MinT: Boosting Generalization in Mathematical Reasoning via Multi-view Fine-tuning

Zhenwen Liang[1], **Dian Yu**[2], **Xiaoman Pan**[2], **Wenlin Yao**[2], **Qingkai Zeng**[1], **Xiangliang Zhang**[1], **Dong Yu**[2]
[1]University of Notre Dame [2]Tencent AI Lab, Bellevue
`{zliang6,qzeng,xzhang33}@nd.edu`
`{yudian,xiaomanpan,wenlinyao,dyu}@global.tencent.com`

Abstract

Reasoning in mathematical domains remains a significant challenge for relatively small language models (LMs). Many current methods focus on specializing LMs in mathematical reasoning and rely heavily on distilling knowledge from powerful yet inefficient large LMs (LLMs). In this work, we explore a new direction that avoids over-reliance on LLM teachers, introducing a multi-view fine-tuning method that efficiently exploits existing mathematical problem datasets with diverse annotation styles. Our approach uniquely considers the various annotation formats as different "views" that may help each other and leverages them in training the model. By postpending distinct instructions to input questions, models can learn to generate solutions in diverse formats in a flexible manner. Experimental results show that our strategy enables relatively small LMs to outperform prior approaches that heavily rely on knowledge distillation, as well as carefully established baselines. Additionally, the proposed method grants the models promising generalization ability across various views and datasets, and the capability to learn from inaccurate or incomplete noisy data. We hope our multi-view training paradigm could inspire future studies in other machine reasoning domains.

1 Introduction

To obtain mathematical reasoning models that are both efficient and effective, a widely explored direction is to specialize general-purpose LMs in mathematics [Fu et al., 2023] by supervised fine-tuning and distilling the knowledge and abilities from larger teacher models into smaller student models [Ho et al., 2022, Shridhar et al., 2022, Magister et al., 2022, Hsieh et al., 2023, Liang et al., 2023]. However, this kind of approach faces certain limitations. Firstly, it heavily relies on CoT explanations of existing data or extra CoT-style data generated by the larger models to train the smaller student model, and the most common choices for teachers are the GPT series and PaLM-540B Chowdhery et al. [2022], which are resource-intensive and costly. Moreover, LLMs might still make errors or fail to sufficiently explain reasoning steps, which could adversely influence the quality of the generated data and subsequently, the performance of the student models.

Instead of relying solely on inefficient LLMs to generate CoT annotations or additional training samples, in this paper, we focus on an under-explored question:

Can we effectively utilize publicly accessible datasets to develop small LMs specialized in mathematical problem solving?

Using existing annotated datasets can reduce manual effort and computational costs compared with relying on LLMs to generate additional annotated data. However, there are also several challenges posed by this direction. First, existing datasets vary significantly in their annotation formats. Additionally, as we collect more data from various data sources such as websites, the potential of encountering irrelevant or even inaccurate data cannot be disregarded. These differences in annotation styles and quality can hinder the effectiveness of these datasets in training math reasoners. Empirically, we observe that merely merging multiple datasets with different annotation formats cannot always improve model performance — in fact, it often has a negative effect.

To address the above challenges, we propose a **M**ulti-V**i**ew Fi**n**e-**T**uning (MinT) paradigm. In this context, the disparate annotation methods employed across different datasets are conceptualized as distinct "views" of mathematical problem solutions. To enhance the model's reasoning ability by fully leveraging existing data, we not only utilize the original views but also expand the solution views in existing math word problem datasets by view transformation. Then we append view-specific instructions to the input questions to guide the models to generate solutions in the desired view. Our underlying assumption is that training the model to comprehend various solution views equates to learning different methods of mathematical reasoning, which inherently helps strengthen its reasoning and generalization capabilities. Extensive experimental results support the efficacy of MinT, indicating that it fosters a variety of generalizations that contribute to enhancing overall performance across all views. Notably, our paradigm can also be used to incorporate noisy datasets, by regarding them as a new view, to further improve the performance of existing views.

2 Our Approach

2.1 Our Views

Clean Chain-of-Thought Explanations (CoT$_{clean}$) The first view, **clean c**hain-**o**f-**t**hought explanations (CoT$_{clean}$), is featured in the GSM8K dataset. This annotation style entails a thorough, step-by-step explanation of the solution process. Each intermediary step is clearly elaborated until the final solution is derived. These explanations serve as a detailed guide, illustrate the logical reasoning behind each step and contribute to the comprehension of the entire solving process.

