DS^2-INSTRUCT: Domain-Specific Data Synthesis for Large Language Models Instruction Tuning

Ruiyao Xu[1]* **Noelle I. Samia**[1]† **Han Liu**[1]†

[1]Northwestern University

ruiyaoxu2028@u.northwestern.edu, {n-samia, hanliu}@northwestern.edu

Abstract

Adapting Large Language Models (LLMs) to specialized domains requires high-quality instruction tuning datasets, which are expensive to create through human annotation. Existing data synthesis methods focus on general-purpose tasks and fail to capture domain-specific terminology and reasoning patterns. To address this, we introduce DS^2-INSTRUCT, a zero-shot framework that generates domain-specific instruction datasets without human supervision. Our approach first generates task-informed keywords to ensure comprehensive domain coverage. It then creates diverse instructions by pairing these keywords with different cognitive levels from Bloom's Taxonomy. Finally, it uses self-consistency validation to ensure data quality. We apply this framework to generate datasets across seven challenging domains, such as mathematics, finance, and logical reasoning. Comprehensive evaluation demonstrates that models fine-tuned on our generated data achieve substantial improvements over existing data generation methods. [1]

1 Introduction

Instruction-tuning Large Language Models (LLMs) requires high-quality data (Wang et al., 2023c; Ouyang et al., 2022; Köpf et al., 2023; Yin et al., 2023; Conover et al., 2023). Since acquiring large-scale datasets for fine-tuning LLMs can be prohibitively expensive due to human annotation costs, many methods use LLMs to synthesize data instead (Wang et al., 2023c; Yuan et al., 2024; Xu et al., 2024; Honovich et al., 2023; Xu et al., 2023b; Mehrabi et al., 2023; Ding et al., 2023). However, these data synthesis methods typically target general-purpose models, overlooking the need for domain-specific tuning which is critical for many real-world applications. For example, a legal chatbot must master law-related terminology, reasoning patterns, and procedures.

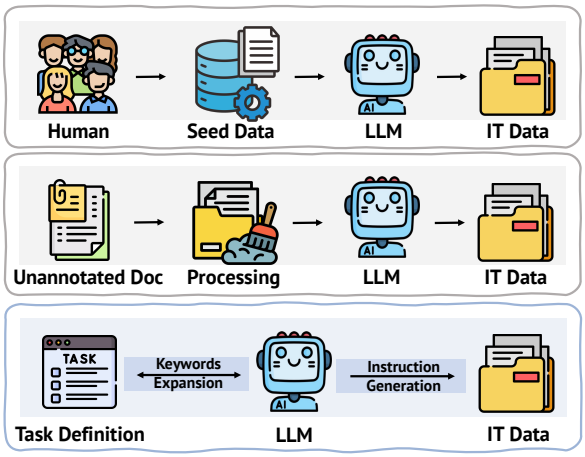


Figure 1: **Top**: Traditional methods require human seed data with manual annotation. **Middle**: Document-based approaches requiring domain-specific corpora and preprocessing. **Bottom**: Our DS^2-INSTRUCT framework generates data using only task definitions through systematic keywords expansion and instruction generation.

Despite the critical importance of domain-specific instruction data, limited work has explored scalable and automatic generation methods tailored for specific tasks or domains (Nayak et al., 2024; Wan et al., 2023). Existing approaches, shown in Figure 1, mainly fall into two main categories. The first category uses domain-specific corpora (Lewkowycz et al., 2022; Taylor et al., 2022; Nayak et al., 2024), but these resources are often proprietary, expensive, or simply unavailable. The second category requires human experts to write seed examples for in-context learning (ICL) prompts (Peng et al., 2023; Zhang et al., 2024; Wang et al., 2023b; Wan et al., 2023). Both approaches demand significant domain expertise, which limits their scalability and hinders efficient domain adaptation.

This paper introduces DS^2-INSTRUCT (**D**omain-

*Corresponding author.

†Co-senior authorship.

[1]Our code is available at https://github.com/rux001/DS2-Instruct.

Findings of the Association for Computational Linguistics: EACL 2026, pages 3368–3384
March 24-29, 2026

Specific Data Synthesis), a zero-shot framework that systematically generates high-quality, domain-specific instruction data. Our framework requires no human supervision, seed examples, or specialized corpora. DS^2-INSTRUCT addresses three core challenges in domain-specific data synthesis:

❶ *How to systematically capture comprehensive domain-specific knowledge?* To ensure *knowledge coverage*, it uses task-informed keyword generation with bi-directional expansion and retrieval augmentation to discover domain concepts from foundational to specialized terminology.

