

SciGLM: Training Scientific Language Models with Self-Reflective Instruction Annotation and Tuning

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have shown promise in assisting scientific discovery. However, such applications are currently limited by LLMs’ deficiencies in understanding intricate scientific concepts, deriving symbolic equations, and solving advanced numerical calculations. To bridge these gaps, we introduce SciGLM, a suite of scientific language models able to conduct college-level scientific reasoning. Central to our approach is a novel self-reflective instruction annotation framework to address the data scarcity challenge in the science domain. This framework leverages existing LLMs to generate step-by-step reasoning for unlabelled scientific questions, followed by a process of self-reflective critic-and-revise. Applying this framework, we curated SciInstruct, a diverse and high-quality dataset encompassing physics, chemistry, math, and formal proofs. We fine-tuned the ChatGLM family of language models with SciInstruct, enhancing their capabilities in scientific and mathematical reasoning. Remarkably, the SciGLM consistently improves both the base model (ChatGLM3-6B-Base) by 4.87% and larger-scale models (32B) by 2.67%, without sacrificing the language understanding capabilities of the base model. This makes SciGLM a suitable foundational model to facilitate diverse scientific discovery tasks. For the benefit of the wider research community, we release SciInstruct, and SciGLM, alongside a self-reflective framework and fine-tuning code.

1 Introduction

Large language models (LLMs) have shown potential to assist and accelerate scientific discovery (Taylor et al., 2022; AI4Science and Quantum, 2023), helping tasks like protein prediction (Chen et al., 2023a), weather forecasting (Bi et al., 2023) and geoscience understanding (Deng et al., 2023). Despite these promising proof-of-concept trials, recent studies (Sun et al., 2023; Wang et al., 2023c;

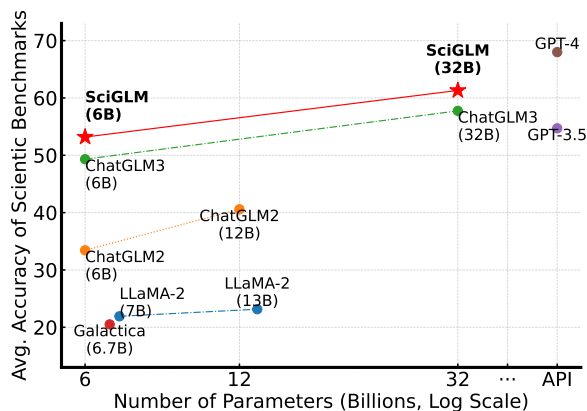


Figure 1: Average accuracy on CEval-Sci, SciEval, MATH, and SAT-Math benchmarks of different LLMs. Our SciGLM models fine-tuned from ChatGLM3 achieve consistent improvement for different parameter sizes.

Rein et al., 2023) show that even advanced LLMs like GPT-3.5 and GPT-4 struggle with basic scientific problems, achieving only 28.52% accuracy on some college-level textbook questions. These scientific questions, such as calculating energy with the Planck distribution, require a diverse set of skills, including finding the correct combination of physical concepts and axioms, choice and deduction of formal equations, and rigorous numerical computing. Before letting LLMs equip these skills to solve basic scientific questions, all ambitious visions of building LLM agents to assist scientific discovery could be unreliable. This brings a substantial incentive for developing a foundational scientific language model.

However, training LLMs to understand science (e.g., physics, chemistry, math) is much more challenging than many general reasoning tasks, requiring more complicated skills. Therefore, the core of improving LLMs’ scientific problem-solving capabilities is to build large-scale and high-quality instruction datasets, which shall cover all the required skills. We thus aggregate data to improve each skill: 1) for scientific concept understanding, we gather a

substantial amount of physics and chemistry questions that require basic science knowledge; 2) for numerical calculation, we crawl and utilize additional and more advanced mathematical calculation data; 3) for rigorous deduction of symbolic equations, we incorporate formal theorem proofs written in Lean. Such a mixture of data sources enables the trained model not to overfit to a single subject but to acquire some general and fundamental skills to solve different scientific tasks.

On the other hand, the scale of available instruction data from the internet for scientific problems is way smaller than other tasks. As scientific content often requires certain expertise to create, and high-quality information is often protected by intellectual property, most data we can legally access only contain question-answer (QA) pairs without detailed chain-of-thought reasoning steps (R). However, merely training LLMs on QA pairs will lead to very bad results and even harm their general language capability. To get high-quality reasoning steps (R) as instruction, we propose a self-reflective instruction annotation framework that asks LLM to autonomously annotate, critique, and revise reasoning steps, with minimal human intervention. Specifically, LLM first tries to generate both reasoning steps and answer the given question (Q) only; then, for those outputs with incorrect answer prediction, we ask LLM itself to identify the error type, based on which to address the error and revise the output, until getting the correct answer. Such a self-reflective annotation framework solely utilizes AI rather than humans to collect reasoning traces (R) as instructions, while guaranteeing the quality and addressing the potential mistakes of existing LLM with careful answer checking and LLM self-reflection.

After consolidating the questions and answers produced by self-reflective annotation, we construct SciInstruct, a comprehensive dataset instruction tuning a scientific language model. In this work, we choose the bilingual ChatGLM (Du et al., 2022; Zeng et al., 2022) series model, an open-source dialogue pre-training model with low deployment barriers, as our backbone base model. By fine-tuning the ChatGLM series model on SciInstruct, we obtain the SciGLM model. We evaluate SciGLM through three types of evaluation tasks, including **scientific test sets**, **mathematical benchmarks**, and **general language and coding tasks**. Through instruction tuning, we achieve a

4.87% improvement over the 6B model, and 2.67% improvement over the 32B model, outperforming many previous state-of-the-art models with the same parameter size, including Galactica (Taylor et al., 2022) for science problems, and MAMmoTH (Yue et al., 2023) for math problems. We also show tuning our instruction datasets does not sacrifice general language understanding capabilities, making SciGLM a good suite of scientific language models for both human-AI communication as well as domain-knowledge expertise.

We highlight our contributions are listed below from three perspectives:

- From the data perspective, we construct SciInstruct, a comprehensive scientific instruction tuning dataset that includes physics, chemistry problems, math, and formal proofs.
- From the method perspective, we propose a self-reflective annotation pipeline for LLMs to autonomously curate instruction datasets.
- From the model perspective, we utilize the ChatGLM series model as the backbone, training SciGLM on SciInstruct, and show performance improvements up to 3.06% and 6.68% over the 6B model on various scientific and mathematical benchmarks, without sacrificing general language understanding tasks.

2 SciInstruct Dataset Construction

Many research (Zeng et al., 2023; Yue et al., 2023) have shown that fine-tuning pre-trained LLMs on high-quality CoT reasoning data can gain performance improvement by enabling the model to better utilize the knowledge memorized through pre-training, follow more accurate and human-readable reasoning styles and language formats. However, the main challenges of constructing scientific instructions include the knowledge and complexity required and the smaller scale of available data.

Therefore, we seek to tackle these obstacles by creating SciInstruct to enhance the LLMs’ scientific problem-solving capabilities. Figure 2 illustrates our meticulous design of essential sub-modules aimed at gathering large-scale, high-quality instructions. These critical sub-modules encompass self-reflective instruction annotation and noisy instruction filtering. SciInstruct comprises a total of **254,051** verified instructions.

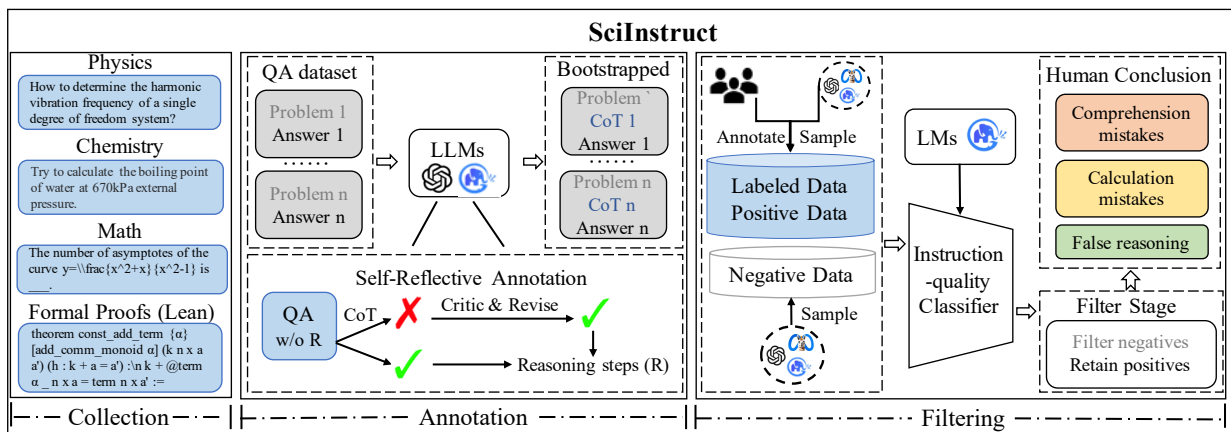


Figure 2: **The pipeline of constructing SciInstruct.** On the far left is a mix of training datasets. The purpose of the annotation is to supplement chain-of-thought processes with reflective generation. The goal of the filter is to train an instruction-quality classifier and only keep high-quality reasoning traces as instructions.

