Reasoning or Spurious Correlations? Applying transformers to propositional logic

Daniel Enström[1], Viktor Kjellberg[1], and Moa Johansson[2]

[1] University of Gothenburg, Gothenburg, Sweden.
{gusensda, guskjevia}@student.gu.se
[2] Chalmers University of Technology, Gothenburg, Sweden.
moa.johansson@chalmers.se

Abstract

We experiment with a transformer model (BART) to investigate its capabilities for learning reasoning in propositional logic from data. Previous work has highlighted the pitfalls when trying to solve this with a classifier: it tends to learn spurious correlations of the dataset, not reasoning. Here, we augment the data with proof steps, and demonstrate that generative models trained fine-tuned on proofs better approximate logical reasoning, also on out-of-distribution data.

1 Introduction

Language models are now being applied to tasks beyond pure generation of text [1, 9]. The ability to reason logically, and produce proofs is one such task [10]. However, it is not always clear if this emerging functionality really corresponds to a model having learnt logical reasoning, or if it is simply picking up on some other pattern in the data, that seemingly allows it to solve reasoning tasks. It has for instance been shown that inducing large language models to "reason step by step" (via the prompt or by emitting intermediate steps) improves results on reasoning tasks [3, 5, 6, 8, 12, 11]. So, *why* does this step-by-step reasoning work? What are the features used here, compared to otherwise? When can we be confident that the model does reasoning, rather than picking up on some other relationship? Prystawski and Goodman investigates this in the context of Bayesian inference [7]. They find that step-by-step reasoning works when concepts not appearing close together in the training data can be linked together by concepts that do. We are interested in studying this in the context of logic reasoning, and take as a starting point the work by Zhang et al. [13]. They showed that the performance of a classifier BERT model [2], trained to accurately predict if problems in propositional logic were satisfiable (or not), turns out to have learnt spurious correlations arising from characteristics of the problem set (e.g. the number of rules). Faced with problems from a different distribution it failed miserably. We instead train two generative seq-to-seq BART [4] models in two different ways: 1) producing a short proof in one go, and 2) sequentially producing the next proof step, given the problem state. The one-go method gives mixed results, while next-rule prediction applied in a step-by-step fashion produces near-perfect cross-distribution accuracy.

2 Data and Model

The SimpleLogic dataset consists of 860 000 propositional logic problems [13]. Each problem includes a query literal, a list of facts (positive literals) and a list of rules represented as Horn clauses with 1 - 3 premises. SimpleLogic problems are divided into three distributions based on the strategy employed to generate them: Label Priority (LP), Rule Priority (RP), and a

balanced version of RP (RP_b). RP for instance, has a potentially spurious feature where the number of rules is higher for queries that hold, while RP_b was designed to remove this correlation. For our experiments, we augmented these datasets with proofs in order to train and evaluate the models. Using the augmented datatset, we then trained two models starting from a pre-trained BART model: the first simply generates a whole candidate proof string in one go, and is called Whole Proof-BART (WP-BART), while the second is based on a neuro-symbolic architecture, and is called Symbolic Iterative Proof-BART (SIP-BART). The SIP-BART model was designed following the idea of chain-of-thought prompting for logical tasks [6, 12]. Here, the BART model is trained to produce the next proof step in an iterative manner resembling forward-chaining, which is then passed to a symbolic module which process the generated output to create a new input state, being passed back to the neural part. The processed stops when either the proof is complete, or the search space has been exhausted. We hypothesise that these methods would force the models to learn more relevant features, and avoid short-cutting reasoning by learning spurious correlations between the problem presentation and the truth value of the query.

Three versions of each model was trained, one for each of the subsets LP, RP and RP_b, and then tested on test sets from each to detect how well each model generalised. Results are reported in relation to problem depth. The minimum depth of 0 means that no rule is required to solve the problem. Deeper problems requires a longer chain of rules that are dependent on other rules in order to prove the truth-value of the query. In SimpleLogic, the maximum depth of a problem is 6.

3 Results and Conclusions

The models are evaluated on the accuracy of the generated label (is the query true or false). We first compare our models with that of Zhang et al. [13] to assess if training also on proofs improve out of distribution performance. We here simply compare the models' accuracy on determining whether a problem is satisfiable or not, as the model in [13] does not produce proofs. Results are show in Tables 1-3 in Appendix A.

The mean accuracy of WP-BART did not improve from the result achieved by Zhang et al. However, there was an improvement on out of distribution (OOD) problems requiring longer proofs. Our results suggest that WP-BART does not pick up spurious statistical features to the same extent. It performs somewhat better OOD, and also does show less variability going from problems with shallow to deep proofs, and also less variability when generalizing to other distibutions, as seen in Table 2 in Appendix A. It also shows similar accuracies on RP and R_b in contrast to the models trained by Zhang et al.

SIP-BART on the other hand, achieved a near-perfect accuracy across all distributions, and only did marginally worse on OOD problems, as seen in Table 3 in Appendix A. The accuracy is high even on deeper problems, with a minimum accuracy of 98.7 % on depth six. This step-by-step approach seems able to generalize well to other distributions, which suggests that it is able to approximate reasoning better than the other model-variants. Furthermore the consistency of the proofs generated by SIP-BART are almost perfect. The few errors that occur can be divided into four different types, which all relate to the fact that we are using a pre-trained transformer model for natural language on a reasoning task:

- Non-existing Rule: The neural part produce a rule that does not exist in the problem description. This may happen because it accidentally replaces a word for a synonym, or misses a premise of a rule.