❷ *How to generate cognitively diverse questions?* To promote *instruction diversity*, it applies Bloom's Taxonomy (Bloom et al., 1956; Forehand, 2010) to create instructions across six cognitive levels.

❸ *How can we ensure the generated instruction-response pairs are of high quality?* To guarantee *data quality*, it employs self-consistency filtering (Wang et al., 2023a) to identify and remove flawed instruction-response pairs.

We apply DS^2-INSTRUCT to generate instruction datasets across diverse domains, including mathematics, finance, and biomedical sciences, using `Qwen2.5-72B-Instruct` (Yang et al., 2024) as the generator. We then fine-tune several models, including `Llama3-8B` (AI@Meta, 2024), `Mistral-7B` (Jiang et al., 2023), and `Qwen2.5-7B` (Yang et al., 2024), on our generated data. Evaluations on seven domain-specific benchmarks show consistent improvements over the base models. Our contributions are threefold:

- We introduce a zero-shot framework that generates domain-specific instruction data without requiring human supervision, seed examples, or specialized corpora.
- We show that incorporating cognitive diversity from Bloom's Taxonomy improves a model's reasoning performance.
- We demonstrate through comprehensive evaluation that models trained on our data substantially outperform those trained with existing methods.

2 Related Work

General-Purpose Instruction Data Synthesis. The emergence of instruction-following capabilities in LLMs has been driven by advances in instruction tuning methodologies (Wei et al., 2022; Sanh et al., 2022; Ouyang et al., 2022). Methods for synthesizing general-purpose instruction data have evolved through several paradigms. Early approaches use manually curated datasets to tune models (Mishra et al., 2021; Wei et al., 2022; Sanh et al., 2022; Wang et al., 2022). This manual process is expensive and does not scale. A second paradigm uses LLMs to expand a small set of human-written seeds (Wang et al., 2023b; Taori et al., 2023). Recent work has explored more systematic approaches to data synthesis. Some methods employ keyword-based or topic-guided generation to enhance diversity and coverage (Ding et al., 2023; Mukherjee et al., 2023). A third paradigm employs LLMs to convert unannotated web documents into instruction-tuning datasets (Wang et al., 2022; Jiang et al., 2025).

Domain-Specific Instruction Tuning. While general-purpose instruction tuning methods have shown success, there is increasing demand for domain-specific instruction tuning that remains underexplored. The primary difficulty is acquiring specialized knowledge. One common strategy converts existing domain-specific documents into instruction data (Nayak et al., 2024; Singhal et al., 2023; Katz et al., 2023; Yang et al., 2023). However, such documents are often proprietary or difficult to obtain. Other methods avoid the need for document collections but instead require human-written seed data or are confined to narrow NLP tasks (Zhang et al., 2024; Wan et al., 2023). Concurrently, Guo et al. (2025) leverage explicit task definitions to generate synthetic data for reinforcement learning fine-tuning, while reusing an initial set of 500 seed samples. In contrast, our DS^2-INSTRUCT framework systematically generates domain-specific data without requiring documents, seeds, or human supervision.

3 Method

In this section, we present DS^2-INSTRUCT, a systematic framework for generating high-quality task-specific instruction-tuning datasets that enable effective adaptation of LLMs to specialized tasks. Our approach addresses the critical challenge of creating diverse and task-relevant training data without requiring extensive human annotation.

3.1 Problem Formulation

Let T_{desc} be a unified task description that encompasses both domain knowledge requirements and task format specifications. Our objective is to gen-

