Formal Premise Selection With Language Models

Szymon Tworkowski*[1], Maciej Mikuła*[1], Tomasz Odrzygóźdź*[5], Konrad Czechowski*[1], Szymon Antoniak*[1], Albert Q. Jiang[3], Christian Szegedy[2], Łukasz Kuciński[4], Piotr Miłoś[4], Yuhuai Wu[2]

[1] University of Warsaw, [2] Google Research
[3] University of Cambridge, [4] Polish Academy of Sciences, [5] IDEAS NCBR

Abstract

Premise selection, the problem of selecting a useful premise to prove a new theorem, is an essential part of theorem proving. Existing language models cannot access knowledge beyond a small context window, and therefore are unsatisfactory at retrieving useful premises (i.e., premise selection) from large databases for theorem proving. In this work, we provide a solution to this problem, by combining a premise selection model with a language model. We first select a handful (e.g., 8) of premises from a large theorem database consisting of $100K$ premises, and present them in the context along with proof states. The language model then utilizes these premises to construct a proof step. We show that this retrieval-augmented prover achieves significant improvements in proof rates compared to the language model alone.

1 Introduction

Language models have been recently applied to theorem proving [17, 25, 12, 14, 24] and program synthesis [7, 2, 20], achieving impressive results. **Premise selection** is a fundamental aspect of formal mathematics [28, 1, 4]. Early works in this domain often relied on symbolic [19, 5] or hybrid [15] approaches. Classical ML algorithms [30, 29, 9] have also proven effective, frequently outperforming symbolic methods by significant margins. More recently, graph neural networks mimicking the symbolic structure of mathematical expressions have shown promising results [22, 33, 10, 18].

Effective retrieval of premises from large databases is still an open challenge. In this work, we propose to approach it with a two-stage procedure, which, to the best of our knowledge, is the first method to do the selection process globally over the whole corpus. Firstly, a premise selection model (PSM) picks a handful (e.g. 8) of premises from a database. These are then presented, along with a proof state, to a premise selection guided language model (PGLM) responsible for generating a proof step. Importantly, our PSM can efficiently query large databases; in our case, we use over 100K lemmas from the entire Isabelle corpus. By providing a relatively small number of premises in context, we allow the PGLM to efficiently retrieve the correct ones, aiming to leverage its in-context learning capabilities [16].

2 Method

Premise selection model (PSM) is based on a batch contrastive learning approach similar to [1, 3, 26, 11]. It encodes proof state and premise text into embeddings. The cosine similarity of a given premise embedding and a proof state embedding estimates their mutual relevance. Premise embeddings can be precomputed and cached, allowing for the use of large databases.

*Equal Contribution

Premise-guided language model (PGLM) is a model for proof step generation. It takes as an input the *current proof state* s and *premises* (names and statements) selected by PSM. These are e.g. $k = 8$ premises from the whole database with the highest relevance to s.

We first train the PSM, then freeze the weights and use it in the training process of PGLM. The PGLM is designed to perform *premise-aware* proof step generation. By design, given the (small) context of k premises, the model selects the relevant ones to be applied in the generated proof step. This setup is motivated by recent findings [16] showing that LMs can grasp dependencies in the text within the same input much better than ones occurring across different training examples. The latter is how the state-only (our baseline model, described below) approach works. We hope that in-context learning helps the model focus on premise selection instead of memorization of frequently-occurring premises (as we hypothesize the state-only models do).

The **State-only model** is a language model that, given a proof state (goal), predicts the proof step. This is the most common setup found in prior work [14, 12, 24], used here as a baseline.

3 Experiments

We conduct our interactive theorem proving experiments on a dataset collected in Isabelle [23] which is one of the largest corpora of formal proofs. To interact with the formal environment, we use PISA [14]. The proof rates are presented in the table below.