Equation Solutions (EQN) The second view, **eq**uatio**n** solutions (EQN), presents each question's solution as an equation assembled from a series of operators and quantities, without any explanations. Although this view lacks the detailed explanation provided by CoT solutions, it offers a high-level representation of the solution and is one of the most prevalent annotation formats in datasets such as Ape210K, MathQA, and CM17K. It captures the essence of the problem-solving process in the form of a mathematical expression, making it an efficient and effective format to solve certain types of problems.

Solution Tree Pre-order Traversal (TREE) The third view, solution **tree** pre-order traversal (TREE), is an abstract representation of the solution. Widely adopted by math word problem solvers as suggested in previous studies[Zhang et al., 2020, Liang et al., 2022, Jie et al., 2022], it adopts the pre-order traversal of the solution tree, which avoids the use of parentheses and thus further simplifies the solution grammar compared with EQN solutions. More importantly, this form reflects a goal-driven solving strategy aligned with human reasoning[Xie and Sun, 2019]. The expression of solutions in this abstract form fosters efficient solution processing and inference.

Table 1: Examples of three views of mathematical solutions.

Question: Beth bakes 4, 2 dozen batches of cookies in a week. If these cookies are shared amongst 16 people equally, how many cookies does each person consume?	
View	**Solution**
CoT$_{clean}$	Beth bakes 4, 2 dozen batches of cookies for a total of 4*2 = $\ll$4*2=8$\gg$8 dozen cookies. There are 12 cookies in a dozen and she makes 8 dozen cookies for a total of 12*8 = $\ll$12*8=96$\gg$96 cookies. She splits the 96 cookies equally amongst 16 people so they each eat 96/16 = $\ll$96/16=6$\gg$ 6 cookies.
EQN	x = 12*(4*2)/16
TREE	/ * 12 * 4 2 16

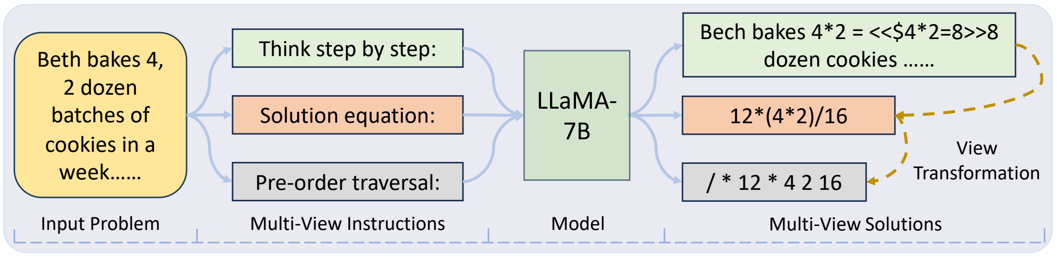


Figure 1: We use MinT to fine-tune a LLaMA-7B model that specializes in math problem solving. First, the original annotation is transformed into multiple different views. Then, the model is trained by instructions to generate different solution forms for one problem.

Noisy Chain-of-Thought Explanations (CoT$_{noisy}$) Besides them, the fourth view, **noisy** chain-of-thought explanations (CoT$_{noisy}$), is similar to CoT$_{clean}$, albeit with noise introduced. This noise may come from incomplete explanations, minor calculation errors, irrelevant domains, or misinterpretation of the problem.

In summary, the CoT$_{clean}$ view provides a detailed explanation, making it the richest in information. It outlines each solution step, mimicking the human method of detailed problem-solving. In contrast, the EQN view is more concise. It captures only the core symbols necessary for the solution, ideal for quick interpretation and computation. Then, the TREE view further simplifies equation representation using a hierarchical approach, which is more concise and coherent. Our multi-view learning framework leverages these different solution forms. By training the model on multiple views, it gains a broader and deeper understanding. This is similar to how teachers encourage students to consider multiple solution paths to understand problems better. As a result, our approach can enhance the model's problem-solving capabilities.

View Transformation As shown in Table 1, the CoT$_{clean}$ view contains both equations and explanations. Therefore, we can simply extract all the equations from the CoT$_{clean}$ view using rule-based detection, and then we combine and transform them into the EQN view. Apart from that, the third view, TREE, can be derived from the EQN view, through well-defined algorithms such as Wang et al. [2018].