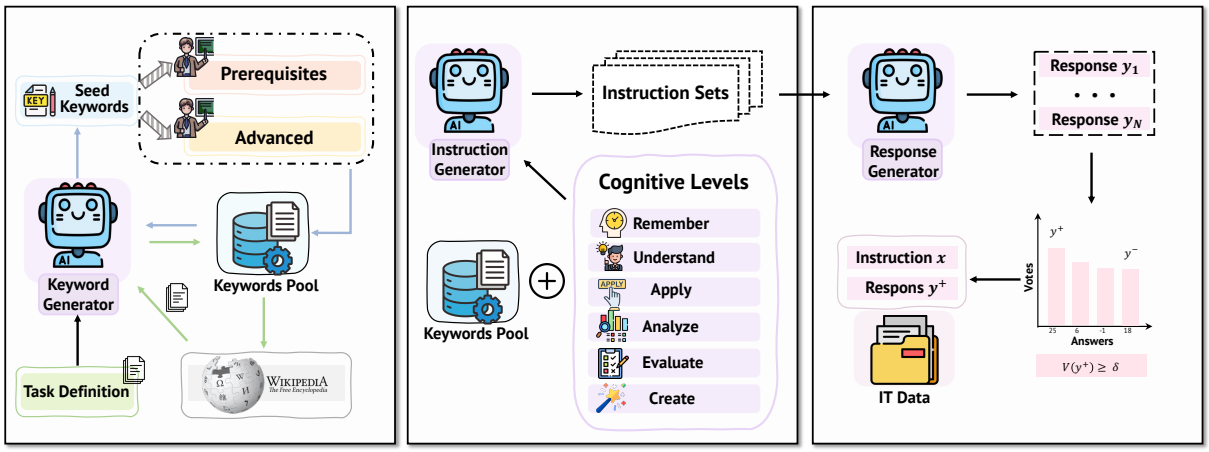


Figure 2: Overview of the DS^2-INSTRUCT framework. Given only a task definition, our approach systematically generates domain-specific instruction-tuning datasets through three stages: ❶ keyword generation using bi-directional expansion and retrieval augmentation, ❷ cognitive-level instruction generation based on Bloom's Taxonomy, and ❸ self-consistency filtering for quality assurance.

erate a high-quality, task-aligned instruction dataset $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{N}$, where x_i represents an instruction and y_i represents the corresponding response.

3.2 Task-Informed Keyword Generation

The keyword generation phase constructs a task-specific knowledge base by systematically identifying and expanding a set of domain-specific concepts. This process begins with a small seed set of keywords derived directly from the task description and iteratively expands it through bi-directional expansion and retrieval augmentation to ensure comprehensive coverage of the relevant knowledge areas while strictly maintaining relevance to the target task.

Initial Keyword Seeding. We initiate the process by generating a foundational set of keywords. Given the task description T_{desc}, we prompt an LLM $\mathcal{M}$ to produce an initial set of n keywords that are most representative of the core domain:

$$K_0 = \text{Parse}(\mathcal{M}(\text{Prompt}_i, T_{desc}, n)) \quad (1)$$

where Prompt_i represents the initial prompt used for keyword generation, and $\text{Parse}(\cdot)$ is a function that extracts a structured list of keywords from the raw textual output of the LLM. This initial set K_0 serves as the foundational knowledge base for subsequent retrieval-augmented expansion.

Bi-Directional Keyword Expansion. To ensure comprehensive coverage of the knowledge space for a specific domain, we propose a bi-directional expansion strategy. Given the initial keyword set K_0, we iteratively use the LLM to systematically generate both prerequisite foundational concepts and advanced specialized topics, enabling full exploration of the domain's knowledge hierarchy.

At each iteration, we randomly sample a subset of keywords from the current pool and use them as in-context learning examples to guide the expansion process. The prerequisite expansion identifies fundamental concepts, basic terminology, and foundational principles that learners must understand before engaging with the sampled concepts. Conversely, the advanced expansion explores specialized subfields, cutting-edge developments, and expert-level topics that build upon the current knowledge base. After each iteration, we integrate the newly generated concepts into the expanding keyword pool. The random sampling ensures that each expansion explores varied conceptual relationships and prevents the generation from converging prematurely on a limited subset of domain knowledge. This systematic and iterative exploration of both foundational and advanced concepts enables comprehensive domain coverage, ensuring that the resulting instruction dataset addresses diverse concepts within the target domain.

Prompt Template for Bi-Directional Keyword Expansion

Task Context: You are an expert in the domain related to: [Task description T_{desc}].
Sample Keywords: [Randomly sampled keywords from current pool].
Instructions: Based on the sample keywords, generate new concepts in two directions:

1. **Prerequisite Concepts**: What fundamental concepts, basic terminology, or foundational principles should learners understand BEFORE studying the sample keywords?

2. **Advanced Concepts**: What specialized subfields, cutting-edge developments, or expert-level topics BUILD UPON the sample keywords?