2.1 Diverse Instruction Collection

Our objective is to build a comprehensive and diverse dataset that encompasses scientific knowledge in terms of depth, wide coverage, and diversity. To achieve this, we will focus on scientific fields and curate several top-tier datasets that are extensively utilized and cover a wide range of scientific disciplines, such as physics, chemistry problems, mathematics, and formal proofs. To initiate the process, we collect questions from a variety of sources, including textbooks, pedagogical materials, and problem sets.

Instruction Subject. As show on the left side of Figure 2, we create data from the following subject:

- **Physics.** This subject aims to address the challenge of processing complex physical problems with step-by-step solutions and assessing the ability of LLMs to comprehend and analyze physics problems. It is observed that public training datasets lack college-level physics knowledge, such as Fundamentals of Physics and Physical Chemistry. To address this gap, we collected a large set of physical questions from various subjects (e.g., dynamics, quantum physics, electrodynamics, etc.) from typical physical textbooks, along with a comprehensive dictionary of physics problems and solutions. Note that most of these physics questions only contain a single answer without a step-by-step solution.

- **Chemistry.** We gathered questions from various chemistry subjects, including college-level General Chemistry, Analytical Chemistry, Biochemistry, and Inorganic Chemistry. Unlike Physics, these Chemistry problems emphasize more on knowledge concepts and require less calculation.

- **Math.** College-level mathematical tasks are characterized by their intricate logic and high complexity. We have gathered a wide array of math problems encompassing various mathematical subjects such as Calculus, Algebra, and Advanced Mathematics. These problems come in different formats, including multi-choice questions, calculations, and complex problem-solving scenarios. Our collection process involves sourcing problems from public Q&A websites, problem sets, and textbooks.

- **Formal Proofs (Lean).** SciInstruct also includes data from formal mathematics. In formal math, theorems and proofs are expressed in formal logic instead of natural language. They look like computer programs. For example, “theorem gcd_self (n : Nat) : gcd n n = n” is a theorem that says: For any natural number n, the greatest common divisor of n and itself is n. And “cases n <=> simp [gcd, mod_self]” is a proof of this theorem. There are many languages for writing formal theorems and proofs; examples include Coq (Barras et al., 1997), Isabelle (Nipkow et al., 2002), and Lean (de Moura et al., 2015). We chose to include data from Lean, as it is currently the most popular in the math community. Therefore, it has abundant theorems and proofs written by mathematicians, covering diverse topics in graduate-level mathematics, such as analysis, geometry, and algebra. Specifically, we process the data from LeanDojo (Yang et al., 2023) and obtain 40,248 instructions for theorem proof. Like 7.1, we form an instruction for each theorem and its proof.

Summary. In total, we gathered 257,143 raw questions, the majority of which lacked step-by-step rea-

soning steps. We aim to supplement these through a self-reflective annotation process.

2.2 Self-Reflective Instruction Annotation

Given a language model π to answer question Q , recent studies (Wei et al., 2022a) have shown that by first forcing it to generate step-by-step reasoning steps (R) first, the overall performance for correctly generating answer A can be significantly improved:

$$P_{\pi}(A | Q) = \mathbb{E}_{R \sim P_{\pi}(R|Q)} P(A | R, Q) \quad (1)$$

This is why many instruction datasets aim to collect high-quality intermediate solutions to train LLMs generating correct step-by-step solutions. The key challenge for the science domain, as we state above, is that most of the QA pairs we collect do not contain ground-truth reasoning paths (R). Getting the correct intermediate reasoning R given QA can be regarded as a discrete latent variable inference problem via posterior sampling. However, in practice, we cannot afford to sample all possible R from LLM. Here we adopt a simplified strategy for solving it: 1) Utilizing a powerful LLM (we use GPT-4-0613), we sample multiple times for each question, recording its reasoning traces as well as the predicted answer; 2) We filter out only the traces with a correct predicted answer, by assuming only with correct traces, LLM can get a correct answer.

LLM Self-Reflection. However, even GPT-4 cannot consistently produce accurate answers after multiple trials, and the above procedure can only collect R for a portion of questions. Drawing on prior research demonstrating the capacity of language models for self-correction (Shinn et al., 2023; Li et al., 2023a), we refine the CoT process using a self-reflective framework, as depicted in the middle of Figure 2. The final reflective generation process entails three stages. Initially, we employ a simple CoT prompt (Prompt 1) to obtain step-by-step solutions for each question. By applying the GPT-4 labeling method, we filter out incorrect solutions, resulting in 19,824 correct solutions. Subsequently, the incorrect solutions from stage one, along with their respective questions, enter stage two. Here, a reflective prompt (Prompt 2) assists GPT-4 in generating correct answers by analyzing its errors from stage one. The newly generated solutions are then filtered again by GPT-4, and the undesirable ones proceed to stage three. In stage three, based on Prompt 2, we incorporate the

Type	Ave. Sci	Ave. Math	Ave. {Sci+Math}
Unfiltered	43.57	48.50	46.03
Filtered	43.85	49.24	46.54

Table 1: **Ablation study of filter step.** We arranged the samples by score and excluded the lowest 10%. The resulting numbers represent the average weighted accuracy on evaluation tasks.

real answer as a direct hint in the prompt (Prompt 3) to further aid in question resolution. The final segment of correct solutions is obtained after the generation and filtering process. We illustrate the reflective generation in Figure 3 and quantify the amount of data generated by each process.

2.3 Filtering Out Low-Quality Instruction

Though the above procedures give us the annotated reasoning steps (R), not all of them are correct. The error can come from two sources: 1) though LLM generates a correct answer, the intermediate reasoning can still be wrong (Lanham et al., 2023); 2) the question and ground-truth solutions transformed via OCR may be incomplete and unable to be successfully compiled. Therefore, we propose an additional step to train an instruction-quality classifier and filter out low-quality instructions.

Quality Data Synthesis. We randomly selected a subset of questions from our labeled dataset of 11,553 questions as positive samples. To generate negative samples, we prompted ChatGLM2-6B, GPT-3.5-turbo-0613, and GPT-4-0613 to provide step-by-step answers to selected questions. We filtered inaccurate answers from ChatGLM2-6B and labeled the solutions from GPT-3.5-turbo and GPT-4 using a formatted prompt method demonstrated in Figure 7. These solutions were merged with annotated solutions from the original dataset to train our classifier. The composition of the merged dataset is detailed in Table 6.

Instruction-quality Classifier. We improved dataset quality by training an instruction-quality classifier based on Table 6 using ChatGLM3-6B-Base features. Using these data, we train an instruction-quality classifier via the feature pre-extracted by a ChatGLM3-6B-Base model. The classifier outputs a logit ranging from -1 to 1, with higher scores indicating more reliable answers. This logit is used to rank and select high-quality data from the noisy dataset. Table 1 demonstrates the effectiveness of supervised fine-tuning on both

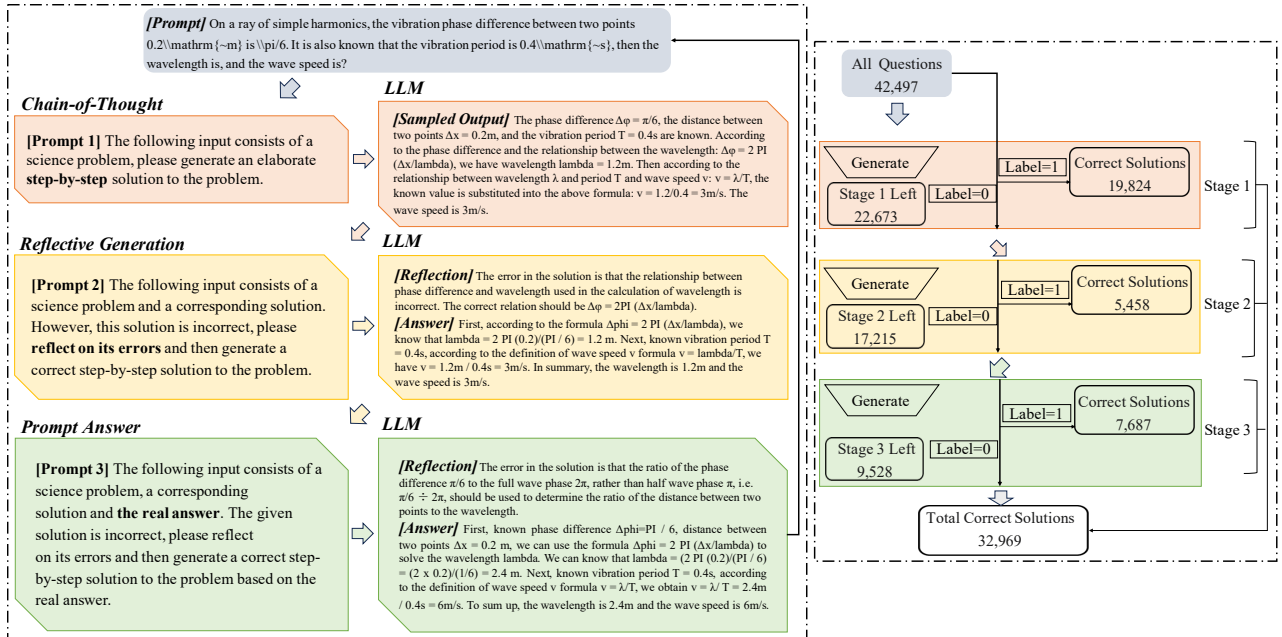


Figure 3: The workflow and quantity of data for self-reflective instruction annotation framework.

