Measuring and Improving BERT's Mathematical Abilities by Predicting the Order of Reasoning

Piotr Piękos
University of Warsaw
piotrpiekos@gmail.com

Mateusz Malinowski*
DeepMind

Henryk Michalewski*
University of Warsaw, Google

Abstract

Imagine you are in a supermarket. You have two bananas in your basket and want to buy four apples. How many fruits do you have in total? This seemingly straightforward question can be challenging for data-driven language models, even if trained at scale. However, we would expect such generic language models to possess some mathematical abilities in addition to typical linguistic competence. Towards this goal, we investigate if a commonly used language model, BERT, possesses such mathematical abilities and, if so, to what degree. For that, we fine-tune BERT on a popular dataset for word math problems, AQuA-RAT, and conduct several tests to understand learned representations better.

Since we teach models trained on natural language to do formal mathematics, we hypothesize that such models would benefit from training on semi-formal steps that explain how math results are derived. To better accommodate such training, we also propose new pretext tasks for learning mathematical rules. We call them (Neighbor) Reasoning Order Prediction (ROP or NROP). With this new model, we achieve significantly better outcomes than data-driven baselines and even on-par with more tailored models. We also show how to reduce positional bias in such models.

1 Introduction

Automatically solving math word problems has a long history dating back to the middle sixties (Bobrow, 1964). Early approaches were rule-based matching systems that solve the problem symbolically. Even though there are some impressive symbolic systems that operate in a relatively narrow domain, the inability to successfully scale them up is sometimes presented as a critique of the good-old-fashioned AI, or GOFAI (Dreyfus et al., 1992). One issue is to create a formalism that covers all the aspects needed to solve these problems. On the other hand, deep learning (LeCun et al., 2015) aims to develop artificial general intelligence that scales better to various problems.

However, despite many successes in computer vision and natural language processing (Devlin et al., 2018; He et al., 2016; Krizhevsky et al., 2012; Lan et al., 2019; Mikolov et al., 2013), data-driven methods evade our dream of building a system with basic, every-day, mathematical skills. As large-scale natural language models become more common (Devlin et al., 2018; Brown et al., 2020), we would expect them to also reason mathematically.

Since natural language understanding also involves symbolic manipulation (Liang, 2016), we treat mathematical reasoning as a language understanding and revisit the data-driven paradigm. For that, we rely on a recent language model, BERT (Devlin et al., 2019), and challenge it with math word problems (Ling et al., 2017). Even though such language models have initially shown promising results, more recent investigation shows they may rely on various biases in their predictions (Hendricks et al., 2018; Brown et al., 2020; Bhardwaj et al., 2020; Kurita et al., 2019). Here, we also follow that line of investigation and show these models can answer correctly without an understanding of the rationale behind it.

Furthermore, as directly predicting answers to math problems often requires multiple steps of reasoning, we show that we can improve BERT's generalization by exposing it to rationales (Ling et al., 2017; Hendricks et al., 2016; Lei et al., 2016). These are, however, only used during training similarly to a teacher that shows a student a justification for each answer. But then, the student is evaluated only on the ability to answer these questions during the college exam correctly with no access to rationales. Finally, to learn a better representation

*Authors have contributed equally.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Short Papers), pages 383–394
August 1–6, 2021.

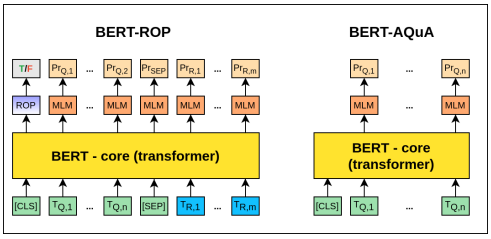

Figure 1: BERT (right) and our novel extension (left). We use shared architecture but we separate question tokens (green blocks) from rationales (blue blocks) using different segment and positional embeddings. We show all three losses. MLM predicts masked tokens (depicted here as $Pr_{Q,k}$). We use ROP or NROP to predict if the ordering of rationale steps is correct. For question-answering, we fine-tune the whole model with a classification layer using softmax. We use the embedding that corresponds to the [CLS] token as the input representation.

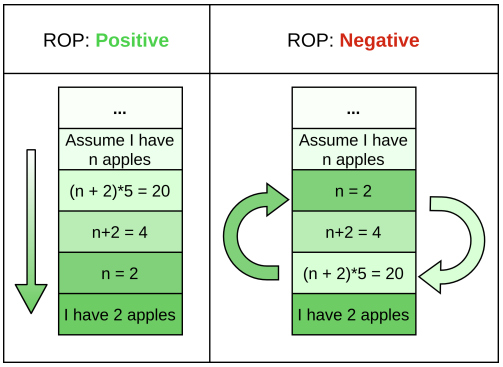

Figure 2: ROP or NROP with positive (left) and negative (right) labels. We randomly swap two rationales and classify if that change has happened.

from rationales and to improve the generalization even further, we introduce novel pretext tasks and corresponding losses, which we name (Neighbor) Reasoning Order Prediction (ROP or NROP). We also show that permutation invariant losses can lead to less biased representations. With that, we outperform other data-driven baselines, and are even on-par with methods that are more tailored to math-world problems and the AQuA-RAT dataset.

2 Methods

We use the following methods, each initialized with BERT-base pre-trained on Wikipedia and Books Corpus (Devlin et al., 2018; Zhu et al., 2015). Note that, in fine-tuning they all have the same number of parameters.