2.2 Multi-View Fine-Tuning

Our approach leverages a method we refer to as **M**ulti-V**i**ew Fi**n**e-**T**uning (MinT 🍀), which guides the model in generating different views of solutions by postpending specific instruction strings to the input questions. This results in multiple unique concatenated instructions for each problem, each guiding the model to produce a corresponding view of the answer, as shown in Figure1.

3 Experiments

Our experiments mainly aims to answer the following research questions:

RQ1*: How does MinT affect the performance of a mathematical reasoning model trained on a single dataset with different views when in comparison to individual fine-tuning on each view?*

RQ2*: How does MinT affect the performance of a mathematical reasoning model trained across different datasets with different views?*

RQ3*: What is the effect of introducing additional noisy training data?*

3.1 Results

3.1.1 Generalization Across Different Views on One Dataset (RQ1)

The GSM8k dataset is selected for this investigation, which is annotated by the CoT$_{clean}$ view, which can be conveniently transformed into EQN and TREE views, thus offering a suitable platform for our investigation. As a baseline, we consider the performance of models that have been fine-tuned on GSM8k using each of the three views individually. Then, we introduce a model that has been fine-tuned with our proposed approach and evaluate it on different views by postpending different instructions.

Table 2 shows that augmenting the data with additional views improves performance on all views. Fine-tuning on the original achieves 35.4% accuracy on the set. Adding EQN and Tree views during training boosts to 36.5%, a 1.1% absolute improvement. More substantial gains are observed on the Tree view (from 32.3% to 37.8%). We can also notice that the performance on TREE view is the best, where the potential reason is that this view has the simplest grammar, hence it is the easiest view for the model to learn its pattern. Also, we note that prior work utilizes additional high-quality data generated by LLMs. As such, we would like to clarify that our work is orthogonal to previous results and thus, we are not intending to make comparisons, though including them as reference points may be informative. In fact, our MinT can seamlessly combine with those data-centric methods, by simply assigning additional views for the generated data from stronger models.

Table 2: Results on different views of GSM8k.

Prior Work		**Accuracy**
Magister et al. [2022] (T5-11B)		38.2%
Yuan et al. [2023] (LLaMA2-7B)		50.3%
Luo et al. [2023] (LLaMA2-7B)		54.9%
Yue et al. [2023] (CodeLLaMA-7B)		58.8%
Train View	**Test View**	**Accuracy**
CoT_{clean}	CoT_{clean}	35.4%
EQN	EQN	30.5%
TREE	TREE	32.3%
CoT_{clean}+EQN	CoT_{clean}	35.9%
CoT_{clean}+EQN	EQN	36.2%
CoT_{clean}+EQN+TREE	CoT_{clean}	36.5%
CoT_{clean}+EQN+TREE	EQN	36.9%
CoT_{clean}+EQN+TREE	TREE	**37.8%**

3.1.2 Generalization Across Different Datasets with Different Views (RQ2)

For this investigation, we utilize four different datasets: GSM8k, MathQA, CM17k, and Ape210k. Our baseline for comparison involves prior best results, the single dataset fine-tuning on LLaMA-7B, and simply merging all four datasets for fine-tuning on LLaMA-7B, as shown in Table 3. The results show that straightforward merging cannot bring any improvements. Contrastively, it even has a negative effect. Alternatively, with our multi-view learning approach to these four datasets, the model obtains a general improvement across all views when additional training data is added. Another notable finding is that the performance of the CoT_{clean} view on the GSM8k dataset gets improved by multi-view fine-tuning with the other three datasets, even the additional datasets actually do not provide any supplementary data in the CoT_{clean} view. This outcome shows a promising generalization ability, illustrating the effectiveness of MinT in better leveraging diverse datasets.

Table 3: Experimental results showing the performance of LLaMA-7B with different fine-tuning methods across four datasets. The simple dataset mixture means the training data is mixed only with their original views. For our method MinT, we train our model and report the performance on all available views. The first column shows the training datasets that are used. Prior best results are: a: Magister et al. [2022], b: Liang et al. [2022], c: Qin et al. [2021], d: Zhao et al. [2020].