Retrieval-Augmented Keyword Extraction. After obtaining the expanded keyword set from bi-directional expansion, we enhance our knowledge base through corpus-based retrieval to reduce hallucination risks and ensure deeper domain coverage (Biderman et al., 2022; Izacard et al., 2023; Borgeaud et al., 2022; Lewis et al., 2020). We construct a diverse retrieval corpus by sampling documents from the Pile (Gao et al., 2020), a multidomain collection of high-quality human-written documents. Specifically, we sample from five complementary datasets: ArXiv, FreeLaw, StackExchange, Wikipedia, and Github. This multidomain approach ensures comprehensive coverage across different writing styles, technical depths, and domain-specific terminology.

For retrieval, given the task description T_{desc} and our expanded keyword set, we randomly sample keyword subsets and combine them with the task description to form retrieval queries. For each query, the retriever scores all documents in our corpus based on term frequency, inverse document frequency, and document length normalization, returning the top-k most relevant passages (Robertson and Zaragoza, 2009). These retrieved passages provide authoritative, grounded context about related concepts and domain-specific knowledge. Given the retrieved passages and the current keyword list, we prompt the LLM to extract additional keywords directly from the passage content.

Prompt Template for Retrieval-Augmented Keyword Extraction

Task Context: You are an expert in the domain related to: [Task description T_{desc}].
Current Keywords: [List of expanded keywords].
Retrieved Passages: The following passages contain comprehensive domain knowledge: [Top-k retrieved passages]
Instructions: Extract additional domain-specific keywords directly from the retrieved passages that are missing from the current list.

3.3 Instruction Generation Across Cognitive Levels

Traditional instruction generation techniques often produce instructions that test only surface-level recall or follow repetitive patterns, failing to assess deeper cognitive skills essential for true domain mastery. Drawing inspiration from educational assessment research, we develop a cognitive-task integration framework based on Bloom's Taxonomy (Bloom et al., 1956; Forehand, 2010) that systematically generates instructions across six cognitive levels, from basic recall to creative synthesis, ensuring diverse cognitive complexity while maintaining strict task format compliance.

Cognitive Framework Integration. We systematically integrate Bloom's Taxonomy (Bloom et al., 1956; Forehand, 2010) to ensure comprehensive cognitive coverage in our instruction generation. The taxonomy consists of six hierarchical cognitive levels $\mathcal{B} = \{R, U, Ap, An, E, C\}$ corresponding to *Remember*, *Understand*, *Apply*, *Analyze*, *Evaluate*, and *Create*. By explicitly incorporating each level into our generation process, we ensure that the resulting instructions evaluate diverse cognitive skills ranging from basic knowledge retention to complex creative synthesis:

Cognitive Levels for Instruction Generation

Remembering: Tests factual recall, definitions, and basic terminology.
Understanding: Requires conceptual explanations and interpretations.
Applying: Demands practical application in concrete scenarios.
Analyzing: Involves decomposing problems and identifying patterns.
Evaluating: Requires critical assessment and justified decisions.
Creating: Necessitates synthesizing information into novel solutions.

Keyword Matching Strategies. To maximize instruction diversity, we combine different question types with two complementary keyword approaches. Single-keyword instructions ensure comprehensive coverage of individual concepts across all cognitive levels, while paired-keyword instructions explore relationships between concepts and increase overall diversity through multi-concept integration. For single keywords, we pair each keyword $k_j \in K^*$ with each cognitive level $b \in \mathcal{B}$. For keyword pairs (k_i, k_j), we focus on cognitive levels $\mathcal{B}_{rel} = \{U, Ap, An, E\}$ that naturally accommodate relational reasoning. This generates a comprehensive instruction set covering both individual concept mastery and inter-concept relationships.

Prompt Template for Instruction Generation

Single-Keyword Instructions: Given keyword [k_j] and cognitive level [b], generate a high-quality instruction following task requirements [T_{desc}]. Ensure

the keyword is the central focus and use appropriate domain terminology.
Paired-Keyword Instructions: Given keyword pair $[k_i, k_j]$ and cognitive level $[b]$, generate an instruction that explores relationships between both keywords while following task requirements $[T_{desc}]$. Focus on multi-concept integration and comparative reasoning.

3.4 Instruction Filtering and Response Generation

After generating a large pool of candidate instructions, we employ a systematic filtering and response generation process to ensure high-quality instruction-response pairs. This process consists of two key stages: self-consistency filtering to remove low-quality instructions and final response generation using majority voting.