1) **BERT-base**. We fine-tune BERT to predict the correct answer and show its transfer to math word problems.

2) **BERT-AQuA**. We use the MLM loss on the AQuA-RAT questions before training to predict correct answer.

3) **BERT-AQuA-RAT**. We use the MLM loss on the AQuA-RAT questions and rationales and show if we can inject knowledge from rationales into BERT.

4) **BERT-(N)ROP**. We use the MLM loss and the novel (N)ROP loss for coherence prediction (defined later) and show if we can improve the results by focusing the model on rationales.

Later in this paper, we propose permutation invariant losses that additionally reduce positional biases of the BERT-base model, and can work with all the pretext tasks described above.

2.1 Architectures, pretext tasks and losses

We base our architecture on BERT (Devlin et al., 2019) that has 12 transformer blocks (Vaswani et al., 2017). As the core, we use the standard configuration described in (Devlin et al., 2019). We use three self-supervised losses. One is the standard Masked Language Modelling (MLM) but extended to work on rationales. Other two are our new losses, (Neighbour) Reasoning Order Prediction (ROP or NROP). Figure 1 shows two variants of our models. Note that, during fine-tuning, rationales and all the self-supervised losses are discarded.

MLM is the Masked Language Modelling (Devlin et al., 2019). We randomly mask 15% of the input tokens by a special token [MASK]. The objective of this loss is to predict the masked token using its context casted as a classification problem over the tokenizer vocabulary. Loss is calculated only on masked tokens. We extend this loss to rationales. First, we randomly choose whether we mask a question or rationale. Next, we follow the procedure above applied to either a question or rationale. However, to encourage binding between questions and rationales, we use the whole context for the predictions. Interestingly, there are parallels between masking numbers and solving mathematical equations, where it can be seen as solving the equation with unknown. For example, $2 + [\text{MASK}] = 4$ becomes $2 + x = 4$. As a consequence, models during training organically deal with mathematical calculations without defining a specific loss for mathematics allowing soft transitions between natural and more formal languages.

ROP is our novel coherence loss. Since rationales are sequences of consecutive reasoning steps, the order of the execution is critical as shown in Figure 2. Following this intuition, we introduce

Reasoning Order Prediction (ROP) that predicts whether the order of the rationale steps is preserved. Hence it encourages the network to pay more attention to rationales. The loss is similar to Sentence Order Prediction (SOP) (Lan et al., 2019), but ours is focused on learning reasoning steps.

NROP is an extension of ROP where only consecutive rationale steps are swapped making the prediction (swap or no swap) task more challenging and, hence, it can arguably lead to a better representation as understanding the correct ordering is more nuanced. Indeed, we observe that our models trained with NROP correctly predict if swap has occurred in about 75% cases, while with ROP in about 78% cases (both on the validation set). This indeed, confirms our hypothesis that NROP task is more challenging than ROP.

3 Results

Dataset. We use AQuA-RAT (Ling et al., 2017). It has about 100k crowd-sourced math questions with five candidate answers (one is correct). Each question has a rationale – a step-by-step explanation of how the answer is computed – that is only available during training. At test time answer predictions are based on questions. The train set has roughly 100k question-answer-rationale triples, while dev and test about 250 question-answer pairs each.

Main results. Table 1 shows our main results. We see that our method is the state-of-the-art among the models with minimal inductive biases and is very competitive to the other two models that are more specific to handle word math problems (e.g., requires programs). Moreover, even though BERT is already a stronger model than LSTM, it is better to use its MLM pretext task and loss on the AQuA-RAT questions (BERT-AQuA) or even better on questions and rationales (BERT-AQuA-RAT). However, models with our novel coherence prediction losses can better learn from rationales (BERT-ROP and BERT-NROP).

Moreover, we observe a highly sensitive relationship between dev and test sets (Figure 3, left), where small changes in the accuracies in the former set can lead to more dramatic changes at test time. Indeed, the correlation of results between both sets is only 0.082. As the validation set is quite small, we propose an extended dev consisting of 5000 randomly chosen samples from the training set extended by the whole dev set. Although not ideal, and the sensitive relationship is still present

Model	Accuracy
Random chance	20.0%
LSTM (Ling et al., 2017)	20.8%
BERT-base (ours)	28.3(±2.0)%
BERT-AQUA (ours)	29.1(±1.7)%
BERT-AQuA-RAT (ours)	32.3(±1.8)%
BERT-ROP (ours)	35.4(±1.0)%
BERT-NROP (ours)	37.0(±1.1)%
AQuA-RAT (Ling et al., 2017)	36.4%
MathQA (Amini et al., 2019)	**37.9%**

Table 1: Comparison of data-driven (first six rows) with two hybrid approaches that use stronger and hence more specific inductive biases (last two rows). Standard deviation estimates (over random initializations) is given in parentheses, where we see our losses can reduce the variability slightly.

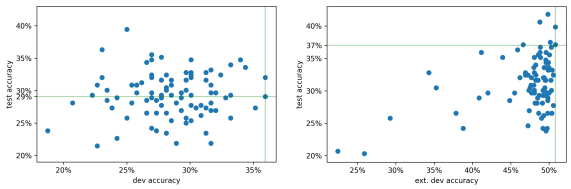

Figure 3: Accuracies for dev and test sets. Green lines show the iteration that maximizes validation accuracy. The image also shows the sensitivity of relationship between test and the original (left) or our extended (right) validation set.

(Figure 3, right), we have increased the correlation to 0.401. With such a new validation set, we report 37% test accuracy but we can also see that 40% is within the reach (Figure 3, right).