	GSM8k			MathQA		CM17k	Ape210k	
Single Dataset Baselines								
Prior Best	38.2[a]			76.6[b]		54.1[c]	70.2[d]	
Single Dataset	35.4			79.9		70.1	74.0	
Simple Dataset Mixture								
GSM8k+MathQA	36.7			79.7		-	-	
Ape210k+CM17k	-			-		76.0	74.9	
All Four Datasets	35.3			81.0		68.9	74.1	
Multi-V**i**ew Fi**n**e-**T**uning (MinT ☘)								
	CoT_{clean}	EQN	TREE	EQN	TREE	EQN	EQN	TREE
GSM8k+MathQA	36.8	35.8	38.1	79.7	80.5	-	-	-
Ape210k+CM17k	-	-	-	-	-	77.1	75.9	74.3
All Four Datasets	38.8	39.2	**40.8**	81.0	**81.3**	**77.6**	**76.0**	74.3

Furthermore, we observed that the accuracy improvement between the single dataset baseline and our method is more obvious on the GSM8k and CM17k in comparison to the MathQA and Ape210k. A possible explanation could be that MathQA and Ape210k already contain a substantial number of training problems, thereby enabling the learning of problem patterns and solving skills directly from their training sets. Consequently, the contribution of external datasets may not be significant in this case. However, for the more challenging GSM8k and CM17k datasets, our multi-view training could enhance accuracies more effectively. Furthermore, it can be observed that the EQN view performs optimally on the Ape210k dataset, which is different from GSM8k. This could potentially be attributed to the fact that the solutions in Ape210k comprise fewer steps, resulting in relatively simpler equations compared to those in GSM8k and MathQA. Consequently, converting these equations into tree traversals may not substantially simplify the solutions, thereby not improving the model performance. The above two behavior patterns are also echoed in Table 4.

3.1.3 Generalization on CoT$_{noisy}$ View (RQ3)

In order to further understand the effects of incorporating external noisy training data, we introduce two additional datasets - ASDiv-CoT and ExamQA. The former provides CoT explanations to problems within the ASDiv dataset, though approximately 30% of these CoTs are incorrect. The latter, ExamQA, provides CoT to multi-subject exam problems, and while the solutions provided are accurate, a large number of them are less related to mathematical reasoning. Table 4 presents our experimental results: when we directly add the two noisy datasets for training, there is a slight decrease in accuracy. However, with a specific postfix to differentiate them from the other three views, the overall performance shows an improvement, which demonstrates the potential of using external noisy data to improve the performance on specific downstream tasks. In addition, the results in Table 3 and 4 indicate that multilingual data can also complement each other and help improving the general reasoning ability.

Table 4: Experimental results showing the performance of LLaMA-7B with different fine-tuning methods. Four datasets indicate the combination of four clean datasets - GSM8k, MathQA, CM17k and Ape210k, while two noisy datasets are ASDiv-CoT and ExamQA.

	GSM8k			MathQA		CM17k	Ape210k	
Simple Dataset Mixture								
Four Datasets	35.3			81.0		68.9	74.1	
Four Datasets + Two Noisy Datasets	31.9			79.7		71.3	73.2	
Multi-V**i**ew Fi**n**e-**T**uning (MinT)								
	CoT$_{clean}$	EQN	TREE	EQN	TREE	EQN	EQN	TREE
Four Datasets	38.8	39.2	40.8	81.0	81.3	77.6	76.0	74.3
Four Datasets + ASDiv-CoT	39.0	39.7	42.2	81.4	81.8	78.2	76.4	75.2
Four Datasets + ExamQA	38.6	38.8	41.0	81.0	82.2	78.1	76.6	75.4
Four Datasets + Two Noisy Datasets	39.2	39.7	**42.4**	82.0	**82.3**	**78.8**	**77.0**	76.1

4 Conclusion

In this paper, we propose MinT, a novel multi-view fine-tuning approach to enhance the mathematical reasoning capabilities of language models. By framing diverse annotation formats across datasets as distinct "view" of solutions, our method enables models to learn from these unique problem-solving perspectives.

Broader Impact We believe MinT provides a scalable and flexible approach for specialized LMs by supervised fine-tuning, with the potential for broader applicability beyond mathematical domains. Many other reasoning tasks, such as commonsense or symbolic reasoning, can be solved through diverse paths. Investigating how to leverage MinT for general flexible reasoning is an exciting future direction.