Subject	Physics & Chemistry	Math	Formal Proofs (Lean)	Total
# Number	123,869	89,934	40,248	254,051

Table 2: Statistics of SciInstruct across subjects.

323 filtered and unfiltered datasets at a 6B scale, showing
 324 that models trained on a filtered dataset perform
 325 better.

326 **Error Analysis.** As our classifier filter is trained on
 327 labeled datasets and generated solutions, errors in
 328 negatively labeled solutions from ChatGLM2-6B,
 329 GPT-3.5-turbo, and GPT-4 can significantly impact
 330 the classifier’s performance. Therefore, we conduct
 331 an error analysis and categorize them into Compre-
 332 hensive mistakes, Calculation mistakes, and False
 333 reasoning. This analysis is detailed in Figure 8,
 334 demonstrating the classifier’s capacity to recognize
 335 these errors in the dataset.

336 **Summary.** Based on the aforementioned key sub-
 337 modules, we have constructed the SciInstruct
 338 dataset, which comprises 254,051 instructions, as
 339 illustrated in Table 2.

340 2.4 Instruction-Tuning with SciInstruct

341 We choose the ChatGLM series model as our founda-
 342 tional model. More specifically, we have opted
 343 for the open bilingual chat LLM, ChatGLM3-6B-
 344 Base. After establishing the base model, we have
 345 standardized all data into a chatbot-style format.
 346 Subsequently, we have fine-tuned the foundational
 347 model using the SciInstruct, enabling us to val-

348 idate our constructed SciInstruct. Throughout
 349 the fine-tuning process, we have conducted experi-
 350 ments using the Huggingface transformers library.
 351 For both the 6B and 32B models, we have utilized
 352 a learning rate of $3e-6$, employed a linear scheduler,
 353 and trained for two epochs. To efficiently train the
 354 model, we have leveraged DeepSpeed (Rajbhandari
 355 et al., 2020) training.

356 3 Experiments

357 3.1 Experimental Setup

358 **Scientific and Mathematical Tasks.** In Ta-
 359 ble 7, the evaluation tasks are summarized. Ten
 360 tasks have been chosen as out-of-domain bench-
 361 mark datasets, encompassing CEval-Hard (Huang
 362 et al., 2023), CEval-Sci (Huang et al., 2023),
 363 MMLU-Sci (Hendrycks et al., 2020), SciEval (Sun
 364 et al., 2023), SciBench (Wang et al., 2023c),
 365 GSM8K (Cobbe et al., 2021), MATH (Hendrycks
 366 et al., 2021), Mathematics (Davies et al.,
 367 2021), SAT-Math (Zhong et al., 2023), MMLU-
 368 Math (Hendrycks et al., 2020) from MathInstruc-
 369 tion, and CEval-MATH (Huang et al., 2023). These
 370 evaluation datasets cover a broad range of subjects,
 371 including physics, chemistry, and math.

372 **General Tasks.** Furthermore, we assess the gen-
 373 eralization abilities of tasks across various scales
 374 when fine-tuning models. These tasks include as-
 375 sessing knowledge abilities (MMLU (Hendrycks
 376 et al., 2020) and CEval (Huang et al., 2023)) and
 377 code generation (MBPP) (Austin et al., 2021).

Model	CEval-Hard	CEval-Sci	MMLU-Sci	SciEval	SciBench	Avg. Sci	Avg. {Sci+Math}
(API, parameter details unknown)							
GPT-4	54.96	60.55	-	73.93	28.52	-	-
GPT-3.5-turbo	41.37	46.83	-	66.97	12.17	-	-
Claude-v1.3	39.14	44.64	-	63.45	-	-	-
(# parameter = 6B~7B)							
LLaMA-2-7B	28.29 [†]	30.00 [†]	30.41	28.37	0.40	23.49	20.89
Galactica-6.7B	11.84 [†]	11.44 [†]	30.68	50.87	-	-	-
ChatGLM2-6B	29.61 [†]	45.71 [†]	37.09 [†]	53.02 [†]	1.54 [†]	33.39	30.55
ChatGLM2-6B-Base	32.90 [†]	40.95 [†]	38.06 [†]	50.38 [†]	1.20 [†]	32.70	30.56
ChatGLM3-6B	36.84 [†]	38.57 [†]	41.78[†]	56.56 [†]	2.40 [†]	35.23	30.91
ChatGLM3-6B-Base	<u>45.40[†]</u>	<u>54.29[†]</u>	40.16 [†]	<u>61.69[†]</u>	2.40 [†]	<u>40.79</u>	<u>41.67</u>
SciGLM (ChatGLM3-6B-Base)	51.97	60.00	<u>41.40</u>	62.09	3.77	43.85	46.54
(# parameter = 12B~13B)							
LLaMA-2-13B	19.74 [†]	19.05 [†]	35.85	36.96	-	-	-
Vicuna-13B	-	-	32.13	53.93	-	-	-
(# parameter = 30B~32B)							
Galactica-30B	-	-	35.53	54.96	-	-	-
ChatGLM3-32B-Base	53.95 [†]	64.29 [†]	50.30[†]	67.38 [†]	4.29 [†]	48.04	50.77
SciGLM (ChatGLM3-32B-Base)	56.58	66.19	49.38	69.84	5.15	49.43	53.44

Table 3: **Results on scientific reasoning tasks.** Experiments indicate that SciGLM fine-tuning consistently outperforms the base ChatGLM3 across various parameter scales. Avg. Sci represents the weighted average score of all scientific tasks within the same evaluation category, while Avg. {Sci+Math} signifies the weighted average score on both scientific and mathematical tasks. Within each parameter setup, **Bold** spotlights the one with best performance, and Underline denotes the second-best result. Results marked as [†] are benchmarked by ourselves.

Evaluation Metrics. The default setting for the fine-tuned SciGLM inherently provides the CoT solution. Hence, we conduct all experiments using CoT settings. To thoroughly and accurately evaluate the capabilities of different models, we employ the accuracy metric for all tasks except for code generation, for which we use the pass@1 metric.

Baselines. We consider the following baselines, GPT-4 (OpenAI, 2023), GLM (Du et al., 2022; Zeng et al., 2022) and LLaMA (Touvron et al., 2023) Base, Continue Pre-training, and Dataset-specific Tuning. We employ a standardized evaluation framework to compare GLM and LLaMA Base baselines fairly. To gauge performance in the MATH task, we utilize zero-shot and 8-shot configurations to determine the highest accuracy. Additionally, for Mathematics, SAT-Math, MMLU, and CEval, we employ a chat module for assessment. When dealing with multiple-choice questions, we formulate the prompt as "Therefore, among A through D, the answer is".

3.2 Main Results and Analysis

Table 3, Table 4, and Table 5 present experimental findings for scientific and mathematical benchmarks, as well as general tasks. The results show

that training on the provided SciInstruct benefits reasoning tasks, improving performance in scientific reasoning (i.e., CEval-Sci, SciEval, SciBench, MMLU-Sci) and transferring to mathematical tasks. Such performance improvement is consistent with different scales of based model parameters, across 6B and 32B. In addition, SciGLM’s performance improvement in scientific reasoning does not sacrifice its general language understanding capabilities. As shown in Figure 4, on standard tasks like MMLU and CEval, SciGLM even achieves slight performance improvement. On code generation like MBPP, the performance is slightly lower but is overall consistent.

Influence of Data Mixture. We further explore how the diverse subjects within the SciInstruct mixture affect downstream tasks when training the SciGLM-6B model. By employing a Leave-One-Out strategy, i.e., omitting one subject at a time from the dataset and retraining, we assess the significance of each subject based on the performance impact across various tasks. As shown in Figure 5, we have an interesting finding: each subject contributes tasks that are not restricted to its immediate domains. For instance, Physics and Chemistry data significantly aid in CEval-Math tasks, while Math