Rationales. We hypothesize that rationales contain information that is either missing or hard to extract from questions. For instance, their structure is different; they are more formal with emphasis on the logical steps. However, testing that hypothesis is non-trivial as there is a confounding factor – adding more rationales results in more data. Therefore, we artificially modify the dataset so that both models (one trained only on questions, and another one on questions and rationales) are trained on roughly the same number of data points. For that, we have estimated that rationales have 1.7 times more tokens than questions. This means that a question combined with rationale has around 3 times more tokens than just a question. If our hypothesis is valid, training on 20% questions and rationales should give better results than training on 60% questions (counting the number of tokens). We therefore created samples of respective sizes of just questions and questions combined with rationales. We show our results in Figure 4. The results suggest that adding more questions is insufficient and only slightly improves the overall performance.

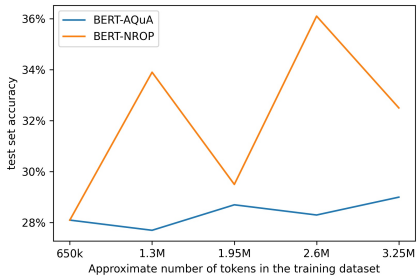

Figure 4: Accuracy scores conditioned on the number of tokens available for training. To support our argument that training on rationales is qualitatively different than questions, we align both together so that we have comparable number of tokens in both cases. Plot shows the progression of the dataset size. Starting with 650K of tokens - 20% dataset BERT-AQuA and 6.66% for BERT-NROP and ending with 3.25M - 100% of dataset for BERT-AQuA and 33.3% dataset for BERT-NROP. This shows that training with rationales leads to a better representation. Even better than training with more questions.

On the other hand, using rationales is more helpful.

Embeddings. To better understand the difference between BERT and BERT+NROP, we analyze their embeddings. For our analysis, we sample 2500 questions with a single operator in rationales, and next we visualise them with T-SNE (Van der Maaten and Hinton, 2008). We show both in Figure 5. We observe that BERT+NROP embeddings preserve more information about different operators.

Permutation consistency. Random guessing on AQuA-RAT yields 20%. With that in mind to separate questions that were solved by chance, we have constructed a new evaluation task – permutation consistency test – where each question gets 5 answers at different positions. Table 2 shows our procedure. Here, models only score a single point if they solve all 5 questions correctly. Hence, a random chance is 0.032% in such experiments.

Table 3 shows our results. BERT+NROP solves almost three times as many questions as BERT. Additionally, further inspection shows that BERT relies on choosing the answers that most stand out, e.g., numbers ending with zeros or floats while every other option is an integer. We didn't observe that simple patterns with BERT+NROP. Questions solved by BERT+NROP usually contain one or two operations and show that BERT+NROP better understands the problem. Below, we exemplify two math problems solved by both models.

Example of a problem solved by BERT+NROP: *8 man work for 6 days to complete a work. How many men are required to complete same work in 1/2 day?*
Answers: *A)93, B)94, C)95, D)96, E)97*

Original question	
How much is 27 / 3	A)13 **B)9** C)3 D)12 E)17
Generated questions	
How much is 27 / 3	**A)9** B)13 C)3 D)12 E)17
How much is 27 / 3	A)13 **B)9** C)3 D)12 E)17
How much is 27 / 3	A)13 B)3 **C)9** D)12 E)17
How much is 27 / 3	A)13 B)12 C)3 **D)9** E)17
How much is 27 / 3	A)13 B)17 C)3 D)12 **E)9**

Table 2: Our generation method for the permutation consistency test. Models get a point only if they solve all them.

Correct Option: *D*
Example of a problem solved by BERT *A ship went on a voyage. After it had traveled 180 miles a plane started with 10 times the speed of the ship. Find the distance when they meet from starting point.?*
Answers: *A)238, B)289, C)200, D)287, E)187*
Correct Option: *C*

Model	Score
Random chance	0.032%
BERT	4.33%
BERT+NROP	**11.02%**
BERT AUG	13.4%
BERT+NROP AUG	19.7%
BERT SEP-NC	15.0%
BERT+NROP SEP-NC	22.7%
BERT SEP-C	16.1%
BERT+NROP SEP-C	**23.9%**

Table 3: Our results for the permutation consistency test.

Drop from 37.0% to 11.02% (Table 3) suggests that models rely strongly on the order of answers. To reduce such a bias, we test several permutation invariant losses.

1) **AUG**. We sample randomly 25 permutations of all the possible answers and use them during training. Original ordering is not used, so there is no order bias. This is a data augmentation technique.

2) **SEP-NC**. The original models are trained on a 5-class classification task, where we build the representation by using questions and all the candidate answers, i.e., $\textbf{BERT}(Q||P)$. Here, $||$ denotes concatenation, Q is the question and P represents the sequence of all answers. In SEP-NC, we block the path between all the candidate answers and the BERT-base. Next, we use a late-fusion to predict if the given candidate answer matches with the question. That is, we use the following formulation $f(\textbf{BERT}(Q)||\textbf{BERT}(C))$, where $C \in P$ is a single candidate answer and f is a multi-layer perception (with two layers). At test time, the model

