not used in an original human proof (Kaliszyk, Chollet, and Szegedy 2017; Kaliszyk and Urban 2015; Bansal et al. 2019a) despite the potential for these premises to be useful in a different proof of the same conjecture. However, RL (in particular Policy Gradient) is known to suffer from high variance in the gradient updates (CITE) It remains unclear which approach is superior in AI-ITP, further motivating a centralised comparison platform.

Recent advancements in NLP have led to more recent work applying pretrained language models in the context of AI-ITP. HyperTree, LeanStep, GPT-f, Curriculum learning Pre-training over proof data (PACT) Pre-training over NLP corpus

State Embedding The state embedding model $E : \mathcal{S} \rightarrow \mathbb{R}^D$ maps the current proof state s' into Euclidean space, with $D \in \mathbb{N}^k$ depending upon the approach. An effective embedding model is critical, as it is the only information used by the tactic and goal selection models. Approaches such as (cite) curriculum and (cite hypertree) use Transformer Encoders, treating the current goal $g \subseteq s'$ as a Natural Language sequence. The final embedding in this case are $E(g) \in \mathbb{R}^{n \times k \times d}$ where n, k, d are the number of goals, tokens and embedding dimension respectively. Other approaches produce a single vector for each goal, such that $E(g) \in \mathbb{R}^{n \times d}$. Architectures used in this case include GNNs (Bansal et al. 2019b; Paliwal et al. 2019; Wang et al.), Tree LSTMs (cite) and seq2seq autoencoders (Wu et al. 2021), ..

Proof Search Proof search follows state embedding, and is represented by the goal model in Figure 1. It is the subpolicy $\pi(g|s')$ mapping from the complete (encoded) environment state $s' \in \mathbb{R}^{n \times D}$ to a subset of goals $g \subseteq s'$. Many approaches in the literature, including GamePad(cite), (Bansal et al. 2019b), (cite curriculum) apply Breadth First Search (BFS) for goal selection. Best first search based on a learned goal score was used in .. TacticZero (Wu et al. 2021) showed an improvement over these by considering the likelihood of proving distinct sets of goals (fringes) equivalent to the original proof. Improvements have also been found in (cite hypertree) through an adapted MCTS algorithm. Other approaches..?

Tactic Selection Tactic selection maps the selected goals to an action, with $\pi(t|g) : \mathbb{R}^{n \times D} \rightarrow \mathcal{T}$. ITP tactics themselves are usually a small, fixed size list. As they also include arguments, which can be arbitrary expressions, tactic selection is a difficult problem. It is common to restrict the argument space to only include other theorems, lemmas or terms available in the current context. In these cases, the tactic and argument models consist of predicting the tactic to apply, followed by predicting the arguments conditioned on the tactic. Despite the much larger action space, generative models such as Transformer Decoders have also been successful in predicting the entire tactic and argument as a sequence of text (cite)

Embedding Architecture

GNN

Transformer

SAT

XXX (framework)

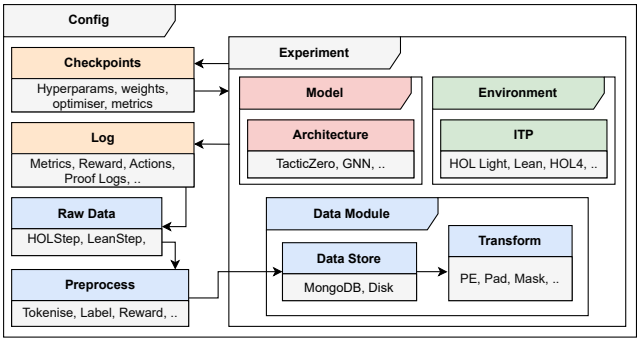

Figure 2: Design of X.

Diagram and explanation of system architecture,

Open Source additions: LeanGym/LeanStep output s-expressions

HOList using non-google tools, and independent of TF1 and ML framework

Curated, large dataset of varied ITPs

	SAT	GNN	Transformer	Ensemble
HOL4 Premise Selection	91%	? 91.8%	?	?
HOList Tactic Accuracy	X%	?	?	?
HOList Rel. Param Acc	X%	N	?	?
HOList TopK Acc	X%	?	?	?
HOLStep	91.5%	91%	8?%	?
INT	?	?	?	?
MIZAR40	?	?	?	?

Table 2: Results for Embedding architectures across Pretraining Tasks

Modular framework
HOL4 Pretraining Task

Embedding Experiments

Supervised

MIZAR, HOLStep, HOL4 pretrain, HOList Pretrain, Lean?

End to End

TacticZero, HOList?

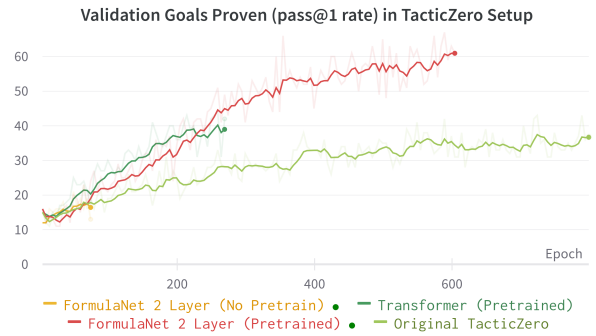

Figure 3: Unseen goals proved in one attempt (pass@1 rate) for different encoder architectures on the TacticZero benchmark.

Qualitative Analysis

Discussion

Conclusion

References

Bansal, K.; Loos, S. M.; Rabe, M. N.; Szegedy, C.; and Wilcox, S. 2019a. HOList: An Environment for Machine Learning of Higher-Order Theorem Proving. ArXiv:1904.03241 [cs].

Bansal, K.; Szegedy, C.; Rabe, M. N.; Loos, S. M.; and Toman, V. 2019b. Learning to Reason in Large Theories without Imitation. Publisher: arXiv Version Number: 3.

Kaliszyk, C.; Chollet, F.; and Szegedy, C. 2017. HolStep: A Machine Learning Dataset for Higher-order Logic Theorem Proving. *ArXiv*.

Kaliszyk, C.; and Urban, J. 2015. MizAR 40 for Mizar 40. *Journal of Automated Reasoning*, 55(3): 245–256.

Paliwal, A.; Loos, S.; Rabe, M.; Bansal, K.; and Szegedy, C. 2019. Graph Representations for Higher-Order Logic and Theorem Proving. ArXiv:1905.10006 [cs, stat].

Wang, M.; Tang, Y.; Wang, J.; and Deng, J. ???? Premise Selection for Theorem Proving by Deep Graph Embedding.

Wu, M.; Norrish, M.; Walder, C.; and Dezfouli, A. 2021. TacticZero: Learning to Prove Theorems from Scratch with Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 34, 9330–9342. Curran Associates, Inc.