Model	GSM8K	MATH	Mathematics	SAT-Math	MMLU-Math	CEval-Math	Avg. Math
(API, parameter details unknown)							
GPT-4	92.00	42.50	-	95.00	-	53.91	-
GPT-3.5-turbo	80.80	34.10	-	70.90	-	40.81	-
Claude-v1.3	-	-	-	-	-	37.66	-
(# parameter = 6B~7B)							
LLaMA-2-7B	14.60	2.50	6.00	26.80	29.80	30.00 [†]	18.28
Galactica-6.7B	10.20	2.20	4.60	17.50	28.00	14.48 [†]	-
WizardMath-7B	54.90	10.70	9.30	25.40	31.10	-	-
MAmmoTH (CoT)-7B	53.60	10.40	9.20	32.70	39.90	-	-
MetaMath-7B	66.50	19.80	-	-	-	-	-
MAmmoTH & MetaMath-7B	66.30	24.10	18.30	41.40	<u>44.40</u>	-	-
ChatGLM2-6B	25.85	6.90 [†]	14.30 [†]	39.55 [†]	38.91 [†]	36.67 [†]	27.03
ChatGLM2-6B-Base	31.54	7.84 [†]	17.10 [†]	34.55 [†]	40.45 [†]	32.22 [†]	27.28
ChatGLM3-6B	29.05	9.92 [†]	11.60 [†]	39.09 [†]	41.07 [†]	21.11 [†]	25.31
ChatGLM3-6B-Base	<u>72.93</u>	25.38[†]	<u>29.30[†]</u>	<u>55.91[†]</u>	31.83 [†]	<u>40.00[†]</u>	<u>42.56</u>
SciGLM (ChatGLM3-6B-Base)	73.62	<u>25.18</u>	31.80	65.46	49.38	50.00	49.24
(# parameter = 12B~13B)							
LLaMA-2-13B	28.70	3.90	11.50	32.70	34.40	18.89 [†]	21.68
Vicuna-13B	28.40	5.80	10.00	34.00	34.10	-	-
WizardMath-13B	63.90	14.00	14.10	24.50	32.10	-	-
MAmmoTH (CoT)-13B	62.00	10.20	9.60	40.90	36.60	-	-
MAmmoTH & MetaMath-13B	71.04	26.18	20.60	48.18	48.25	-	-
(# parameter = 30B~32B)							
Galactica-30B	41.70	12.70	11.80	37.70	37.90	-	-
ChatGLM3-32B-Base	<u>81.80</u>	<u>31.60[†]</u>	38.60[†]	<u>67.73[†]</u>	<u>50.10[†]</u>	<u>51.11[†]</u>	<u>53.49</u>
SciGLM (ChatGLM3-32B-Base)	83.70	32.86	<u>35.00</u>	76.36	61.29	55.56	57.46

Table 4: **Results on mathematical reasoning.** SciGLM outperforms all base models across various parameter scales. The Avg. Math represents the weighted average score of all mathematical tasks within the evaluation category. Within each parameter setup, **Bold** spotlights the one with best performance, and Underline denotes the second-best result. Results marked as [†] are benchmarked by ourselves.

Model	MMLU	CEval	MBPP	Avg.
GPT-4	86.40	68.70	83.00	81.30
ChatGLM3-6B-Base	61.32	67.09	55.80	60.27
SciGLM (6B-Base)	61.38	67.16	45.00	57.56
ChatGLM3-32B-Base	69.05	79.94	58.20	79.37
SciGLM (32B-Base)	70.08	79.64	56.60	68.78

Table 5: **Results on general language understanding tasks.** Fine-tuning does not sacrifice most language tasks and only drops a bit on the code generation task.

and Formal Proof improve SciBench performance. This shows that our mixture dataset enables LLMs to acquire some general reasoning skills for solving scientific questions instead of merely overfitting to a certain task distribution, and achieving universal improvement on different downstream tasks.

Influence of Data Scaling. One central question for instruction dataset creation is how many samples are needed for a model to learn specific skills (Yuan et al., 2023). Prior works (Zhou et al., 2023) have shown that for dialogue tasks, as few as 1000 high-quality instruction samples can lead

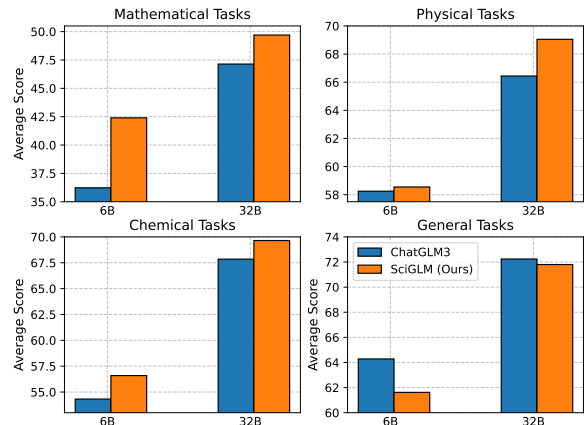


Figure 4: SciGLM consistently improving the ChatGLM3-Base for different scientific tasks, without sacrificing general language understanding capabilities

to significant improvements. We’re interested in analyzing data scaling law for scientific problem-solving. Through retraining the model with varying data proportions and analyzing the outcomes in science and math, as shown in Figure 6. One interesting pattern we find is initial data augmenta-

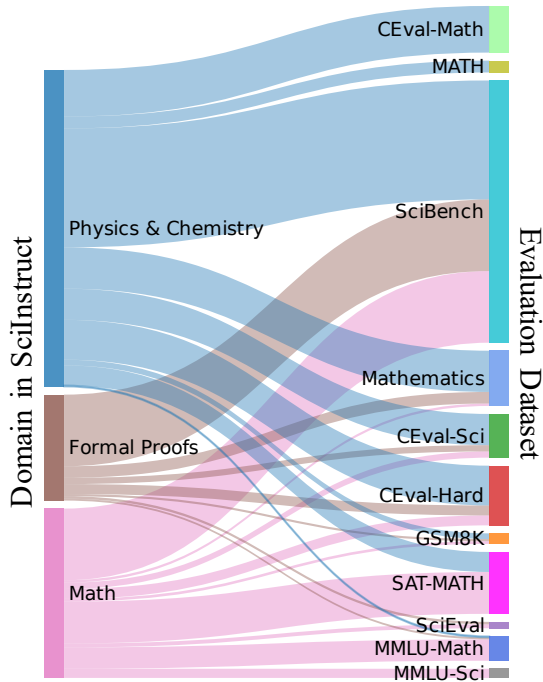


Figure 5: **Influence of different domains in fine-tuning data towards target tasks.** Weight is calculated by $(\text{Acc. (SciGLM)} - \text{Acc. (Exclude subjects)}) / \text{Acc. (SciGLM)}$ under the leave-one-out subject setting.

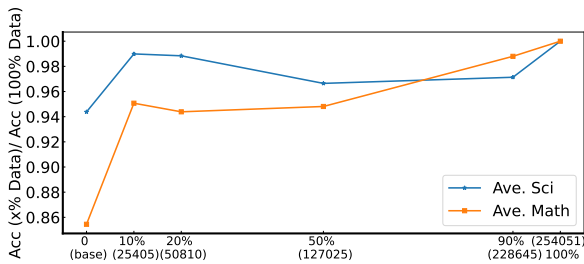


Figure 6: **Performance improvement over different scale of instruction data.** The x-axis represents the proportion of instruction per domain, and the y-axis represents the relative performance compared with SciGLM trained with the whole set.

tion of 10% yields improvements, but further additions show no significant gains until surpassing a 50% threshold. We hypothesize that the early gains are due to the model learning basic reasoning and task formatting, which requires fewer instruction data (less than 10k). Advancing to more complex skills, such as equation deduction, necessitates a larger dataset for effective learning and generalize. Future research on improving data quality could potentially lower the data requirement for LLM skill learning.

4 Related Works

Recently, there have been advances to bridge the gaps in reasoning difficulty and evaluation sub-

jects from three perspectives for scientific reasoning with LLMs.

High-level evaluation like SciBench (Wang et al., 2023c) and GPQA (Rein et al., 2023), which evaluates the scientific reasoning capabilities of LLMs at the college level and even graduate level. In addition, SciEval (Sun et al., 2023) provides a multi-level LLMs evaluation benchmark to address data leakage problems and subjective question/answer evaluation ability issues.

Continued pre-training like Galactica (Taylor et al., 2022) and MINERVA (Lewkowycz et al., 2022), which continue to train their respective base LLMs on multiple web texts including science-related or math-related corpus. This continued pre-training approach explores the potential of LLMs for science and contributes to the open-source models for the scientific community, but it is computationally expensive.

Dataset-specific fine-tuning like RFT (Yuan et al., 2023), WizardMath (Luo et al., 2023), MAMmoTH (Yue et al., 2023), and MetaMath (Yu et al., 2023), which constructs certain datasets including GSM8K and MATH, conducts supervised fine-tuning of LLMs and evaluates the popular benchmarks GSM8K and MATH. MAMmoTH not only improves the in-domain (IND) performance like GSM8K but also generalizes to broader mathematical tasks by building a spectrum of math instruction tuning datasets including in-domain and out-of-domain datasets. This series of methods mainly focuses on mathematical reasoning tasks.

5 Conclusion

In this work, we present a self-instructive annotation framework to create a high-level and high-quality dataset, SciInstruct, to enrich the scientific knowledge of LLMs. Using SciInstruct, we train SciGLM from ChatGLM series models, which significantly improve many scientific and mathematical benchmarks over the base models and outperform many state-of-the-art LLMs that have an order of magnitude more parameters. Our research underscores the significance of diverse training data as well as LLM self-annotation and correctness for enhancing general reasoning capability, even for hard domains like science. In the future, we aim to enhance the model’s capabilities by synthesizing data purely from unstructured data (like academic papers and knowledge base) and exploring more advanced self-improving techniques to bootstrap LLM reasoning.

6 Limitation

In this section, we discuss more limitations during the research of SciInstruct and SciGLM.

Scale of dataset and model. Even though our training dataset has expanded to approximately 254k, improving model performance still necessitates access to an even larger dataset. Presently, our model’s experimental outcomes are carried out at 6B and 32B parameters, leading to a relatively better performance. However, it’s important to note that these performances are constrained by the model’s scale. Moving forward, it’s worth exploring the potential benefits of leveraging larger-scale datasets and models to further improve performance.

Using data classifier to enhance the generation of models. In line with what was discussed in Section 2.3, we employ an instruction-quality classifier to boost the instruction quality, yielding improved performance as shown in Table 1. However, we anticipate that the data classifier, also referred to as the reward model, could provide even greater benefits. One particular avenue of improvement could be bootstrapping data to improve the ability of the base model.