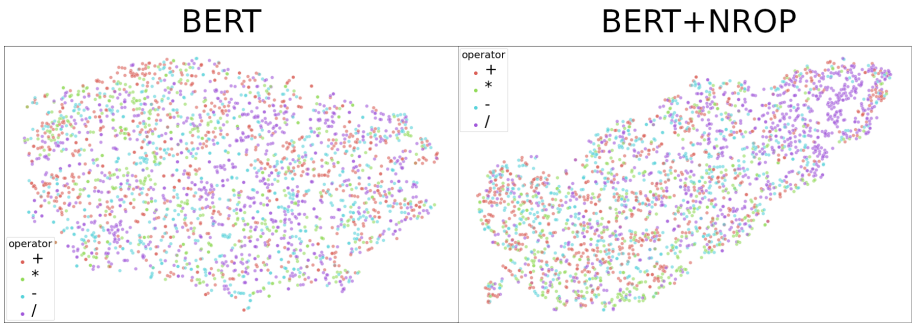

Figure 5: BERT and BERT+NROP embeddings. Colours represent different operators in rationales (T-SNE). BERT+NROP embeddings better separate operators.

is prompted to score all five candidate answers and select the one with the highest score. Appendix has more information about this method.

3) **SEP-C**. As models trained with SEP-NC do not have access to all the possible answers, their biases to answer positions are significantly reduced. However, these models cannot compare each answer to all other candidate answers. Here, we use the following formulation $f(\textbf{BERT}(Q||P)||\textbf{BERT}(C))$ to measure the compatibility of the input (question Q and all the candidate answers P) with the given candidate answer $C \in P$. We also reset the positional encoding between every possible answer in P. In such a way, we hypothesise the network can learn a less biased representation, and on the other hand, use relationship between the candidate answers. Table 3 shows SEP-NC and SEP-C vastly outperform the original model on the permutation consistency test. Details are in the appendix.

SEP-NC and SEP-C improve permutation consistency tests. Yet, they give similar results to original methods in accuracy measuring task. They achieve respectively 33.5% (SEP-NC) and 35.4% (SEP-C).

Questions difficulty. To better understand the models' performance, we check which questions are difficult for the model. We categorize questions by their difficulty for BERT-NROP and BERT. To estimate a question's difficulty, we have ranked the candidate answers according to the model's uncertainties. For instance, if the correct answer has the 2nd largest probability, we assign to that question difficulty two. With that, we group questions into 5 difficulty categories, from the easiest: $D_1, .., D_5$.

Manual inspection shows that for BERT+NROP: D_5 requires additional knowledge or implicitly defined numbers (e.g., adding first 100 numbers), D_4 requires geometry or non-linear equations and systems, D_3 requires solving linear systems with a few basic operations, D_2 requires solving simple equations, and D_1 has one or two basic operations with clearly written numbers. We show an example from each group in the supplementary material. We didn't observe a similar pattern for BERT with the exception of the easiest group D_1 where the model chooses the answer that is somewhat different from other candidates. We provide an example of each group in the supplementary materials.

Finally, we also compare the difficulty of questions with the difficulty perceived by humans. For that, we have conducted a small-group human study, where we have asked participants to solve some AQuA-RAT questions and rate their difficulty. We find a positive correlation between the difficulty measured by our models (as described above) to the difficulty judged by humans. We give more details in the appendix.

Conclusions. We have investigated if BERT (Devlin et al., 2019) – a pre-trained, large language model – can deal with mathematical reasoning. We find that its representation is biased (Brown et al., 2020; Bhardwaj et al., 2020; Kurita et al., 2019) also in mathematics. We investigate and describe that bias. Our novel pretext tasks and losses reduce that bias, but the network still finds shortcuts. We hope our work will spark interest of the community in developing language models capable of mathematical reasoning.

Acknowledgements. We thank Wang Ling (DeepMind) for his comments and suggestions on our draft. Also, we thank Piotr Biliński and all participants of the 2020 Machine Learning Project course at the University of Warsaw for the conversations about the project. All experiments were performed using the Entropy cluster funded by NVIDIA, Intel, the Polish National Science Center grant UMO-2017/26/E/ST6/00622 and ERC Starting Grant TOTAL. The work of Henryk Michalewski was supported by the Polish National Science Center grant UMO-2018/29/B/ST6/02959.

Impact Statement

Our research follows the data-driven paradigm for creating general-purpose language models with some mathematical skills. We expect that mathematically aware language models will broaden the spectrum of topics they can understand, increasing their reliability and making them more useful.

Improving mathematical abilities and coherence in language models is likely to affect question-answering or dialogue systems, search engines or text summarization systems.

One considerable risk in developing language models at scale is that they could use various workarounds and biases to achieve their results. We have shown that issues in the context of mathematical reasoning. Such problems can become hazardous when wrong numbers could lead to bad decisions. Additionally, a person could easily fall into the fallacy that the order of magnitude is correct even if the answer is incorrect. As we showed, the model can favour round numbers over the ones close to the right answer. To mitigate the risk, we encourage considering additional tests and investigating the models more rigorously.

References

Stephan Alaniz and Zeynep Akata. 2019. Explainable observer-classifier for explainable binary decisions. *arXiv preprint arXiv:1902.01780*.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. 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 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Rishabh Bhardwaj, Navonil Majumder, and Soujanya Poria. 2020. Investigating gender bias in bert. *arXiv preprint arXiv:2009.05021*.

Daniel G Bobrow. 1964. Natural language input for a computer problem solving system.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in neural information processing systems*.

Eugene Charniak. 1969. Computer solution of calculus word problems. In *Proceedings of the 1st international joint conference on Artificial intelligence*, pages 303–316.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Rodney R Cocking, Rodney T Cocking, and Jose P Mestre. 1988. *Linguistic and cultural influences on learning mathematics*. Psychology Press.