Self-Consistency Filtering. To identify and remove low-quality instructions where the model exhibits uncertainty, we employ a self-consistency mechanism, which has been used for improving performance on reasoning tasks by generating multiple solutions using LLMs and choosing the most frequent final answer (Wang et al., 2023a; Yu et al., 2025; Shi et al., 2022; Li et al., 2022; Prasad et al., 2025). For each candidate instruction $x \in \mathcal{I}$, we prompt the LLM $\mathcal{M}$ to generate N independent responses $\{y_1, y_2, \ldots, y_N\}$. We then extract the final answer from each response using an answer extraction function $\text{ans}(\cdot)$ and compute the vote function $V(\cdot)$:

$$V(y) = \frac{1}{N} \sum_{m=1}^{N} \mathbf{1}(\text{ans}(y_m) = \text{ans}(y)) \quad (2)$$

The vote function $V(\cdot)$ returns the relative frequency of each final answer, where $\mathbf{1}(\cdot)$ is the indicator function. This approach quantifies instruction quality based on response consistency. The intuition is that it is less likely to consistently generate the same incorrect answer across multiple attempts. Therefore, instructions that consistently produce the same answer across multiple generations are considered high-quality, while those yielding diverse or conflicting responses indicate ambiguity or poor formulation. We filter out instructions where $\arg\max_{y \in \{y_1, \ldots, y_N\}} V(y) < \tau$, retaining only those with sufficient response consistency.

Final Response Generation. For each high-quality instruction $x_i \in \mathcal{I}_{filtered}$, we generate the final response y_i by selecting the answer that achieved the highest vote frequency during the consistency evaluation. The final dataset $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{I}_{filtered}|}$ combines instructions generated from both single-keyword and paired-keyword strategies, providing comprehensive cognitive coverage while maintaining high quality through self-consistency validation.

4 Experiments

4.1 Experimental Setup

Datasets and Benchmark Tasks. To evaluate the ability of DS^2-INSTRUCT to adapt to specific domains, we use seven different publicly available datasets spanning the challenging domains of mathematics, medicine, logical reasoning, science, and finance (Xie et al., 2023; Jin et al., 2021; Liu et al., 2020; Hendrycks et al., 2021; Cobbe et al., 2021; Jin et al., 2019a; Rein et al., 2024). These benchmarks were chosen to test a wide array of reasoning and knowledge-intensive skills. A detailed summary of the datasets, their respective task types, sizes, and primary evaluation metrics is provided in Appendix A.

Baseline Methods. We compare DS^2-INSTRUCT against four baseline approaches: (1) *Zero-Shot* evaluation using vanilla pre-trained models without fine-tuning, (2) a domain-aware version of *Self-Instruct* (Wang et al., 2023b) where we use human-crafted domain-specific seed questions and iteratively expand the instruction collection by randomly selecting five examples during each iteration, (3) *InstructMix* (Xu et al., 2023a) which first extracts core skills using LLM, then generates instruction-response pairs that exhibit randomly chosen combinations of these skills, and (4) *ExploreInstruct* (Wan et al., 2023) that leverages the exploration power of LLMs to actively navigate the domain space and obtain domain-focused instructions.

Base Models. For the main experiment, we use `Qwen2.5-72B-Instruct` (Yang et al., 2024) as the data generator for all baseline methods and DS^2-INSTRUCT to ensure fair comparison across different instruction generation approaches. For fine-tuning evaluation, we employ three target models: `Qwen2.5-7B` (Yang et al., 2024), `Llama-3.1-8B` (Touvron et al., 2023), and `Mistral-7B-v0.1` (Jiang et al., 2023). This configuration allows us to assess the transferability of generated instructions across different model architectures while maintaining consistency in the data generation process.

Implementation Details. To ensure fair compari-

Table 1: Statistics of generated instruction datasets across different domains.