References

Microsoft Research AI4Science and Microsoft Azure Quantum. 2023. The impact of large language models on scientific discovery: a preliminary study using gpt-4. *arXiv preprint arXiv:2311.07361*.

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2023. **Llemma: An open language model for mathematics**.

Bruno Barras, Samuel Boutin, Cristina Cornes, Judicaël Courant, Jean-Christophe Filliatre, Eduardo Gimenez, Hugo Herbelin, Gerard Huet, Cesar Munoz, Chetan Murthy, et al. 1997. *The Coq proof assistant reference manual: Version 6.1*. Ph.D. thesis, Inria.

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. 2023. **Accurate medium-range global weather forecasting with 3d neural networks**. *Nature*, 619:1–6.

Bo Chen, Xingyi Cheng, Yangli ao Geng, Shen Li, Xin Zeng, Boyan Wang, Jing Gong, Chiming Liu, Aohan Zeng, Yuxiao Dong, Jie Tang, and Le Song. 2023a. **xtrimopglm: Unified 100b-scale pre-trained transformer for deciphering the language of protein**. *bioRxiv*.

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

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. 2023b. Theoremqa: A theorem-driven question answering dataset. *arXiv preprint arXiv:2305.12524*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

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

Alex Davies, Petar Veličković, Lars Buesing, Sam Blackwell, Daniel Zheng, Nenad Tomašev, Richard Tanburn, Peter Battaglia, Charles Blundell, András Juhász, et al. 2021. Advancing mathematics by guiding human intuition with ai. *Nature*, 600(7887):70–74.

Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris Van Doorn, and Jakob von Raumer. 2015. The Lean theorem prover (system description).

Cheng Deng, Tianhang Zhang, Zhongmou He, Qiyuan Chen, Yuanyuan Shi, Le Zhou, Luoyi Fu, Weinan Zhang, Xinbing Wang, Chenghu Zhou, et al. 2023. Learning a foundation language model for geoscience knowledge understanding and utilization. *arXiv preprint arXiv:2306.05064*.

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2022. Compositional semantic parsing with large language models. *arXiv preprint arXiv:2209.15003*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335.

617	Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2023. Critic: Large language models can self-correct with tool-interactive critiquing. <i>arXiv preprint arXiv:2305.11738</i> .	673
618		674
619		675
620		676
621		677
622	Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. <i>arXiv preprint arXiv:2009.03300</i> .	678
623		
624		
625		
626	Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. <i>arXiv preprint arXiv:2103.03874</i> .	
627		
628		
629		
630		
631	Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In <i>Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)</i> , pages 523–533.	
632		
633		
634		
635		
636		
637	Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, et al. 2023. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. <i>arXiv preprint arXiv:2305.08322</i> .	
638		
639		
640		
641		
642		
643	Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. <i>Advances in neural information processing systems</i> , 35:22199–22213.	
644		
645		
646		
647		
648	Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. Parsing algebraic word problems into equations. <i>Transactions of the Association for Computational Linguistics</i> , 3:585–597.	
649		
650		
651		
652		
653	Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamile Lukosiute, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, Saurav Kadavath, Shannon Yang, Thomas Henighan, Timothy Maxwell, Timothy Telleen-Lawton, Tristan Hume, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. 2023. <i>Measuring faithfulness in chain-of-thought reasoning</i> . <i>CoRR</i> , abs/2307.13702.	
654		
655		
656		
657		
658		
659		
660		
661		
662		
663		
664		
665	Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. 2023. Platypus: Quick, cheap, and powerful refinement of llms. <i>arXiv preprint arXiv:2308.07317</i> .	
666		
667		
668	Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. <i>Advances in Neural Information Processing Systems</i> , 35:3843–3857.	673
669		674
670		675
671		676
672		677
	Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Heng Huang, Jiuxiang Gu, and Tianyi Zhou. 2023a. Reflection-tuning: Data recycling improves llm instruction-tuning. <i>arXiv preprint arXiv:2310.11716</i> .	678
	Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023b. Making language models better reasoners with step-aware verifier. In <i>Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)</i> , pages 5315–5333.	679
		680
		681
		682
		683
		684
	Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. <i>arXiv preprint arXiv:1705.04146</i> .	685
		686
		687
		688
	Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. 2023. The flan collection: Designing data and methods for effective instruction tuning. <i>arXiv preprint arXiv:2301.13688</i> .	689
		690
		691
		692
		693
	Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. <i>arXiv preprint arXiv:2308.09583</i> .	694
		695
		696
		697
		698
		699
	Tobias Nipkow, Markus Wenzel, and Lawrence C Paulson. 2002. <i>Isabelle/HOL: a proof assistant for higher-order logic</i> .	700
		701
		702
	Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. <i>arXiv preprint arXiv:2112.00114</i> .	703
		704
		705
		706
		707
		708
	OpenAI. 2023. <i>Gpt-4 technical report</i> .	709
	Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? <i>arXiv preprint arXiv:2103.07191</i> .	710
		711
		712
		713
	Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. <i>arXiv preprint arXiv:2304.03277</i> .	714
		715
		716
	Samyram Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In <i>SC20: International Conference for High Performance Computing, Networking, Storage and Analysis</i> , pages 1–16. IEEE.	717
		718
		719
		720
		721
		722
	David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. <i>arXiv preprint arXiv:2311.12022</i> .	723
		724
		725
		726
		727

728	Subhro Roy and Dan Roth. 2016. Solving general arithmetic word problems. <i>arXiv preprint arXiv:1608.01413</i> .	2023c. Scibench: Evaluating college-level scientific problem-solving abilities of large language models. <i>arXiv preprint arXiv:2307.10635</i> .	783
729			784
730			785
731	Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. <i>arXiv preprint arXiv:2110.08207</i> .	Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022c. Self-consistency improves chain of thought reasoning in language models. <i>arXiv preprint arXiv:2203.11171</i> .	786
732			787
733			788
734			789
735			790
736			
737	Noah Shinn, Beck Labash, and Ashwin Gopinath. 2023. Reflexion: an autonomous agent with dynamic memory and self-reflection. <i>arXiv preprint arXiv:2303.11366</i> .	Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. 2023d. How far can camels go? exploring the state of instruction tuning on open resources. <i>arXiv preprint arXiv:2306.04751</i> .	791
738			792
739			793
740			794
741	Liangtai Sun, Yang Han, Zihan Zhao, Da Ma, Zhennan Shen, Baocai Chen, Lu Chen, and Kai Yu. 2023. Scieval: A multi-level large language model evaluation benchmark for scientific research. <i>arXiv preprint arXiv:2308.13149</i> .	Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hananeh Hajishirzi. 2022d. Self-instruct: Aligning language model with self generated instructions. <i>arXiv preprint arXiv:2212.10560</i> .	797
742			798
743			799
744			800
745			801
746	Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. <i>arXiv preprint arXiv:2210.09261</i> .	Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. <i>arXiv preprint arXiv:2109.01652</i> .	802
747			803
748			804
749			805
750			806
751			
752	Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. <i>arXiv preprint arXiv:2211.09085</i> .	Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022a. Chain-of-thought prompting elicits reasoning in large language models. In <i>Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022</i> .	807
753			808
754			809
755			810
756			811
757	Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. <i>arXiv preprint arXiv:2302.13971</i> .	Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. <i>Advances in Neural Information Processing Systems</i> , 35:24824–24837.	812
758			813
759			814
760			
761			815
762			816
763	Boshi Wang, Xiang Deng, and Huan Sun. 2022a. Iteratively prompt pre-trained language models for chain of thought. <i>arXiv preprint arXiv:2203.08383</i> .	Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. <i>arXiv preprint arXiv:2111.02080</i> .	817
764			818
765			819
766	Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2022b. Towards understanding chain-of-thought prompting: An empirical study of what matters. <i>arXiv preprint arXiv:2212.10001</i> .	Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. <i>arXiv preprint arXiv:2304.12244</i> .	820
767			821
768			822
769			823
770			
771	Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. <i>arXiv preprint arXiv:2305.04091</i> .	Kaiyu Yang, Aidan Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. 2023. LeanDojo: Theorem proving with retrieval-augmented language models. In <i>Neural Information Processing Systems (NeurIPS)</i> .	824
772			825
773			826
774			827
775			828
776	Peiyi Wang, Lei Li, Liang Chen, Feifan Song, Binghui Lin, Yunbo Cao, Tianyu Liu, and Zhifang Sui. 2023b. Making large language models better reasoners with alignment. <i>arXiv preprint arXiv:2309.02144</i> .	Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models.	829
777			830
778			831
779			832
780			833
781	Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang.		834
782			835
			836
			837
			838

839 Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak
840 Shafran, Karthik Narasimhan, and Yuan Cao. 2022.
841 React: Synergizing reasoning and acting in language
842 models. *arXiv preprint arXiv:2210.03629*.

843 Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu,
844 Zhengying Liu, Yu Zhang, James T Kwok, Zhen-
845 guo Li, Adrian Weller, and Weiyang Liu. 2023.
846 Metamath: Bootstrap your own mathematical ques-
847 tions for large language models. *arXiv preprint*
848 *arXiv:2309.12284*.