Method	Proof rate, full	Proof rate, ≥ 1 premise	Proof rate, 0 premises
Sledgehammer [5] (baseline)	22.4%	17.7%	27.5%
State-only (baseline)	39.8%	14.7%	67.1%
PGLM+PSM (ours)	**43.1%**	**19.6%**	**68.6%**
PGLM+PSM $\cup$ *State-only*	**47.2%**	**22.6%**	**73.9%**

Table 1: Proof rate is evaluated using a best-first search solver, similar to the one mentioned in [14], on a test set of 1000 theorems. We split the test dataset into proofs originally using and not using premises; denoted ≥ 1 premise and 0 premises, respectively. For the sledgehammer baseline we use 50s timeout per proof.

Our method, *PGLM+PSM*, performs significantly better on theorems that require at least one premise and fares well on the entire test set. This indicates that the proposed two-stage method is efficient in premise retrieval. Furthermore, a significant improvement is observed when *PGLM+PSM* and the state-only model are combined. This is especially visible on the full test set, indicating that both methods have complementary strengths.

4 Conclusion and future work

We present a simple method integrating premise selection with language models, which is guided by an external retriever model. We show proof rate improvements when compared to a state-only baseline and demonstrate that our model is capable of generating novel proofs that utilise premises.

We speculate that scaling up our approach will further increase its capabilities. In particular, we hypothesise that in-context premise selection performance will improve due to better generalisation to unseen premises. If true, it would indicate better reasoning potential of the underlying language model, and as such is an attractive research direction.

References

[1] Alex A. Alemi, Francois Chollet, Niklas Een, Geoffrey Irving, Christian Szegedy, and Josef Urban. Deepmath - deep sequence models for premise selection, 2016.

[2] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021.

[3] Kshitij Bansal, Sarah M. Loos, Markus N. Rabe, Christian Szegedy, and Stewart Wilcox. Holist: An environment for machine learning of higher order logic theorem proving. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 454–463. PMLR, 2019.

[4] Kshitij Bansal, Christian Szegedy, Markus N. Rabe, Sarah M. Loos, and Viktor Toman. Learning to reason in large theories without imitation, 2019.

[5] Sascha Böhme and Tobias Nipkow. Sledgehammer: judgement day. In *International Joint Conference on Automated Reasoning*, pages 107–121. Springer, 2010.

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[7] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

[8] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling, 2021.

[9] Thibault Gauthier and Cezary Kaliszyk. Premise selection and external provers for hol4. In *Proceedings of the 2015 Conference on Certified Programs and Proofs*, CPP '15, page 49–57, New York, NY, USA, 2015. Association for Computing Machinery.

[10] Zarathustra A. Goertzel, Jan Jakubův, Cezary Kaliszyk, Miroslav Olšák, Jelle Piepenbrock, and Josef Urban. The isabelle enigma, 2022.

[11] Jesse Han, Tao Xu, Stanislas Polu, Arvind Neelakantan, and Alec Radford. Contrastive finetuning of generative language models for informal premise selection. *6th Conference on Artificial Intelligence and Theorem Proving*, 2021.

[12] Jesse Michael Han, Jason Rute, Yuhuai Wu, Edward W. Ayers, and Stanislas Polu. Proof artifact co-training for theorem proving with language models. ICLR, 2022.

[13] Mauro Jaskelioff and Stephan Merz. Proving the correctness of disk paxos. *Archive of Formal Proofs*, June 2005. http://isa-afp.org/entries/DiskPaxos.html, Formal proof development.

[14] Albert Qiaochu Jiang, Wenda Li, Jesse Michael Han, and Yuhuai Wu. Lisa: Language models of isabelle proofs. *6th Conference on Artificial Intelligence and Theorem Proving*, 2021.

[15] Daniel Kühlwein, Jasmin Christian Blanchette, Cezary Kaliszyk, and Josef Urban. Mash: Machine learning for sledgehammer. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *Interactive Theorem Proving*, pages 35–50, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[16] Yoav Levine, Noam Wies, Daniel Jannai, Dan Navon, Yedid Hoshen, and Amnon Shashua. The inductive bias of in-context learning: Rethinking pretraining example design. In *International Conference on Learning Representations*, 2022.