Furthermore, MinT demonstrates effective control over language model fine-tuning. By guiding the model with simple instruction strings, we can take advantage of different types and even incomplete and irrelevant data, while still performing well for downstream tasks. This opens possibilities for future design of large-scale general instruction tuning and task-specific fine-tuning.

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *NAACL-HLT*, pages 2357–2367, 2019.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL `https://lmsys.org/blog/2023-03-30-vicuna/`.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. Specializing smaller language models towards multi-step reasoning. *ICML*, 2023.

Namgyu Ho, Laura Schmid, and Se-Young Yun. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*, 2022.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. Learning to solve arithmetic word problems with verb categorization. In *EMNLP*, pages 523–533, 2014.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

Zhanming Jie, Jierui Li, and Wei Lu. Learning to reason deductively: Math word problem solving as complex relation extraction. In *ACL*, pages 5944–5955, 2022.

Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. Parsing algebraic word problems into equations. *TACL*, 3:585–597, 2015.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. Mawps: A math word problem repository. In *NAACL*, pages 1152–1157, 2016.

Nate Kushman, Luke Zettlemoyer, Regina Barzilay, and Yoav Artzi. Learning to automatically solve algebra word problems. In *ACL*, pages 271–281, 2014.

Zhongli Li, Wenxuan Zhang, Chao Yan, Qingyu Zhou, Chao Li, Hongzhi Liu, and Yunbo Cao. Seeking patterns, not just memorizing procedures: Contrastive learning for solving math word problems. In *Findings of ACL*, pages 2486–2496, 2022.

Zhenwen Liang, Jipeng Zhang, Lei Wang, Wei Qin, Yunshi Lan, Jie Shao, and Xiangliang Zhang. Mwp-bert: Numeracy-augmented pre-training for math word problem solving. In *Findings of NAACL*, pages 997–1009, 2022.

Zhenwen Liang, Wenhao Yu, Tanmay Rajpurohit, Peter Clark, Xiangliang Zhang, and Ashwin Kaylan. Let gpt be a math tutor: Teaching math word problem solvers with customized exercise generation. *arXiv preprint arXiv:2305.14386*, 2023.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*, 2022.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing english math word problem solvers. In *ACL*, pages 975–984, 2020.

Swaroop Mishra, Matthew Finlayson, Pan Lu, Leonard Tang, Sean Welleck, Chitta Baral, Tanmay Rajpurohit, Oyvind Tafjord, Ashish Sabharwal, Peter Clark, et al. Lila: A unified benchmark for mathematical reasoning. In *EMNLP*, pages 5807–5832, 2022.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*, 2022.

Jinghui Qin, Xiaodan Liang, Yining Hong, Jianheng Tang, and Liang Lin. Neural-symbolic solver for math word problems with auxiliary tasks. In *ACL*, pages 5870–5881, 2021.

Subhro Roy and Dan Roth. Solving general arithmetic word problems. In *EMNLP*, pages 1743–1752, 2015.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. Distilling multi-step reasoning capabilities of large language models into smaller models via semantic decompositions. *arXiv preprint arXiv:2212.00193*, 2022.

Minghuan Tan, Lei Wang, Lingxiao Jiang, and Jing Jiang. Investigating math word problems using pretrained multilingual language models. *MathNLP 2022*, page 7, 2022.

Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. Translating a math word problem to a expression tree. In *EMNLP*, pages 1064–1069, 2018.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*, 35: 24824–24837, 2022.

Zhipeng Xie and Shichao Sun. A goal-driven tree-structured neural model for math word problems. In *IJCAI*, pages 5299–5305, 2019.

Dian Yu, Kai Sun, Dong Yu, and Claire Cardie. Self-teaching machines to read and comprehend with large-scale multi-subject question-answering data. In *Findings of EMNLP*, pages 56–68, 2021.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*, 2023.

Jiaxin Zhang and Yashar Moshfeghi. Elastic: numerical reasoning with adaptive symbolic compiler. *NeurIPS*, 35:12647–12661, 2022.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. Graph-to-tree learning for solving math word problems. In *ACL*, pages 3928–3937, 2020.

Wei Zhao, Mingyue Shang, Yang Liu, Liang Wang, and Jingming Liu. Ape210k: A large-scale and template-rich dataset of math word problems. *arXiv preprint arXiv:2009.11506*, 2020.