849 Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting
850 Dong, Chuanqi Tan, and Chang Zhou. 2023. Scal-
851 ing relationship on learning mathematical reason-
852 ing with large language models. *arXiv preprint*
853 *arXiv:2308.01825*.

854 Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wen-
855 hao Huang, Huan Sun, Yu Su, and Wenhua Chen.
856 2023. Mammoth: Building math generalist models
857 through hybrid instruction tuning. *arXiv preprint*
858 *arXiv:2309.05653*.

859 Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao
860 Liu, Yuxiao Dong, and Jie Tang. 2023. Agenttuning:
861 Enabling generalized agent abilities for llms. *arXiv*
862 *preprint arXiv:2310.12823*.

863 Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang,
864 Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu,
865 Wendi Zheng, Xiao Xia, et al. 2022. Glm-130b:
866 An open bilingual pre-trained model. *arXiv preprint*
867 *arXiv:2210.02414*.

868 Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang,
869 Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen,
870 and Nan Duan. 2023. Agieval: A human-centric
871 benchmark for evaluating foundation models. *arXiv*
872 *preprint arXiv:2304.06364*.

873 Chunting Zhou, Pengfei Liu, Puxin Xu, Srinu Iyer, Jiao
874 Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu,
875 Lili Yu, et al. 2023. Lima: Less is more for alignment.
876 *arXiv preprint arXiv:2305.11206*.

877 Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei,
878 Nathan Scales, Xuezhi Wang, Dale Schuurmans,
879 Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022.
880 Least-to-most prompting enables complex reason-
881 ing in large language models. *arXiv preprint*
882 *arXiv:2205.10625*.

7 Appendix 883

7.1 Instruction Example 884

Instruction Format for Formal Math. 885

This is an example of an instruction format for Lean.

Problem: theorem gcd_self (n : Nat) : gcd n
n = n

Answer: cases n <=> simp [gcd, mod_self]

Instruction Format for Physics, Chemistry, and 886

Math. To make our dataset consistent, we define 887
a unified template for the problem-solving process 888
that matches the logical flow of human thinking for 889
Physics, Chemistry, and Math. For each question, 890
we first list the basic analysis of which scientific 891
concepts (or knowledge points) this question asks, 892
and then present step-by-step reasoning steps of 893
the detailed solution, and eventually summarize the 894
answer. 895

This is an example of an instruction format.

Problem: * * *.

Answer:

Analysis: This question examines * * * knowl-
edge points.

Step 1: * * *.

Step i: * * *.

Step n: * * *.

To sum up, the answer to this question is * * *.

7.2 Dataset of Data Classifier 896

Source	# Positive Samples	# Negative Samples	Total
Original Clean Data	11,553	0	11,553
ChatGLM2-6B	0	23,106	23,106
GPT-3.5-Turbo-0613	745	2075	2,820
GPT-4-0613	332	3,898	4,230
Total	12,630	29,079	41,709

Table 6: Composing data based on a limited set of la-
beled examples to train a data classifier and filter out
noisy instructions.

7.3 Evaluation Tasks 897

- CEval-Hard (Huang et al., 2023) comprises the 898
eight subjects at the hard level from CEval. 899
- CEval-Sci (Huang et al., 2023) is selected from 900
CEval including physical, chemical, and mathe- 901
matical, subjects. 902
- MMLU-Sci (Hendrycks et al., 2020) is similar to 903
CEval-Sci and includes physical, chemical, and 904

Eval Dataset	# Samples	Answer Form	Subject
CEval-Hard	152	Multi-choice	$\triangle, \blacktriangle, \diamond, \blacklozenge, \blackstar$
CEval-Sci	210	Multi-choice	$\triangle, \diamond, \blacktriangle, \blackstar, \clubsuit$
MMLU-Sci	1,855	Multi-choice	$\S, \triangle, \diamond, \blacktriangle, \blackstar, \clubsuit$
SciEval	15,901	Multi-choice	$\blacklozenge, \blackstar, \clubsuit$
MATH	5,000	Open-formed	$\dagger, \ddagger, \S, \triangle, \diamond, \blacktriangle, \blacktriangledown$
Mathematics	1,000	Open-formed	$\dagger, \ddagger, \diamond, \blacktriangle$
SAT-Math	220	Multi-choice	$\ddagger, \triangle, \blacktriangledown$
MMLU-Math	974	Multi-choice	$\S, \triangle, \diamond, \blacktriangle$
CEval-Math	90	Multi-choice	$\triangle, \triangle, \diamond, \blacktriangle$

Table 7: Overview of Scientific Evaluation Tasks. (\dagger : Pre-Algebra, \ddagger : Inter-Algebra, \S : Algebra, \triangle : Probability, \diamond : NumTheory, \blacktriangle : Statistics, \blacktriangle : Calculus, \blacktriangledown : Geometry, \blacklozenge : Physics, \blackstar : Chemistry, \clubsuit : Biology)

mathematical subjects selected from MMLU.

- SciEval (Sun et al., 2023) serves as a comprehensive, multi-disciplinary benchmark for assessing subjective question-answering capabilities.
- MATH (Hendrycks et al., 2021) is a standard benchmark for evaluating the mathematical ability of LLMs.
- Mathematics (Davies et al., 2021) mainly includes Algebra, NumTheory, and Calculus.
- SAT-Math (Zhong et al., 2023) is a math task including Intermediate Algebra, Probability, and Calculus.
- MMLU-Math (Hendrycks et al., 2020) evaluation is the same as MAMmoTH (Yue et al., 2023), including Algebra, Probability, NumTheory, and Calculus.
- CEval-MATH (Huang et al., 2023) only includes mathematical subjects extracted from CEval.

7.4 Evaluation Baselines

- GPT-4 (OpenAI, 2023): We consider the closed-source model GPT-4, which uses CoT prompting.
- GLM (Du et al., 2022; Zeng et al., 2022) and LLaMA (Touvron et al., 2023) Base: We select ChatGLM2-6B (Base), ChatGLM3-6B (Base), and ChatGLM3-32B (Base) as the base models. In addition, we consider LLaMA-2-7B and LLaMA-2-13B as baselines.
- Continue Pre-training: We include Galactica (Taylor et al., 2022) although we note that its continued pre-training does not necessarily

include college-level scientific reasoning.

- Dataset-specific Tuning: For the dataset-specific tuning, we consider WizardMath (Luo et al., 2023), MAMmoTH-CoT (Yue et al., 2023), and MetaMath (Yu et al., 2023), which adapt to the MATH dataset.

7.5 Scientific Reasoning Datasets

Mathematical Datasets. In the past, the emphasis in mathematical reasoning tasks was primarily on fundamental arithmetic reasoning datasets (Koncel-Kedziorski et al., 2015; Roy and Roth, 2016; Patel et al., 2021), addressing basic mathematical problems such as AddSub (Hosseini et al., 2014). Later on, to address realistic math word problems, some more difficult datasets are proposed (Ling et al., 2017; Amini et al., 2019; Cobbe et al., 2021; Azerbayev et al., 2023). To construct a more difficult and diversified dataset, NumGLUE (Mishra et al., 2022) and LiLA (Mishra et al., 2022) have been introduced to enhance current research. However, their focus remains primarily on elementary, middle school, and high school math problems. Given the rapid advancement of LLMs, to assess their capacity and constraints in addressing more intricate math problems, MMLU (Hendrycks et al., 2020) with college-level math problems has been proposed. To address the more challenging college-level math problems, PoT (Chen et al., 2023b; Wang et al., 2023c) is proposed. Our proposed SciInstruct covers more complex and more diversified math problems at the college level.

Scientific Datasets. To address the scientific ability of LLMs, SciBench (Wang et al., 2023c) proposes various solutions like calling existing tools (python and Wolfram). SciEval (Sun et al., 2023) presents a multi-level evaluation benchmark for scientific research. GPQA (Rein et al., 2023) builds a graduate-level Q&A benchmark to evaluate more difficult physics and chemistry questions.

7.6 General Reasoning with LLMs

Assisted by Chain-of-Thought prompting (Wei et al., 2022b; Kojima et al., 2022; Wang et al., 2022c) and Tree-or-Though (Yao et al., 2023), LLMs have brought decent reasoning performance improvements. Especially, on the challenging BIG-Bench tasks, the CoT technique has already outperformed human ability (Suzgun et al., 2022). Later on, to address reasoning tasks, a few studies (Nye et al., 2021; Wang et al., 2022a,b; Li et al., 2023b;

Wang et al., 2023b; Yu et al., 2023; Drozdov et al., 2022; Zhou et al., 2022) propose various methods to leverage LLMs and retain step-by-step processes. With the rapid development of LLM Agents, several works (Yao et al., 2022; Zeng et al., 2023) propose to utilize external tools to enhance the reasoning capabilities of LLMs. For example, ReAct (Yao et al., 2022) attempts to call existing tools like web search engines to improve the reasoning skills of LLMs. Indeed, leveraging the programs as thought processes is a natural way. Several recent studies have utilized programs as thought processes, such as PoT (Chen et al., 2022), to improve the reasoning abilities of LLMs. To enhance LLMs’ reasoning performance in solving mathematical or scientific problems with PoT, several methods have been suggested, including Self-critic (Gou et al., 2023), Self-eval (Xie et al., 2021), and Plan-and-solve (Wang et al., 2023a). For example, self-critic (Gou et al., 2023) and self-eval (Xie et al., 2021) propose to adopt self-evaluation to improve the generated program’s robustness. Different from these two methods, plan-and-solve (Wang et al., 2023a) adopts more detailed planning instructions to generate a high-level reasoning plan for LLMs. Indeed, these methods have demonstrated that they can obtain great capabilities over PoT.