[17] Wenda Li, Lei Yu, Yuhuai Wu, and Lawrence C. Paulson. Isarstep: a benchmark for high-level mathematical reasoning. In *International Conference on Learning Representations*, 2021.

[18] Zhaoyu Li, Binghong Chen, and Xujie Si. Graph contrastive pre-training for effective theorem reasoning, 2021.

[19] Jia Meng and Lawrence Paulson. Lightweight relevance filtering for machine-generated resolution problems. *J. Applied Logic*, 7:41–57, 03 2009.

[20] Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models. In *Deep Learning for Code Workshop*, 2022.

[21] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2018.

[22] Aditya Paliwal, Sarah Loos, Markus Rabe, Kshitij Bansal, and Christian Szegedy. Graph representations for higher-order logic and theorem proving, 2019.

[23] Lawrence C. Paulson. Isabelle: The next 700 theorem provers. 1993.

[24] Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. Formal mathematics statement curriculum learning, 2022.

[25] Stanislas Polu and Ilya Sutskever. Generative language modeling for automated theorem proving, 2020.

[26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[27] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[28] Christian Szegedy. A promising path towards autoformalization and general artificial intelligence. In Christoph Benzmüller and Bruce Miller, editors, *Intelligent Computer Mathematics*, pages 3–20, Cham, 2020. Springer International Publishing.

[29] Agnieszka Słowik, Chaitanya Mangla, Mateja Jamnik, Sean B. Holden, and Lawrence C. Paulson. Bayesian optimisation with gaussian processes for premise selection, 2019.

[30] Josef Urban, Geoff Sutcliffe, Petr Pudlák, and Jiří Vyskočil. Malarea sg1 - machine learner for automated reasoning with semantic guidance. In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *Automated Reasoning*, pages 441–456, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[32] Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax, May 2021.

[33] Mingzhe Wang, Yihe Tang, Jian Wang, and Jia Deng. Premise selection for theorem proving by deep graph embedding, 2017.

A Experimental setup

A.1 LM setup

For language modeling, we use a decoder-only transformer [31] with 30M non-embedding parameters. The setup (weight initialization, positional embeddings, and other architectural hyperparameters) is exactly the same as in GPT-J [32]. We use a pretrained BPE tokenizer from [27]. Similarly to GPT-f [25], the loss function is calculated only on the proof step tokens. As a context for proof step generation we use one sentence representing the proof state for state only model, and premises + proof state sentence for PGLM+PSM setup.

For the *PGLM+PSM* model, in our main result, we provide it with top $k = 4$ premises from the premise selection model.

All of the models are pretrained on The Pile [8] - GitHub + arXiv dataset for 500k steps with context length of 2048 as in [6] and total batch size of 2^{17} tokens per update.

A.2 PSM setup

We modify the InfoNCE [21] loss by only using row-wise softmax (column-wise softmax is ablated). Batch size of 512 proof states is used. We also randomly sample 1536 additional negative premises within a batch (512 proof states and 2048 premises in each batch, for each proof state there is exactly one positive premise and 2047 negatives), and we find it helpful to the score (see Tab. 2). We use a non-pretrained, 6-layer decoder-only transformer (15M non-embedding parameters).

B Dataset and Environment

Isabelle [23] is an interactive theorem prover (ITP). It allows mathematical formulas to be expressed in a formal language and provides tools for proving those formulas, which are verified by a logical kernel. Its main application is the formalization of mathematical proofs and in particular formal verification, which includes proving the correctness of computer hardware or software and proving properties of computer languages and protocols. Each Isabelle library is composed of theories. A proof for a given theorem is a sequence of **proof steps**, with each step being a proof tactic or part of a declaration. Each subsequent proof step changes the state (referred to as **proof state**) of the current proof. A proof step can make use of **premises**, which are simply references to definitions, axioms, or previously proven theorems. This theorem proving setting constitutes a Partially Observable Decision Process and thus can be represented by a sequential decision process in a certain environment. An example of such an environment is the PISA environment [14], which we used for all the experiments. We trained models using a dataset mined from the Archive of Formal Proofs (AFP)[13] and all the standard libraries available in Isabelle. The dataset consists of 220K lemmas, with a total of 2.4M (proof state, proof step) pairs. For the premise selection task, we chose the proof steps that utilised at least one premise, which resulted in 400K training examples.