Appendix

4.1 Our Method

Formally, each question Q is paired with an instruction string p_i drawn from the set $\mathcal{P}$, which includes all possible instructions. When Q is concatenated with p_i, it results in a unique string for each question. This provides the necessary guidance for the model to generate a corresponding answer a_i from the answer set $\mathcal{A}$. As such, for each question, we formulate multiple sequences $s_i = Q + p_i + a_i$. Consequently, during the training phase, the model processes a large number of these sequences s_i, enhancing its understanding and generalization across multiple views.

To optimize the model, the next-word-prediction loss L is calculated for each sequence:

$$L(s_i) = -\sum_{j=1}^{len(s_i)-1} \log P(s_{i,j+1}|s_{i,1:j};\Theta), \tag{1}$$

where P denotes the model's conditional probability distribution over the next token, facilitated by the Softmax function of the model's logits, $s_{i,j}$ represents the j^{th} token of sequence s_i, and Θ embodies the model parameters. However, to enhance the model's focus on generating accurate answers, we exclusively backpropagate the loss calculated on the answer part, denoted as L_{a_i}:

$$L_{a_i}(s_i) = -\sum_{j=len(Q+p_i)+1}^{len(s_i)-1} \log P(s_{i,j+1}|s_{i,1:j};\Theta). \tag{2}$$

It ensures that the model focuses on learning to produce precise answers, contributing to its mathematical reasoning ability. During the evaluation, we adopt the same instruction concatenation and assess the model's performance on each individual view.

4.2 Mathematical Datasets for Training and Testing

GSM8k *(CoT$_{clean}$, EQN, TREE)* The GSM8K dataset Cobbe et al. [2021] is a curated set of 8.5K high-quality elementary-level math word problems in English, authored by human problem writers. It is split into approximately 7.5K problems for training and 1K for testing purposes. The problems are annotated with their comprehensive step-by-step solutions, providing the Clean Chain-of-Thought Explanations (CoT$_{clean}$) view.

MathQA *(EQN, TREE)* MathQA [Amini et al., 2019] contains English mathematical problems from GRE examinations. Nevertheless, some of the problems in this dataset have quality concerns. Several efforts [Tan et al., 2022, Li et al., 2022, Liang et al., 2022] have been conducted to cleanse and filter the MathQA dataset. In our experiment, we adopt the version referenced in Liang et al. [2022], wherein all solutions are re-annotated by an equation composed of the four arithmetic operators and numbers, reflecting the Equation Solutions (EQN) view and we also transform that to the TREE view.

Ape210k *(EQN, TREE)* The Ape210k dataset Zhao et al. [2020] is a large-scale, template-rich collection of math word problems (MWPs) in Chinese, containing 210,488 problems and 56,532 solution templates. The view of the solutions in Ape210k mirrors that in MathQA. Our experiment incorporates its 200K training problems and 50K testing problems.

CM17k *(EQN)* The CM17K dataset [Qin et al., 2021] comprises four types of Chinese MWPs (arithmetic, one-unknown linear, one-unknown non-linear, equation set), which is different from MathQA and Ape210k. Therefore, we only have the EQN view for the solutions in this dataset.

4.3 Additional Noisy Datasets for Training

ASDiv-CoT *(CoT$_{noisy}$)* The ASDiv dataset [Miao et al., 2020] consists of 2,305 English MWPs that are diverse in language patterns and problem types. We employ the few-shot CoT predictions of GPT-3 provided by Wei et al. [2022][1] on this dataset as one of the CoT$_{noisy}$views for training.

[1]`https://github.com/jasonwei20/chain-of-thought-prompting`.

With an accuracy of 71.3%, approximately 30% of the predictions are spurious. The inclusion of this dataset shows the adaptability and broad applicability of our method to inaccurate LLM-generated data.

ExamQA ***(CoT$_{noisy}$)*** The ExamQA dataset [Yu et al., 2021] is a comprehensive Chinese dataset of real-world exams, containing 638k multiple-choice instances across various subjects (e.g., sociology, education, and psychology). We manually filter a subset with 20k problems that contain numbers and equations in their answers by hand-crafted rules. Despite each problem in this subset being annotated with its ground truth and step-by-step solutions, we inevitably introduce many problems that are less relevant to the math subject. This dataset also serves as one of the CoT$_{noisy}$ views, also showing the generalizability of our approach.