7.7 Instruction Tuning in LLMs

To align language models with human preferences and effective objectives, some works (Zeng et al., 2023; Xu et al., 2023) design instruction tuning. The instruction tuning aims to mine the potential capabilities by aligning with and responding to human preference. Early on, instruction tuning focuses on improving the instruction-following abilities of LLMs for general purposes. Represented by FLAN (Wei et al., 2021) and T0 (Sanh et al., 2021), they aim to understand the generalization abilities of LLMs for instruction tuning. Later, to comprehend the efficacy and performance of enlarging the instructional tuning datasets on models, FLAN-v2 (Chung et al., 2022; Longpre et al., 2023) has been proposed to validate this goal. However, these methods build training instruction tuning datasets in a human-annotated manner. Recently, various studies (Wang et al., 2022d; Xu et al., 2023; Peng et al., 2023; Zeng et al., 2023; Zhou et al., 2023; Wang et al., 2023d) have started to construct synthetic instruction following datasets distilled from some LLMs like GPT-3.5-turbo and GPT-4. Like

these works, Platypus (Lee et al., 2023) constructs a small-scale instruction for the following dataset by utilizing a domain-specialized dataset, aiming to enhance the reasoning capabilities of LLMs.

7.8 GPT Labeling

As shown in Figure 7, we present the process of GPT-4 labeling.

7.9 Detailed Error Analysis

As shown in Figure 8, we give detailed error examples, which are summarized in Section 2.3.

7.10 Detailed Experimental Results

Domain-wise Results in Figure 4. We present the detailed subject results in Table 8 and Table 9 for Figure 4.

7.11 SciInstruct Examples

We present SciInstruct examples for Math(Figure 10), Physics (Figure 11), Chemistry (Figure 12), and Formal Proofs (Figure 13).

7.12 SciInstruct Examples

As shown in Figure 9, we present a concrete example and model output from SciEval, including the ChaGLM3-6B-Base and SciGLM.

GPT-4 Labeling	
Prompt: The following input includes three segments. It starts with a question, followed by a step-by-step solution to the question, and ends with the true answer. If the answer obtained in the solution is equivalent to the real one, output '1', otherwise output '0'.	
Input	
Q: A photon moving at speed c , a person chasing light at $0.9c$, that person observes the speed of light is?	
Solution: According to Einstein’s theory of special relativity, we cannot simply add up the velocities. In relativity, the superposition of velocities follows the Lorentz transformation, rather than the classical addition. For the speed of light, there is a special case in which the observer measures the speed of light as c , regardless of the observer’s speed. This means that even if an observer is moving at $0.9c$, the speed of light he measures is still c . Therefore, the speed of light observed by this person is c .	
Real Answer: c .	
GPT-4 Output: 1	

Figure 7: We present the process of GPT-4 labeling. Given a prompt, question, solution, and real answer, GPT-4 can give the final label output.

model	Ave.	CEval				MMLU				SciEval
		Avg.	College Physics	High School Physics	Middle School Physics	Avg.	College Physics	Conceptual Physics	Middle School Physics	Physics.
ChatGLM3-6B-Base	54.32	63.16	42.11	63.16	84.21	45.59	39.22	55.32	34.44	46.95
SciGLM (ChatGLM3-6B-Base)	56.59	66.67	47.37	63.16	89.47	46.52	34.31	55.32	41.06	47.56
ChatGLM3-32B-Base	67.85	78.95	57.89	78.95	100.00	56.76	42.16	62.55	57.62	51.22
SciGLM (ChatGLM3-32B-Base)	69.65	80.70	63.16	78.95	100.00	58.61	41.18	66.81	57.62	56.71

Table 8: Detailed results of physical subjects for Figure 1. **Bold** denotes the best result in the closed-source models and open-source models.

model	Ave.	CEval				MMLU			SciEval
		Avg.	College Chemistry	High School Chemistry	Middle School Chemistry	Avg.	College Chemistry	High School Chemistry	Chemistry.
ChatGLM3-6B-Base	58.25	66.67	45.83	63.16	95.00	49.83	45.00	52.22	55.76
SciGLM (ChatGLM3-6B-Base)	58.55	68.25	41.67	73.68	95.00	48.84	47.00	49.75	57.27
ChatGLM3-32B-Base	66.44	69.84	41.67	78.95	95.00	63.04	51.00	68.97	60.15
SciGLM (ChatGLM3-32B-Base)	69.05	71.42	45.83	78.95	95.00	66.67	57.00	71.43	61.40

Table 9: Detailed results of chemical subjects for Figure 1. **Bold** denotes the best result in the closed-source models and open-source models.

Comprehension mistakes**Explanation**

Answers that do not resolve the task exactly as instructed, often including an output that is insufficient.

Example

Q: A particle moving in a straight line in the x direction has the coordinates $x = 5t^2 - t^3$ at time t , where x is measured in meters and t in seconds. Find: displacement and average velocity in the 4th second.

A: The displacement in 4 seconds can be obtained by calculating the coordinates at $t = 4$ minus the coordinates at $t = 0$, Displacement = $x(4) - x(0) = 5 \times 4^2 - 4^3 - 5 \times 0^2 + 0^3 = 5 \times 16 - 64 - 0 + 0 = 80 - 64 = 16$. Average velocity in 4 seconds can be obtained by calculating displacement divided by time, that is $average_velocity = displacement/time = 16/4 = 4$. In summary, the displacement in 4 seconds is 16 meters, the average speed is 4 meters/second, and the distance traveled by the particle is 16 meters.

Error: Wrong comprehension. The answer should be displacement and average velocity 'in the 4th second' rather than 'in 4 seconds'.

Calculation mistakes**Explanation**

Answers that fail to accurately calculate or simplify mathematical expressions, especially when encountering integration and differentiation.

Example

Q: In a shell with an inner radius of R_1 and an outer radius of R_2 , the charge density in the shell is $\rho = a + br$, Where r is the distance from the center to the observation point, and there is no charge distribution elsewhere in the space. (a) Find the expression that the electric field everywhere in the space depends on r .

A: First of all, according to Gauss's law, the total electric flux of any closed surface inside the closed surface is equal to the total charge inside the closed surface. According to the conditions in the topic, the charge density in the shell is $\rho = a + br$, the total charge in the shell can be obtained as: $Q = \int_{R_1}^{R_2} \int_0^\pi \int_0^{2\pi} (a + br) \cdot r^2 \sin\theta d\phi d\theta dr = 4\pi \int_{R_1}^{R_2} (a + br)r^2 dr = 4\pi \left(\frac{a}{4}(R_2^4 - R_1^4) + \frac{b}{5}(R_2^5 - R_1^5) \right)$. According to the Gauss theorem, inside the spherical shell, the flux of the electric field is zero. Therefore, we only need to produce the electric field outside the shell. Outside the shell, according to Coulomb's Law, the electric field generated by the charge can be expressed as: $E = \frac{1}{4\pi\epsilon_0} \frac{Q}{r^2}$ Substituting Q into this formula yields: $E = \frac{1}{4\pi\epsilon_0} \frac{4\pi \left(\frac{a}{4}(R_2^4 - R_1^4) + \frac{b}{5}(R_2^5 - R_1^5) \right)}{r^2}$ simplified, we can get electric field depends on the expression of r as follows: $E = \frac{1}{\epsilon_0} \left(\frac{a}{r^2}(R_2^4 - R_1^4) + \frac{b}{r^2}(R_2^5 - R_1^5) \right)$

Error: Wrong integration.

False reasoning**Explanation**

Solutions consist of false reasoning steps or made-up information in reasoning.

Example

Q: The escape work of polonium and barium is 4.52 eV, and 2.50 eV, respectively. Calculate the cut-off frequencies of barium and polonium.

A: According to the theory of photoelectric effect, the relationship between escape work and cutting potential can be expressed by the following formula escape work = cutting potential - photon energy. Apply the relationship between escape work and cutting potential to the case of barium and polonium. For polonium, cutting potential = escape work + photon energy = 4.52 eV + $h\nu$. For barium, cutting potential = escape work + photon energy = 2.50 eV + $h\nu$, where h is Planck's constant and ν is the frequency of light.

In the visible range, the frequency range of light is approximately 4.3×10^{14} Hz to 7.5×10^{14} Hz. Now we can calculate the cut-off frequencies of barium and polonium respectively. For polonium, $4.52 \text{ eV} + h\nu_{\text{cut}} = 4.30 \text{ eV}$. $h\nu_{\text{cut}} = 4.30 \text{ eV} - 4.52 \text{ eV}$. $h\nu_{\text{cut}} = -0.22 \text{ eV}$. For barium, $2.50 \text{ eV} + h\nu_{\text{cut}} = 4.30 \text{ eV}$. $h\nu_{\text{cut}} = 4.30 \text{ eV} - 2.50 \text{ eV}$. $h\nu_{\text{cut}} = 1.80 \text{ eV}$.

In summary, the cut-off frequency of polonium is -0.22 eV , and that of barium is 1.80 eV . Because the cutoff frequency is the frequency of light and must be positive, barium can be used as a phototube cathode material in the visible range. Final answer: Barium can be used as a phototube cathode material in the visible range.