Metric	CFA	PubMedQA	MedQA	GPQA	LogiQA	GSM8K	MATH
Avg. Instruction Length (words) ↑	154.92	119.73	140.65	144.98	169.39	108.49	116.24
Unique Verb-Noun Pairs ↑	5,706	2,572	6,834	6,684	5,211	1,855	1,630
Avg. Pair Occurrences ↓	2.69	2.32	2.16	2.02	2.33	2.25	6.19
Std. Dev. of Occurrences ↓	17.01	5.35	7.54	6.27	18.94	9.94	148.77

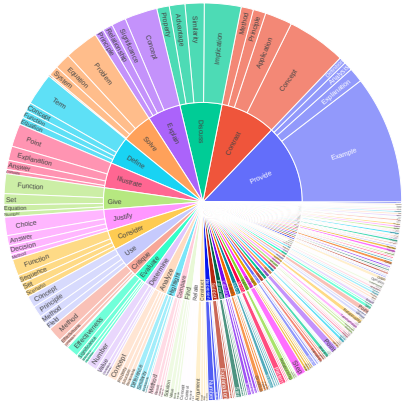
son across all methods, we utilize 6000 instruction-response pairs for each baseline and task. For our proposed DS^2-INSTRUCT, we begin with 50 initial seed keywords and perform bi-directional expansion where we generate 5 new keywords in each direction per iteration, conducting 100 iterations total. We employ LoRA (Hu et al., 2022) for parameter-efficient fine-tuning with rank $r = 8$ and $\alpha = 16$ for parameter-efficient adaptation. For detailed implementation specifications, hyperparameter configurations, and training details for both settings, please refer to Appendix C.

4.2 Data-Centric Analysis

Data Statistics. Table 1 presents essential statistics of the data generated by our proposed DS^2-INSTRUCT framework. We analyze comprehensive metrics including average instruction length, linguistic diversity measures such as unique verb-noun pairs, average occurrence of verb-noun pairs in each instruction, and standard deviation of average occurrences across the dataset. The linguistic diversity metrics are particularly important as they indicate the richness of the generated instructions. Higher numbers of unique verb-noun pairs suggest more varied linguistic patterns, while lower average occurrences indicate better distribution without excessive repetition (Wei et al., 2022; Sanh et al., 2022). The standard deviation of occurrences measures the consistency of term distribution, where lower values generally indicate more balanced vocabulary usage (Longpre et al., 2023).

Diversity. To evaluate instruction diversity, we analyze verb-noun pairs with frequency above 10 in the mathematics domain, comparing Domain-Aware Self-Instruct and DS^2-INSTRUCT in Figure 3. The sunburst plots show root verbs (inner circle) and direct nouns (outer circle), with top 4 pairs displayed for each verb. Figure 3(a) reveals that Domain-Aware Self-Instruct produces highly concentrated distributions with significantly fewer unique verb-noun pairs, clustering around a limited set of computational verbs like "Calculate",

(a) Domain-Aware Self-Instruct



(b) DS^2-INSTRUCT

Figure 3: Root verb-noun pairs in instructions of (a) Domain-Aware Self-Instruct and (b) DS^2-INSTRUCT in the math domain. The inner circle represents root verbs and the outer circle represents direct nouns. For each verb, we show the top 4 verb-noun pairs.

"Solve" and "Find" paired with basic objects such as "Expression" and "Equation". This concentration indicates reduced diversity and potential overfitting to common instruction patterns. In contrast, Figure 3(b) demonstrates that DS^2-INSTRUCT generates substantially more unique verb-noun pairs with more uniformly distributed patterns and diverse mathematical concepts.

Quality. In addition to statistics and diversity of generated data, we evaluate the quality of the generated instructions and responses. Following previous work (Wang et al., 2023b), we randomly sample 500 instruction-response pairs and ask an expert

Table 2: Performance comparison across domain-specific benchmarks. Results show accuracy (%) for multiple-choice tasks and exact match (%) for problem-solving tasks. For each model family, the best results are **bold** and second-best are underlined.