Mark G Core, H Chad Lane, Michael Van Lent, Dave Gomboc, Steve Solomon, and Milton Rosenberg. 2006. Building explainable artificial intelligence systems.

Scott Crossley, Tiffany Barnes, Collin Lynch, and Danielle S McNamara. 2017. Linking language to math success in an on-line course. *International Educational Data Mining Society*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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. Association for Computational Linguistics.

Hubert L Dreyfus, L Hubert, et al. 1992. *What computers still can't do: A critique of artificial reason*. MIT press.

John Rupert Firth. 1961. *Papers in Linguistics 1934-1951: Repr*. Oxford University Press.

Lynn S Fuchs, Douglas Fuchs, Donald L Compton, Sarah R Powell, Pamela M Seethaler, Andrea M Capizzi, Christopher Schatschneider, and Jack M Fletcher. 2006. The cognitive correlates of third-grade skill in arithmetic, algorithmic computation, and arithmetic word problems. *Journal of Educational Psychology*, 98(1):29.

Lynn S Fuchs, Douglas Fuchs, Karla Stuebing, Jack M Fletcher, Carol L Hamlett, and Warren Lambert. 2008. Problem solving and computational skill: Are they shared or distinct aspects of mathematical cognition? *Journal of educational psychology*, 100(1):30.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958, Online. Association for Computational Linguistics.

Caglar Gulcehre, Sarath Chandar, Kyunghyun Cho, and Yoshua Bengio. 2016. Dynamic neural turing machine with soft and hard addressing schemes. *arXiv preprint arXiv:1607.00036*.

Hossein Hajipour, Mateusz Malinowski, and Mario Fritz. 2020. Ireen: Iterative reverse-engineering of black-box functions via neural program synthesis. *arXiv preprint arXiv:2006.10720*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity mappings in deep residual networks. In *European conference on computer vision (ECCV)*, pages 630–645. Springer.

Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. 2016. Generating visual explanations. In *European Conference on Computer Vision*, pages 3–19. Springer.

Lisa Anne Hendricks, Kaylee Burns, Kate Saenko, Trevor Darrell, and Anna Rohrbach. 2018. Women also snowboard: Overcoming bias in captioning models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 771–787.

Danqing Huang, Shuming Shi, Chin-Yew Lin, Jian Yin, and Wei-Ying Ma. 2016. How well do computers solve math word problems? large-scale dataset construction and evaluation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 887–896, Berlin, Germany. Association for Computational Linguistics.

W Lewis Johnson. Agents that learn to explain themselves.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. 2019. Measuring bias in contextualized word representations. *arXiv preprint arXiv:1906.07337*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

H Chad Lane, Mark G Core, Michael Van Lent, Steve Solomon, and Dave Gomboc. 2005. Explainable artificial intelligence for training and tutoring. Technical report, UNIVERSITY OF SOUTHERN CALIFORNIA MARINA DEL REY CA INST FOR CREATIVE ....

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature*, 521(7553):436–444.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

Percy Liang. 2016. Learning executable semantic parsers for natural language understanding. *Communications of the ACM*, 59(9):68–76.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Meghann Lomas, Robert Chevalier, Ernest Vincent Cross, Robert Christopher Garrett, John Hoare, and Michael Kopack. 2012. Explaining robot actions. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pages 187–188.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Mateusz Malinowski, Carl Doersch, Adam Santoro, and Peter Battaglia. 2018. Learning visual question answering by bootstrapping hard attention. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–20.

Mateusz Malinowski and Mario Fritz. 2014. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in neural information processing systems*, pages 1682–1690.

Jose P Mestre. 2013. The role of language comprehension in mathematics and problem solving. In *Linguistic and cultural influences on learning mathematics*, pages 201–220. Taylor and Francis.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. 2014. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212.

Richard J Murnane, John B Willett, M Jay Braatz, and Yves Duhaldeborde. 2001. Do different dimensions of male high school students' skills predict labor market success a decade later? evidence from the nlsy. *Economics of Education Review*, 20(4):311–320.

Emilio Parisotto, Abdel-rahman Mohamed, Rishabh Singh, Lihong Li, Dengyong Zhou, and Pushmeet Kohli. 2016. Neuro-symbolic program synthesis. *arXiv preprint arXiv:1611.01855*.

Maria Chiara Pasolunghi, Cesare Cornoldi, and Stephanie De Liberto. 1999. Working memory and intrusions of irrelevant information in a group of specific poor problem solvers. *Memory & Cognition*, 27(5):779–790.

Markus N Rabe, Dennis Lee, Kshitij Bansal, and Christian Szegedy. 2020. Mathematical reasoning via self-supervised skip-tree training. *arXiv preprint arXiv:2006.04757*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. *arXiv preprint arXiv:1904.01557*.

Shuming Shi, Yuehui Wang, Chin-Yew Lin, Xiaojiang Liu, and Yong Rui. 2015. Automatically solving number word problems by semantic parsing and reasoning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1132–1142.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. Bertje: A dutch bert model. *arXiv preprint arXiv:1912.09582*.

Amber Y Wang, Lynn S Fuchs, and Douglas Fuchs. 2016. Cognitive and linguistic predictors of mathematical word problems with and without irrelevant information. *Learning and individual differences*, 52:79–87.