Error: False reasoning, cutting potential should be W/h .

Figure 8: Error analysis of three types of self-reflection for noisy instruction filtering.

<p>Problem How does electric field affect capacitance?</p> <p>A. Electric field decreases capacitance. B. Electric field doesn't affect capacitance. C. Electric field changes capacitance based on geometry. D. Electric field increases capacitance.</p>
<p>Correct Answer B</p>
<p>Answer produced by ChatGLM3-6B-Base ✗ D</p>
<p>Answer produced by SciGLM (ChatGLM3-6B-Base) ✔ B</p>

Figure 9: An example problem from SciEval is accurately solved by SciGLM (ChaGLM3-6B-Base). For this problem, the category is physics; the topic is Electric Forces and Fields; the ability is Scientific Calculation; the type is multiple-choice; the task name is SocraticQA.

<p>Problem Using the method of undetermined coefficients to answer the below question. $y'' - 2y' + 5y = \sin x, y(0) = 1, y'(0) = 1$</p>
<p>Answer Analysis: The key points of this problem are the undetermined coefficients and the differential equation of parameter variations. This test uses the Method of undetermined coefficients to solve differential equation problems. Step1: the auxiliary equation is $r^2 - 2r + 5 = 0$ with roots $r = 1 \pm 2i$, so the complementary solution is: $y_c(x) = e^x (c_1 \cos 2x + c_2 \sin 2x)$. Try the particular solution: $y_p(x) = A \cos x + B \sin x$, $y_p' = -A \sin x + B \cos x$ Step2: substituting, we have: $(-A \cos x - B \sin x) - 2(-A \sin x + B \cos x) + 5(A \cos x + B \sin x) = \sin x$ $\Rightarrow (4A - 2B) \cos x + (2A + 4B) \sin x = \sin x$. Then: $4A - 2B = 0, 2A + 4B = 1 \Rightarrow A = \frac{1}{10}, B = \frac{1}{3}$ and the general solution is: $y(x) = y_1(x) + y_p(x) = e^x (c_1 \cos 2x + c_2 \sin 2x) + \frac{1}{10} \cos x + \frac{1}{3} \sin x$ But, $1 = y(0) = c_1 + \frac{1}{10} \Rightarrow c_1 = \frac{9}{10}$ and $1 = y'(0) = 2c_2 + c_1 + \frac{1}{5} \Rightarrow c_2 = -\frac{1}{26}$ Step3: Thus the solution to the initial-value problem is: $y(x) = e^x \left(\frac{9}{10} \cos 2x - \frac{1}{26} \sin 2x \right) + \frac{1}{10} \cos x + \frac{1}{3} \sin x$. To sum up, the answer is: $y(x) = e^x \left(\frac{9}{10} \cos 2x - \frac{1}{26} \sin 2x \right) + \frac{1}{10} \cos x + \frac{1}{3} \sin x$.</p>

Figure 10: An instruction example on the math subject.

Problem

Two lightbulbs, one rated 30 W at 120 V and another rated 40 W at 120 V, are arranged in two different circuits.

- a. The two bulbs are first connected in parallel to a 120 V source.
 i. Determine the resistance of the bulb rated 30 W and the current in it when it is connected in this circuit.
 ii. Determine the resistance of the bulb rated 40 W and the current in it when it is connected to this circuit.

- b. The bulbs are now connected in series with each other and have a 120 V source.
 i. Determine the resistance of the bulb rated 30 W and the current in it when it is connected in this circuit.
 ii. Determine the resistance of the bulb rated 40 W and the current in it when it is connected to this circuit.

c. In the spaces below, number the bulbs in each situation described, in order of their brightness.

(1= brightest, 4 = dimmest)

- ___ 30 W bulb in the parallel circuit
 ___ 40 W bulb in the parallel circuit
 ___ 30 W bulb in the series circuit
 ___ 40 W bulb in the series circuit

d. Calculate the total power dissipated by the two bulbs in each of the following cases.

- i. The parallel circuit
 ii. The series circuit

Answer

Analysis: Use formulas to calculate resistance, current, and total power.

(a) i: Calculate the resistance and current of the light bulb.

$$P = V^2/R \text{ given } R=480 \Omega \text{ and}$$

$$V=IR \text{ given } I=0.25 \text{ A}$$

ii: Calculate the resistance and current of the light bulb.

$$P = V^2/R \text{ given } R=360 \Omega \text{ and}$$

$$V=IR \text{ given } I=0.33 \text{ A}$$

(b) i./ii. Calculate the resistance and current of the light bulb.

The resistances are unchanged 480 Ω and 360 Ω . The total resistance in series is 480 Ω + 360 Ω = 840 Ω making the total current $I = V/R = 0.14 \text{ A}$ which is the same value for both resistors in series.

(c) Compare the brightness of the light bulb.

The bulbs are brightest in parallel, where they provide their labeled values of 40 W and 30 W. In series, it is the larger resistor (the 30 W bulb) that glows brighter with a larger potential difference across it in series. This gives the order from top to bottom as 2 1 3 4.

(d) i: Calculate the total power consumed by two light bulbs.

In parallel, they each operate at their rated voltage so they each provide their rated power and $P_T = 30\text{W} + 40\text{W} = 70\text{W}$

ii: Calculate the total power consumed by two light bulbs

$$\text{In series } P_T = V_T^2/R_T = 17\text{W}$$

In summary,

(a) i: $P = 480 \Omega$ and $V = 0.25\text{A}$

ii: $P = 360 \Omega$ and $V = 0.33\text{A}$

(b) i/ii: $P = 840 \Omega$ and $V = 0.14\text{A}$

(c) This gives the order from top to bottom as 2 1 3 4.

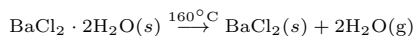
(d) i: $P_T = 70\text{W}$

ii: $P_T = 17\text{W}$

Figure 11: An instruction example on the physics subject.

Problem

Consider a mixture of the two solids, $\text{BaCl}_2 + 2\text{H}_2\text{O}$ (FM 244.26) and KCl (FM 74.551), in an unknown ratio. (The notation $\text{BaCl}_2 \cdot 2\text{H}_2\text{O}$ means that a crystal is formed with two water molecules for each BaCl_2 .) When the unknown is heated to 160°C for 1 h, the water of crystallization is driven off:



A sample originally weighing 1.7839 g weighed 1.5623 g after heating. Calculate the weight percent of Ba, K, and Cl in the original sample.

Answer

Analysis: The content of this question is to calculate the weight percentage.

Step1: Formula and atomic masses: Ba(137.327), Cl(35.453), K(39.098), H_2O (18.015), KCl (74.551), $\text{BaCl}_2 \cdot 2\text{H}_2\text{O}$ (244.26), H_2O

lost = 1.7839 – 1.5623 = 0.2216 g = 1.2301×10^{-2} mol of H_2O . For 2 mol H_2O lost, 1 mol $\text{BaCl}_2 \cdot 2\text{H}_2\text{O}$ must have been present.

$\frac{1}{2}$ (1.2301×10^{-2} mol H_2O lost) = 6.1504×10^{-3} mol $\text{BaCl}_2 \cdot 2\text{H}_2\text{O}$ = 1.5024 g. The Ba and Cl contents of the $\text{BaCl}_2 \cdot 2\text{H}_2\text{O}$ are

$$\text{Ba} = \left(\frac{137.33}{244.26} \right) (1.5024 \text{ g}) = 0.84469 \text{ g}$$

$$\text{Cl} = \left(\frac{2(35.453)}{244.26} \right) (1.5024 \text{ g}) = 0.43613 \text{ g}$$

Step2: Because the total sample weighs 1.783 g and contains 1.5024 g of $\text{BaCl}_2 \cdot 2\text{H}_2\text{O}$, the sample must contain 1.7839 – 1.5024 = 0.2815 g of KCl , which contains

$$\text{K} = \left(\frac{39.098}{74.551} \right) (0.2815) = 0.14763 \text{ g}$$

$$\text{Cl} = \left(\frac{35.453}{74.551} \right) (0.2815) = 0.13387 \text{ g}$$

Weight percent of each element:

$$\text{Ba} = \frac{0.84469}{1.7839} = 47.35\%$$

$$\text{K} = \frac{0.14763}{1.7839} = 8.28\%$$

$$\text{Cl} = \frac{0.43613+0.13387}{1.7839} = 31.95\%$$

In summary, the weight percent of Ba is 47.35%, the weight percent of K is 8.28%, the weight percent of Cl is 31.95%.

Figure 12: An instruction example on the chemistry subject.

Problem lemma unfold_sub α [subtraction_monoid α] (a b c : α) (h : a + -b = c) : a - b = c :=
Answer by rw [sub_eq_add_neg, h]
Problem lemma subtype.exists_pi_extension ι : Sort* α : $\iota \rightarrow$ Sort* [ne : Π i, nonempty (α i)] p : $\iota \rightarrow$ Prop (f : Π i : subtype p, α i) : g : Π i : ι , α i, (λ i : subtype p, g i) = f :=
Answer begin tactic.classical, refine (λ i, if hi : p i then f (i, hi) else classical.choice (ne i), funext _), rintro (i, hi), exact dif_pos hi end

Figure 13: Examples on formal proofs.