NaturalProver: Grounded Natural Language Proof Generation with Language Models

Sean Welleck[12*], Jiacheng Liu[1*],
Ximing Lu[2], Hannaneh Hajishirzi[12], Yejin Choi[12]

[1]University of Washington, [2]Allen Institute for Artificial Intelligence, *Equal Contribution

Introduction. We envision assistive systems for *informal* mathematics that suggest proof steps or solutions to a user, inspired by the use of language models in formal proof assistants (e.g. [4, 6, 7, 8, 11]) and informal premise selection [3, 5, 13]. We study two new generation tasks in natural mathematical language: suggesting the next step in a proof, and full-proof generation.

We develop NaturalProver, a language model that generates proofs by conditioning on background references (theorems, definitions), and optionally enforces their presence with constrained decoding. NaturalProver improves the quality of next-step suggestions and generated proofs over fine-tuned GPT-3 [1], with either retrieved or human-provided references, according to human evaluations from university-level mathematics students.

NaturalProver is capable of proving short (2-6 step) theorems and providing next-step suggestions that are rated as correct and useful more than 50% of the time, which is to our knowledge the first demonstration of these capabilities using neural language models.

Data. We create a NaturalProofs-Gen dataset with data adapted from the ProofWiki domain of NaturalProofs [13]. Each example pairs a theorem $\mathbf{x}$ with a gold proof $\mathbf{y} = (y_1, \ldots, y_T)$, where each y_t is a variable-length proof step. Each proof mentions references $\{\mathbf{r}_1, \ldots, \mathbf{r}_{R_y}\}$ from a reference set of roughly 33k theorems and definitions, analogous to how Wikipedia articles reference other pages. For example, Figure 1 shows a 4-step proof with references in blue. We use splits from NaturalProofs for training, and create evaluation sets with 100 validation and 100 test theorems.

Tasks. The **proof generation** task is to generate a proof $\mathbf{y}$ given theorem $\mathbf{x}$. The **next-step** task is to generate a next step y_t given theorem $\mathbf{x}$ and proof history $y_{<t}$ from a gold proof. We consider an additional *provided* setting where the model is given gold references $\{\mathbf{r}^*_1, \ldots, \mathbf{r}^*_{R_y}\}$.

Methods. We study a vanilla language model and two 'knowledge-grounded' variations, along with the effect of constrained decoding. For each model, we fine-tune GPT-3 Curie, a ≈13B parameter autoregressive transformer language model trained on internet text.[1]

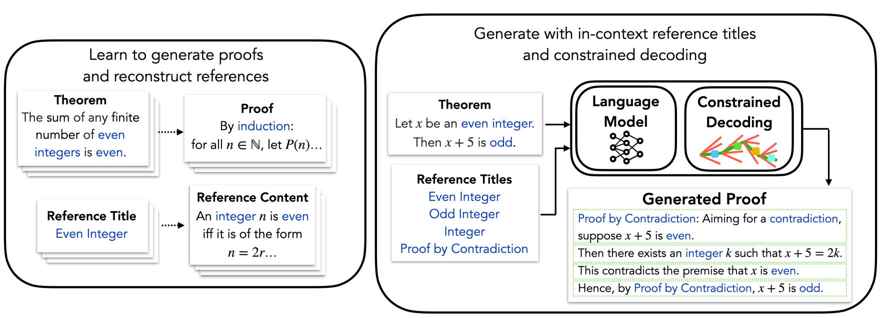

Figure 1: NaturalProver proves `Even Integer Plus 5 is Odd`.

[1]`https://blog.eleuther.ai/gpt3-model-sizes/`. We use the OpenAI API. We also release open-source GPT-J and GPT-2 models, fine-tuning and evaluation code, and the NaturalProofs-Gen dataset.

	Reasoning Errs (↓)			Lexical Errs (↓)		Per-Step (↑)		Full Proof (↑)	
Method	Ref.	Eqn.	Other	Lang.	Sym.	Useful	Correct	Useful	Correct
GPT-3 (curie)	30.92	32.54	40.15	5.61	5.24	25.69	28.18	20%	13%
Retrieved	23.52	37.55	23.66	4.54	6.19	41.54	33.56	32%	24%
Provided	25.84	35.93	25.23	8.41	5.35	39.60	26.30	35%	24%
+constrained	**23.61**	**28.54**	**18.45**	5.58	3.65	**46.57**	**35.41**	**45%**	**32%**
Next-step	19.70	26.32	19.10	8.57	5.86	51.43	42.86	–	–

Table 1: Human evaluation results for full-proof and next-step generation (bottom).