Lei Wang, Dongxiang Zhang, Lianli Gao, Jingkuan Song, Long Guo, and Heng Tao Shen. 2018. Mathdqn: Solving arithmetic word problems via deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Huijuan Xu and Kate Saenko. 2016. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *European Conference on Computer Vision*, pages 451–466. Springer.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. 2016. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 21–29.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

Yanyan Zou and Wei Lu. 2019. Text2Math: End-to-end parsing text into math expressions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5327–5337, Hong Kong, China. Association for Computational Linguistics.

A AQuA-RAT example

Question: *A starts a business with Rs.40,000. After 2 months, B joined him with Rs.60,000. C joined them after some more time with Rs.120,000. At the end of the year, out of a total profit of Rs.375,000, C gets Rs.150,000 as his share. How many months after B joined the business, did C join?*
Options: *A) 30, B) 32, C) 35, D) 36, E) 40*
Rationale:

Assume that C was there in the business for x months

$$A : B : C = 40000 * 12 : 60000 * 10 : 120000 * x$$
$$= 40 * 12 : 60 * 10 : 120x = 40 : 5 * 10 : 10x$$
$$= 8 : 10 : 2x$$
$$= 4 : 5 : x$$
$$\textit{C's share} = 375000 * x/(9 + x) = 150000$$
$$=> 375x/(9 + x) = 150$$
$$=> 15x = 6(9 + x)$$
$$=> 5x = 18 + 2x$$
$$=> 3x = 18$$
$$=> x = 18/3 = 6$$

It means C was there in the business for 6 months. Given that B joined the business after 2 months. Hence C joined after 4 months after B joined
Answer is B

Additional examples are in the supplementary material.

B Input representation

All BERT variants use the representation that corresponds to a special token [CLS] that we put at the beginning of the whole input sequence consisting of question tokens followed by rationale tokens, and in the downstream, question-answering task, rationale tokens are replaced by the answer options. With that, the classification uses the contextual embedding of [CLS] that captures the entire input. MLM classifies over the entire vocabulary of possible words while the other two losses use a binary cross-entropy loss for the predictions.

C Training protocol

We train all our architectures on AQuA-RAT using the following training phases. In all cases, we choose our best model based on the performance on the validation set (dev set), and report the final performance on the test set.

Pre-training. Each model is pre-trained on a large corpus of texts written in natural language sampled from English Wikipedia and BooksCorpus (Devlin et al., 2018; Zhu et al., 2015). We use this as the base (BERT-base) model that is also used in all other variants of BERT. In practice, we initialize all the models with the weights using the HuggingFace library (Wolf et al., 2019) and don't keep final layer for fine-tuning. Our model therefore has the same number of weights as BERT-base.

Self-supervision. Here, we use our newly introduced losses, ROP and NROP, where our models use questions and possibly rationales from the AQuA-RAT dataset. Both questions and rationales use the same word embeddings. However, to distinguish between both modalities we use two segment embeddings. The first one for all the question tokens, and the second one for all the rationale tokens. That is, the segment embedding is shared among all the question tokens, and separately among all the rationale tokens. We use dynamic masking (Liu et al., 2019). Here, tokens are randomly masked for each batch. We naturally extend this approach to other losses that we use in this phase. That is, ROP and NROP negative examples are randomly recreated every k epochs, where $k = 2$ in our case.

Fine-tuning is the last training phase. Here, once our models have learnt the representation during the self-supervised phase, we tune such a representation to the question-answering downstream task. In this task, our input consists of question tokens and possible answer options. There are five such options that comes with the dataset. Like other methods, we tread this as a five-class classification task where the classification head is added on top of the final embedding of the input. We consider the embedding corresponding to the first (from the left) [CLS] token as such the final representation.

D Implementation details

In our experiments, we use four TITAN V GPUs. We use a multi-gpu setup. In the pre-training phase, we use batch size equals to four for each GPU device. Therefore the effective batch size equals to sixteen. We use the learning rate $5 \cdot 10^{-5}$ and trained the models for 24 epochs. In the fine-tuning phase, we use early stopping criteria, based on the accuracy score on the validation set. We use the following criteria. If the model does not improve the performance in 15 consecutive epochs, we stop training, and evaluate a model that yields the highest validation performance. We use ADAM optimizer with learning rate 10^{-5} and gradient clipping that sets the maximal gradient's norm to one. All our settings use the same hyper-parameters but they differ due to the random initialization of our self-supervised networks (during the self-supervised training phase) and the classification networks (during the fine-tuning phase). Self-supervision phase takes around 4 days on 4 GPUs, whereas fine-tuning takes 8 hours on a single GPU.

E Permutation invariant methods

In the main paper, we have shown that typical models can use positional biases in achieving answers. This results in a low permutation consistency score (Table 3 in the main paper). To handle that issue, we have defined extra variants that do not use positional encodings for the answer options and instead they rely on the retrieval mechanics where input representations are matched against the candidate answers. Here, we describe two such variants.

E.1 Original methods

Original models create an embedding of a sentence extended by possible questions. This embedding is then transformed by a linear layer to predict the correct answer. That is,

$$o_1 = f_1(\textbf{BERT}(Q||P))$$

where o_1 is a 5-dimensional vector with probabilities for each possible answer, Q is a question, P are all possible answers, $||$ represents concatenation, f_1 is a single fully connected layer from 768-dimensional space to 5-dimensional space with the softmax activation. **BERT** is a BERT-base sentence embedding. The same approach is used for BERT+(N)ROP.