- Inapplicable Rule: The model has mistaken the conclusion of a rule for a fact, likely as facts are listed immediately after the rule in the input string.
- Unexhausted Search Space: The model prematurely concludes that the query is false, before exhausting the search space.
- Spurious Match: The model proves the wrong query by mixing it up with a synonymous word.

To summarise, the effect of learning spurious correlations for determining the validity of a propositional logic problem, identified by Zhang et al. [13], seem to be reduced by training the model on not only the problem description, but also on associated proofs. Already WP-BART, which is trained on whole proofs, seem to perform better out of distribution. For SIP-BART, the effect of these spurious correlations all but disappears, as evident in its capability to solve the problems of bigger depths. While we cannot fully rule out other unknown correlations, our results are consistent with prior work on step-by-step reasoning, and the hypothesis put forward in [7]: that adding step-wise inferences help the model learn how to connect also distant concepts in the training data. We also identify four new types of errors that still occur in SIP-BART. All are related to using a pre-trained transformer language model to do reasoning - none would appear in a symbolic theorem prover. We believe these types of errors are worth being aware of, as transformers are increasingly being applied to natural language tasks also involving logical reasoning.

Acknowledgement

The computations and storage of data were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) at Chalmers Centre for Computational Science and Engineering (C3SE), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

References

[1] Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc V Le. Massive exploration of neural machine translation architectures. *arXiv preprint arXiv:1703.03906*, 2017.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[3] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[4] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[5] 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, 2021.

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

[7] Ben Prystawski and Noah D. Goodman. Why think step-by-step? Reasoning emerges from the locality of experience, 2023.

[8] Markus N. Rabe, Dennis Lee, Kshitij Bansal, and Christian Szegedy. Mathematical reasoning via self-supervised skip-tree training. *arXiv: Learning*, 2020.

[9] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[10] David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*, 2019.

[11] Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online, August 2021. Association for Computational Linguistics.

[12] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2022.

[13] Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang, and Guy Van den Broeck. On the paradox of learning to reason from data, 2022.

Appendix A

Train	Test	0	1	2	3	4	5	6	Mean
RP	RP	99.8	100.0	99.4	98.9	98.6	96.9	95.9	**98.5**
RP	LP	99.9	99.9	99.0	94.3	83.8	65.6	50.0	**84.7**
RP	RP_b	99.2	99.2	98.6	98.0	96.6	93.9	89.1	**96.4**
LP	RP	97.4	92.5	64.5	60.2	67.6	72.6	69.9	**75.0**
LP	LP	99.8	99.8	99.8	99.6	98.8	97.2	95.4	**98.6**
LP	RP_b	97.7	93.3	60.2	56.7	63.9	68.7	68.5	**72.7**
RP_b	RP	99.8	99.9	99.5	98.9	98.6	97.9	96.9	**98.8**
RP_b	LP	99.7	99.4	99.3	96.4	87.6	72.6	57.2	**87.5**
RP_b	RP_b	99.6	99.5	99.0	98.4	98.0	96.7	94.1	**97.9**

Table 1: Accuracies from the Zhang et al. [13] BERT model. The integers refer to the depth of the ground-truth proof. Mean is the average across all depths.

Train	Test	0	1	2	3	4	5	6	Mean
LP	LP	100.0	100.0	92.6	90.2	89.8	91.2	93.3	93.9
LP	RP	100.0	99.9	83.3	65.5	67.3	72.0	76.5	80.6
LP	RP_b	100.0	99.9	82.1	64.9	66.3	74.0	83.0	81.4
RP	LP	84.7	85.4	79.2	73.9	71.6	68.5	63.4	75.2
RP	RP	84.3	88.5	87.9	87.6	85.0	79.8	78.1	84.5
RP	RP_b	87.7	88.3	88.8	87.7	85.4	80.7	79.7	85.5
RP_b	LP	84.1	94.3	89.6	85.1	80.9	76.6	71.6	83.2
RP_b	RP	84.0	94.2	94.3	92.2	89.3	84.8	82.2	88.7
RP_b	RP_b	87.2	93.6	94.1	93.3	88.6	86.8	85.7	89.9

Table 2: Accuracies from WP-BART. The integers refer to the depth of the ground-truth proof. Mean is the average across all depths.

Train	Test	0	1	2	3	4	5	6	Mean
LP	LP	99.9	99.9	99.8	99.9	99.5	99.6	99.5	99.7
LP	RP	100.0	99.9	99.7	99.2	99.1	99.3	98.7	99.4
LP	RP_b	100.0	99.8	99.7	99.0	99.2	99.3	99.2	99.4
RP	LP	100.0	99.8	99.7	99.4	98.8	98.7	98.7	99.3
RP	RP	100.0	100.0	99.9	99.6	99.6	99.7	99.5	99.7
RP	RP_b	100.0	100.0	99.8	99.6	99.4	99.6	99.7	99.7
RP_b	LP	100.0	99.8	99.7	99.4	99.0	98.9	98.7	99.4
RP_b	RP	99.9	100.0	99.9	99.6	99.7	99.6	99.6	99.8
RP_b	RP_b	100.0	100.0	99.8	99.5	99.5	99.7	99.7	99.7

Table 3: Accuracies from SIP-BART trained on the different distributions. The integers refer to the depth of the ground-truth proof. Mean is the average accuracy across all depths.