4.4 Evaluation on Held-out Dataset

In order to further assess the multi-view problem solving abilities of our method, we evaluated it on the held-out dataset, MAWPS Koncel-Kedziorski et al. [2016], which contains 2,373 English MWPs annotated with Equation Solutions (ES) view. It integrates several earlier datasets in Hosseini et al. [2014], Kushman et al. [2014],Koncel-Kedziorski et al. [2015] and Roy and Roth [2015] and thus serves as a comprehensive benchmark. Three training data settings are used: GSM8k only, four clean datasets, and all six datasets, where respective models are all trained with MinT.

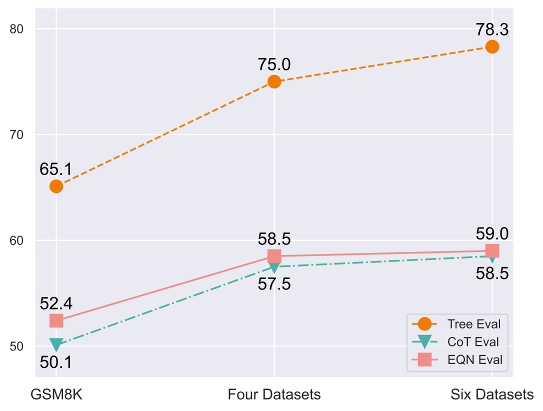


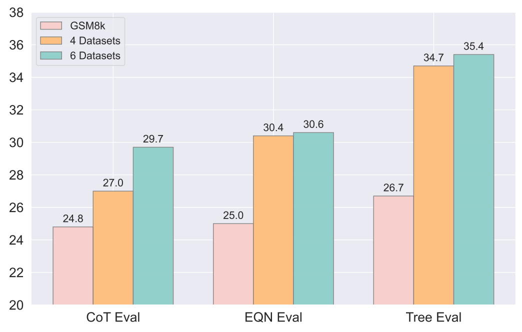
Figure 2: Experimental results on the MAWPS dataset. X-axis indicates the training datasets and Y-axis indicates the accuracy.

Our results in 2 are similar to the observations from our previous experiments: increasing the number of datasets used in training boosts performance across all views. It is noteworthy that although the MAWPS dataset is originally annotated using the EQN view, our model manages to attain an accuracy of 58.5% when attempting to solve problems using step-by-step CoTs. This finding indicates that the problem solving ability acquired from the training datasets can indeed be transferred to the held-out datasets. More interestingly, it suggests that our method could serve tasks like multi-view data annotation.

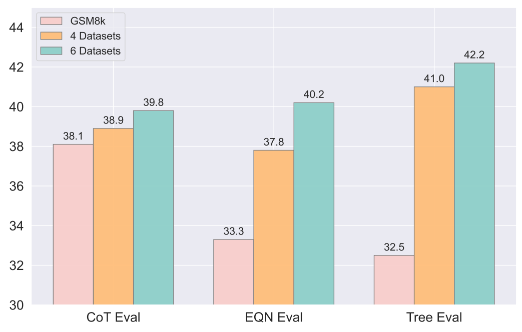
4.5 Adaptivity on Different Backbones

In this experiment, we replace our original backbone model LLaMA-7B with other two state-of-the-art model, namely BLOOMz-7B Muennighoff et al. [2022] and Vicuna-7B Chiang et al. [2023]. This substitution allows us to observe how well our method adapts to different language model architectures. We also employ three models for this task: one trained on the GSM8k dataset, another trained on a combination of four clean math datasets, and a third trained on a total of six datasets. For each of these models, multi-view fine-tuning is employed during the training process.

As illustrated in Figure 3, our method demonstrates the same pattern on both backbones compared to the LLaMA-7B backbone, i.e., incorporating more training data can further enhance performance with the aid of multi-view training. Also, it is notable that the Vicuna backbone has a better performance on the CoT$_{clean}$ view, this is because the Vicuna model is more familiar with the "explanation" style data than symbolic equations by continual fine-tuning on dialogues. This means that the success of our method is not restricted to one specific model, it also extends to other language models and benefits from their own characters. This consistent performance across different backbones validates the robustness of our approach and supports its potential applicability in a wider range.



(a) Results on BLOOMz-7B.



(b) Results on Vicuna-7B.