E.2 SEP-NC

In SEP-NC and SEP-C, we use separate embeddings for a question and **SEP**arate embedding for a candidate answer. They differ, however, in the fact that SEP-C has access to all five possible answers, while SEP-NC has access only to one prompted candidate answer. Therefore NC stands for ńo candidates", while C stands for ćandidates".

We train the SEP-NC model on a binary classification task to predict whether each candidate answer C is correct. The method produces two embeddings, one for question and another one for a candidate answer $C \in P$, and next concatenates them. That is,

$$o_2 = f_2(\textbf{BERT}(Q)||\textbf{BERT}(C))$$

where o_2 is an estimated probability that C is a correct answer, P is the sequence of all possible answers, f_2 is a single fully connected layer from 1536 (768 * 2) dimensional space to 1-dimensional space with the sigmoid activation. Note that, all candidate answers are independent of the question. That is, BERT cannot use positional biases in deriving an answer. At test time, the model is prompted to score all five candidate answers and select the one with the highest score. We naturally extended that approach to BERT+ROP and BERT+NROP. Table 3 (the main paper) shows a significant improvement over the baseline method.

E.3 SEP-C

SEP-NC method could be too restrictive as it does not allow the model to compare against different answers. Therefore, we propose another approach that 1) alleviate the issue with positional biases, but 2) can compare between different answer options. We call that approach SEP-C.

Originally for each token, a positional encoding is assigned based on its position. In SEP-C, before assigning positional encoding, we artificially reset the position at the beginning of each possible answer. For example, if possible answers are: $a)10, b)20, c)30, d)40, e)50$ they are changed into $10; 20; 30; 40; 50$ and after the tokenization, we get the following list of tokens: ['1','0', ';', '2', '0', ';', '3', '0', ';' ,'4', '0', ';', '5', '0']. Modified positional encoding will assign value based only on the relative position to the beginning of the current possible answer. Therefore, in the example above, each '0' will receive the same positional encoding, and '1' will get the same positional encoding as '2', '3', and so on.

Formally, we have

$$o_3 = f_3(\textbf{BERT}(Q||P_m)||\textbf{BERT}(C))$$

where P_m is the sequence of all the possible answers but modified as explained above. Note that, in this formulation, the model can use the information for all the possible answer options, but their order is not taken into account. Table 3 (the main paper) shows a significant improvement over the baseline method.

E.4 Human study

We carried an initial human study on the group of 16 volunteers from University of Warsaw. Volunteers were Mathematics and Informatics students from the Faculty of Mathematics, Informatics and Mechanics. We asked the participants to solve questions sampled from the AQuA-RAT dataset. We are interested in the relation between BERTs difficulty, BERT+NROP difficulty and human difficulty. Therefore to have a full image we would like to have 2 questions for each question difficulty pair, for example (D_1 BERT, D_2**:** BERT+NROP)

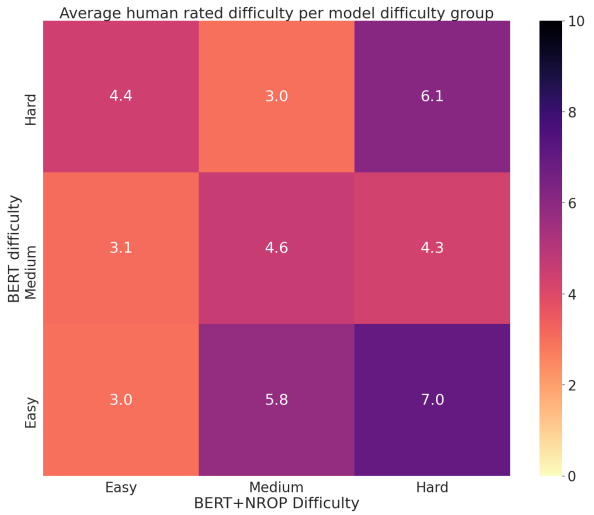

Figure 6: The average human-judged difficulty for questions from each model difficulty group.

. However, that would give 25 combinations and 50 questions if we wanted to have 2 questions per combination. That would be too much to ask from a volunteer participant. In order to reduce the number of questions, we group our 5 difficulty groups into 3 categories as follows.

- Easy: D_1
- Medium: D_2 and D_3 combined
- Hard: D_4 and D_5 combined

Because of that we have only 9 possible combinations and by sampling 2 questions from each combination we still have a feasible number of questions (18).

Apart from solving the question, we asked to rate question difficulty on a scale from 1 (the simplest) to 10 (the most challenging). In general, our participants were knowledgeable in math and solved all the questions correctly. With that grouping we now

The average human-rated difficulty for each of 9 combinations is presented in Figure 6. The results show that the progression of human difficulty is correlated with the difficulty judged by the models. Additionally, the human difficulty seems to be more sensitive to BERT+NROP difficulty than to BERTs. In other words, increasing the difficulty of BERT+NROP will increase the human difficulty more than the increasing difficulty of BERT. This observation fits our previous observations that BERT+NROP solves the most straightforward questions while BERT is looking for some leaks, like looking for the roundest answer.

dataset	A	B	C	D	E
train	21.03%	22%	22.87%	19.95%	14.15%
dev	27.17%	25.98%	16.93%	19.69%	10.24$
test	24.80%	22.83%	20.87%	18.11%	13.38%

Table 4: Answer distribution in each dataset.

F Distribution of answers

Table 4 shows the distribution of the answers in the AQuA-RAT (Ling et al., 2017) dataset in all the folds. Imbalance in distributions could potentially be used by models to find easy, shortcut solutions. For instance, a constant classifier that always choose the first answer (A) gets about 24% test accuracy.