The knowledge-grounded models condition on references, $p_\theta(\mathbf{y}|\mathbf{x}, R)$. As language model context windows prevent conditioning on full reference documents, we condition on reference *titles*, and fine-tune on (title, content) pairs, which lets the model memorize the associated content. For example, Fig 1 shows `Even Integer` and its content. We study 3 variants:

1. **Baseline.** This model is simply fine-tuned on the $12.5k$ (theorem, proof) training examples. At test time, the model is given a theorem and uses greedy decoding to generate a proof.
2. **Retrieved.** This model is conditioned on *retrieved* references, $p_\theta(\mathbf{y}|\mathbf{x}, \hat{\mathbf{r}}_1, \ldots, \hat{\mathbf{r}}_{20})$. We use a pretrained joint retrieval model from [13], which was trained on NaturalProofs to map each theorem to the references in its ground-truth proof. At test time, we condition on a test theorem and its top-20 retrieved reference titles, and use greedy decoding.
3. **Provided.** This model is conditioned on human-provided references, $p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{r}^*_1, \ldots, \mathbf{r}^*_{R_\mathbf{y}})$, meaning $\{\mathbf{r}^*_1, \ldots, \mathbf{r}^*_{R_\mathbf{y}}\}$ is the set of reference-titles mentioned in a ground-truth proof. At test time, the model receives a test theorem and reference titles from a ground-truth proof.

Constrained decoding. We use constrained decoding to improve reference usage in the provided setting, as references are known to be relevant to a proof of the theorem. We generate step-by-step by sampling multiple step candidates, keeping those with high log-probability and reference-coverage in a beam, and continuing to the next step.

Evaluation. We created a schema of reasoning and lexical errors and an online system for per-step and full proof annotation. We recruited 15 students from the Departments of Mathematics and Applied Mathematics at the University of Washington as annotators. Annotators label the $\{0, 1\}$ correctness, usefulness, and presence of errors in each proof step, then rate the full proof's correctness and usefulness. We also find positive correlations between human judgments and automatic lexical (e.g. Gleu) and grounding (e.g. Reference-F1) metrics and discuss these results in the talk.

Main results. We show our main human evaluation results in Table 1. Knowledge-grounding, either retrieved or human provided, improves proof generation. Constrained decoding further improves the provided-knowledge model, with 32% of its proofs rated as correct and 45% rated as useful as an aid for human proof writers. On the per-step level, 35% of its proof steps are correct and 47% are useful, increasing to 51% useful and 43% correct given a correct proof-so-far. On the other hand, our models often struggle with correctly deploying and utilizing references (23.6% reference error rate), doing symbolic derivations (28.5% equation error rate), and longer proofs. We give quantitative and qualitative analyses of these successes and errors in the talk.

Looking forward. Our results suggest that useful interactive proof assistants for informal mathematics are plausible as methods improve further. Investigating architectural improvements [14], iterative improvement [2, 7], and pretraining [9], as well as the role of formalization [10, 12] in informal proof generation are interesting future directions.

References

[1] 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, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff 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. *ArXiv*, abs/2005.14165, 2020.

[2] Bhavana Dalvi, Oyvind Tafjord, and Peter Clark. Towards teachable reasoning systems. *ArXiv*, abs/2204.13074, 2022.

[3] Deborah Ferreira and André Freitas. Natural language premise selection: Finding supporting statements for mathematical text. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2175–2182, Marseille, France, May 2020. European Language Resources Association.

[4] Jesse Michael Han, Jason Rute, Yuhuai Wu, Edward Ayers, and Stanislas Polu. Proof artifact co-training for theorem proving with language models. In *International Conference on Learning Representations*, 2022.

[5] Jesse Michael Han, Tao Xu, Stanislas Polu, Arvind Neelakantan, and Alec Radford. Contrastive finetuning of generative language models for informal premise selection. In *AITP*, 2021.

[6] Albert Qiaochu Jiang, Wenda Li, Jesse Michael, Han Openai, and Yuhuai Wu. LISA: Language models of ISAbelle proofs. In *AITP*, 2021.

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

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

[9] Markus Norman Rabe, Dennis Lee, Kshitij Bansal, and Christian Szegedy. Mathematical reasoning via self-supervised skip-tree training. In *International Conference on Learning Representations*, 2021.

[10] Christian Szegedy, editor. *A Promising Path Towards Autoformalization and General Artificial Intelligence*, 2020.

[11] Josef Urban and Jan Jakubův. First neural conjecturing datasets and experiments. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020.

[12] Qingxiang Wang, Chad Brown, Cezary Kaliszyk, and Josef Urban. Exploration of neural machine translation in autoformalization of mathematics in Mizar. In *CPP 2020 - Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs, co-located with POPL 2020*, 2020.

[13] Sean Welleck, Jiacheng Liu, Ronan Le Bras, Hannaneh Hajishirzi, Yejin Choi, and Kyunghyun Cho. Naturalproofs: Mathematical theorem proving in natural language. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[14] Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2022.