C Premise selection - ablation study

We investigate what contributes to the performance of our retrieval model by reducing its expressive power to a 1-layer transformer (first experiment), as well as removing our negative sampling strategy (second experiment). We observe a significant drop in recall with the changes.

Model	recall@1	recall@4	recall@8	recall@16	recall@64	recall@128
1L transformer	0.168	0.347	0.447	0.565	0.663	0.809
6L transformer	0.203	0.408	0.516	0.621	0.781	0.832
6L transformer + neg.	**0.230**	**0.446**	**0.561**	**0.656**	**0.793**	**0.839**

Table 2: Retrieval metrics (top-k recall) comparison. On the test dataset, we measure percentage of situations, where given a proof state, the ground truth premise has been retrieved among top-k according to the PSM model. The *6L transformer + neg.* entry refers to a model utilizing our negative sampling strategy with 1536 additionally sampled negatives (see A.2 for details).

D Proofs

```
Theorem 1:
lemma reachable_steps: "<exists> xs. steps xs <and> hd
xs = s<ŝub> <and> last xs = x" if "reachable x"

Original proof:
using that
unfolding reachable_def
proof induction
case base
then
show ?case
by (inst_existentials "[sŝub>0]"; force)
next
case (step y z)
from step.IH
guess xs
by clarify
with step.hyps
show ?case
apply (inst_existentials "xs @ [z]")
apply (force intro: graphI)
by (cases xs; auto)+
qed

Our proof:
using that
unfolding reachable_def
by (fastforce dest: reaches_steps)
```

Proof 1: Our model is capable of proposing short and neat proofs when compared to the original.

```
Theorem 2:
lemma (in wf_digraph) iapath_dist_ends: "<And>u p v.
iapath u p v <Longrightarrow> u <noteq> v"

Original proof:
unfolding pre_digraph.gen_iapath_def
by (metis apath_ends)

Our proof:
by (unfold gen_iapath_def) (auto dest:
apath_nonempty_ends)
```

Proof 2: Exemplary proof that state-only model failed to close, whereas our PGLM+PSM managed to derive a fundamentally different proof without using metis - in contrast to original proof.

E Inputs comparison

```
<|PREMISE_NAME|>less_top_ennreal
<|PREMISE|>"x < top <longleftrightarrow> (<exists>r<ge>0. x = ennrealr)"
<|PREMISE_NAME|>fact_dvd_higher_pderiv
<|PREMISE|>"[:fact n :: int:] dvd (pderiv ^^ n) p"
<|PREMISE_NAME|>sameDom_sym
<|PREMISE|>"sameDom inp inp' = sameDom inp' inp"
<|PREMISE_NAME|>moebius_inverse
<|PREMISE|>assumes "a * d <noteq> b * c" "c * z + d <noteq> 0" shows "moebius
d (-b) (-c) a (moebius a b c d z) = z"
<|ISA_OBS|>proof (prove) goal (1 subgoal): 1. prv (neg <phi>R)
<|PREV_STEPS|>have "prv (neg <phi>R)"<|PROOF_STEP|>
```

Input 1: Exemplary input for PGLM+PSM model with top-4 premises. Input is a single sentence, here, for readability, split into multiple lines.

```
<|ISA_OBS|>proof (prove) goal (1 subgoal): 1. prv (neg <phi>R)
<|PREV_STEPS|>have "prv (neg <phi>R)"<|PROOF_STEP|>
```

Input 2: Exemplary input for classical State-only model. Input is a single sentence, here, for readability, split into multiple lines.