G Negative results

While developing our self-supervised losses, we have developed another loss that turned out to be unhelpful. Here, we describe that loss as some its parts could be insightful for others. (N)ROP is a local loss focusing on rationales but not on the connections between questions and rationales. For that, we have developed Question Rationale Alignment (QRA). QRA changes a rationale with 50% probability to a randomly chosen rationale from the current batch. However, simply changing rationales would result in trivially solvable task in most cases. All the model would have to do is check whether numbers in the rationale and the question match. Hence, we mask number tokens with a special token QRA alone or QRA combined with NROP does not improve the results, it gives it gives 33.9% accuracy on the test in the best combination, so we didn't include it in the main results.

H Related work

We are inspired by the following research.

BERTology. We use BERT (Devlin et al., 2019) as our core. It uses Transformers (Vaswani et al., 2017); powerful neural architectures that applies a trainable function to all the pairs of input embeddings. It also uses masking that covers a fraction of the input words and requires the network to predict the hidden words based on the context. With both ingredients, the meaning (representation) of a word emerges from the "company it keeps" (Firth, 1961). In practice, often, such representations are pre-trained on large textual corpora with no need for annotations, and next fine-tuned on the downstream tasks. BERT's strong performance has resulted in the Cambrian explosion of studies of the

inner working mechanisms and various modifications (Clark et al., 2019; de Vries et al., 2019; Lan et al., 2019; Liu et al., 2019; Sanh et al., 2019; Radford et al.; Raffel et al., 2019; Yang et al., 2019). Finally, our Reasoning Order Prediction (ROP) is inspired by Sentence Order Prediction (SOP) (Lan et al., 2019). However, ROP works with multiple rationale sentences, where by changing the order we force the network to understand the consecutive "reasoning" steps. We have also further extended ROP to a more difficult Neighbor Reasoning Order Prediction (NROP).

Language and math. Development psychologists (Cocking et al., 1988; Mestre, 2013) often argue for the necessity of learning languages and point out that those with limited language skills are in danger of under-performing at school. Moreover, it is also believed that language studies involve discipline in learning and manipulating formal structures, and thus may promote the development of the organization of thoughts also required in mathematical reasoning. The similarity between linguistic competence and mathematics is especially pronounced when solving math word problems (Fuchs et al., 2006, 2008; Wang et al., 2016). Interestingly, attention appears to be crucial in problem solving (Fuchs et al., 2006; Pasolunghi et al., 1999). (Crossley et al., 2017) show that language skills are correlated with the performance in mathematical tests also among the university students. In particular, they pointed out that ability to use complex syntactic structures and cohesion devices are linked to better scores in a blended discrete mathematics course. We take inspiration from all such studies and decide to build our mathematical model based on language models.

Math word problems. Solving math word problems is a significant component of the mathematics curriculum and is taught very early, thoroughly, and universally. Such the emphasize is often motivated by that solving them is among the best predictors of employability, and is considered as a distinct area of mathematical competence (Murnane et al., 2001; Wang et al., 2016). Since solving such problems is unique to human intelligence, math word problems are also interesting for the AI community. This results in various approaches, more traditional symbolic methods, neural networks, and neuro-symbolic methods. (Bobrow, 1964; Charniak, 1969; Shi et al., 2015; Ling et al., 2017; Amini et al., 2019; Parisotto et al., 2016; Wang et al., 2018; Zou and Lu, 2019) as well as datasets (Ling et al., 2017; Amini et al., 2019; Huang et al., 2016; Saxton et al., 2019) An interesting approach is proposed in (Rabe et al., 2020), in which authors use self-supervised tasks on parsing trees of formal expressions. This approach requires syntax trees, and hence we would have to use an external parser. As our goal was to make an end to end model, we did not experiment with it, but there are no obstacles against using it in symbiosis with our methods. (Geva et al., 2020) also proposes self-supervised training for improving mathematical abilities in language models. We, however, focused on a data-driven approach to exclude choice biases and therefore restricted ourselves from using generated data.

Rationales. In human communication, we always expect there is some rationale behind each decision. Hence, we set the same expectations to our artificial agents. Symbolic or semi-symbolic architectures naturally produce justifications as a sequence of formulas in some formal language (Lane et al., 2005; Core et al., 2006; Lomas et al., 2012; Johnson; Liang, 2016; Malinowski and Fritz, 2014). Ideally, such rationales would also be shared and communicated to us through some language. The latter approach is especially appealing when applied to black-box neural networks. For instance, (Hendricks et al., 2016) propose a system that classifies the input image as well as it produces a textual explanation on "why this class is suitable for the given image".

Systems that produce explanations either in the form of the language (Ling et al., 2017; Hendricks et al., 2016), attention (Bahdanau et al., 2014; Mnih et al., 2014; Gulcehre et al., 2016; Malinowski et al., 2018; Xu and Saenko, 2016; Yang et al., 2016), phrase selection (Lei et al., 2016), distillation into programs (Hajipour et al., 2020), or decision trees (Alaniz and Akata, 2019) can potentially increase the transparency of the black-box neural networks. However, most of these approaches create rationales posthoc where the justification is conditioned on answers or by querying the network. In our work, we use rationales to learn a finer representation that can potentially lead to better decisions. In this sense, our technique is conceptually closer to methods that derive answers based on the program and use rationales paired with questions to guide the program induction process (Ling et al., 2017).