Figure 3: Experimental results with different backbones on GSM8K. X-axis indicates the evaluation views and Y-axis indicates the accuracy.

4.6 Ablation Study

To demonstrate the effectiveness of our proposed approach. We implement three ablated studies: *1) Only use the original view for every dataset, and postpending the corresponding instructions to the training samples; 2) Only use the EQN view from GSM8k, MathQA, CM17k and Ape210k and keep the instructions; 3) Only use the* TREE *view from GSM8k, MathQA and Ape210k and also keep the instructions.*

Table 5: We use GSM8k, MathQA, CM17k and Ape210k in our ablation study with different training strategies described in 4.6 in *Italics*. The first ablation (a1) aims to investigate the impact of the instructions, while the rest two ablations (a2, a3) can examine the improvement brought by the generalization from other views.

	GSM8k	MathQA	CM17k	Ape210k
Simple Dataset Mixture	35.3	81.0	68.9	74.1
a1. Original View with Instructions	35.5	80.2	76.6	74.7
a2. Only EQN View with Instructions	34.6	79.9	77.5	74.0
a3. Only TREE View with Instructions	39.6	81.0	-	75.2
MinT (Our method)	**40.8**	**81.3**	**77.6**	**76.0**

As shown in Table 5, the first ablation proves that the instructions can bring some improvements on certain datasets. And the other two ablations show that unifying the views of solution can also bring some improvements to the performance. Nonetheless, when compared with these baselines and ablations, our proposed MinT performs the best, thereby reaffirming that MinT truly capitalizes on the generalization across diverse solution views of mathematical problems.

4.7 Case Study

Figure 4 illustrates a sample problem from the MAWPS dataset, which is fed into three distinct models discussed in Section 4.4. The first model yields an completely wrong solution, possibly due to the significant disparities in solution patterns between GSM8k and MAWPS, leading to the model's inability to generalize for problems in the held-out set. The second model's solution, though only partially correct and incomplete, suggests an improvement in its reasoning capabilities. The third model, trained with all six datasets, effectively solves the problem in this example, thereby affirming the efficacy of our methodology in improving mathematical reasoning ability.

4.8 Instructions

In this paper, we propose to use different instructions to guide the generation of distinct solution views. The specific instructions that we used are listed in Figure 5. It is notable that We separate the English and Chinese instructions, which we find helpful to the final performance. The possible

Problem:	Only GSM8k Model	Four Datasets Model	Six Datasets Model
Megan had 40 homework problems . She finished 26 of them but still had 2 pages of problems to do . If each page has the same number of problems on it , how many problems are on each page ?	Megan finished 26 / 40 = <<26/40=.65>>.65 pages. There are 2 /.65 = <<2/.65=3>>3 problems on each page. ✗	Since Megan had 40 problems and she has finished 26 of them, she still has to do 40-26 = <<40-26=14>>14 problems. ✗	She still had to do 40-26 = <<40-26=14>>14 problems. There are 2 pages of problems so each page has 14/2 = <<14/2=7>>7 problems. ✓

Figure 4: Our case study on the held-out dataset - MAWPS.

reason is that mixed languages are relatively rare in the pre-training corpus and thus the LMs are not familiar with this pattern.

View	Instruction (English)	Instruction (Chinese)
$\text{CoT}_{\text{Clean}}$	Think step by step:	逐步求解:
EQN	Solution Equation:	列等式求解:
TREE	Pre-order Transversal of Solution Tree:	列前序遍历表达式求解:
$\text{CoT}_{\text{Noisy}}$	General Idea:	大体思路:

Figure 5: This table shows the instructions we used for different views and languages.

4.9 Discussion

4.9.1 Limitations and Future Work

Nonetheless, our work is not without limitations. While we demonstrate generalization on held-out datasets, evaluation on more diverse views and tasks would further validate the capabilities of our proposed method. Also, there may be optimal combinations and proportions of data across views that our current work does not explore in depth. Another unexplored aspect is whether multiple views can augment each other through shared learning, potentially enhancing the overall accuracy. Techniques such as ensemble learning or majority voting could provide avenues for further improvement in this regard. Last but not least, there are more views that could be integrated into MinT training, such as programs Amini et al. [2019], Zhang and Moshfeghi [2022], and Python codes Mishra et al. [2022], Chen et al. [2022]. We leave these possibilities in our future work.