	Methods	Finance	Medicine		Science	Logic	Math		Avg.
		CFA	PubMedQA	MedQA	GPQA	LogiQA	GSM8K	MATH	
Qwen2.5	Zero-Shot	53.71	60.34	48.35	26.32	32.77	71.53	49.44	48.92
	Self-Instruct	55.83	61.47	48.92	**27.18**	42.56	72.38	53.67	51.72
	InstructMix	57.91	62.15	47.83	26.09	43.88	75.42	56.24	53.07
	ExploreInstruct	56.47	60.89	47.15	25.93	41.19	76.67	51.38	51.10
	DS^2-INSTRUCT	**58.34**	**63.62**	**50.98**	26.35	**44.29**	**78.94**	**60.12**	**54.95**
Llama3	Zero-Shot	9.76	24.92	27.38	12.51	9.77	12.60	3.45	14.34
	Self-Instruct	33.72	25.34	31.58	19.87	24.16	19.83	11.47	23.71
	InstructMix	36.89	27.41	33.29	17.52	15.88	35.64	14.92	25.94
	ExploreInstruct	34.58	24.77	30.46	18.24	22.35	33.81	13.06	25.32
	DS^2-INSTRUCT	**56.94**	**48.56**	**39.15**	**30.35**	**35.33**	**58.34**	**23.16**	**42.83**
Mistral	Zero-Shot	4.61	14.92	14.03	8.64	9.52	14.75	1.42	9.70
	Self-Instruct	14.26	26.83	39.74	8.15	29.67	18.52	5.94	20.30
	InstructMix	42.15	31.48	30.15	23.91	26.72	38.26	8.67	30.29
	ExploreInstruct	38.64	28.35	37.92	19.47	25.18	17.93	11.74	24.86
	DS^2-INSTRUCT	**55.72**	**39.11**	**42.99**	**30.16**	**32.23**	**42.75**	**21.82**	**37.83**

Table 3: Quality assessment of generated instruction-response pairs.

Quality Metric	GSM8K (%)	MedQA (%)
Valid instruction	96.83	92.42
Valid response	91.26	94.89
Domain-appropriate	94.91	90.87

annotator (an author of this work) to label whether each instance is valid in terms of instruction clarity, response correctness, and domain appropriateness. Table 3 reports the percentage of valid instructions and responses across datasets. We also provide a detailed analysis of Bloom's Taxonomy to quantify the proportion of generated instructions at each cognitive level and present illustrative case studies in Appendix D.

4.3 Main Results

We evaluate DS^2-INSTRUCT against several representative instruction-generation baselines across seven domain-specific benchmarks and three model families: `Qwen2.5`, `Llama3`, and `Mistral`. As shown in Table 2, DS^2-INSTRUCT consistently achieves the best average performance within each model family, outperforming all baseline methods across nearly all domains.

Notably, the performance gains are most pronounced for weaker base models. For `Mistral`, which exhibits the lowest zero-shot performance, DS^2-INSTRUCT improves the average score to 37.83%. Similarly, on `Llama3`, our method raises the average performance from 14.34% in the zero-shot setting to 42.83%, substantially exceeding prior instruction-generation approaches. Even for the stronger `Qwen2.5` model, DS^2-INSTRUCT delivers consistent improvements. The consistent improvements across diverse domains and model architectures highlight the broad applicability and effectiveness of our framework.

4.4 Additional Experiments

To further validate the robustness and generalizability of our approach, we conduct additional experiments examining two critical dimensions: the impact of training data scale and the effectiveness across models of varying sizes.

Impact of Training Data Scale. We investigate how training data size affects model performance by conducting experiments with varying numbers of training instances ranging from 1,000 to 6,000 samples using `Mistral-7B` as the base model. As illustrated in Figure 4, all three datasets demonstrate consistent performance improvements as training data increases, though with distinct scaling patterns. Notably, even at 6,000 samples, the curves have not fully saturated, suggesting that further performance improvements may be achievable

with larger training datasets. This scaling behavior validates the effectiveness of our data synthesis approach and indicates that the quality of synthesized instructions remains high even as quantity increases.

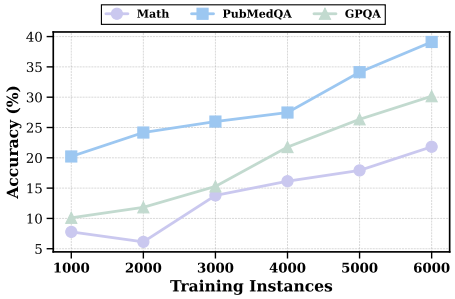


Figure 4: Impact of Training Data Size on Model Performance.

Performance Across Model Scales. To assess the scalability of our approach, we evaluate models of different sizes from the `Qwen` family (3B, 7B, 14B parameters) on Math and GPQA benchmarks. Figure 5 shows that our DS^2-INSTRUCT method consistently outperforms zero-shot baselines across all model scales. Notably, the performance gains diminish as model size increases, suggesting that larger models may rely less heavily on domain-specific instruction data and can better leverage their inherent reasoning capabilities.

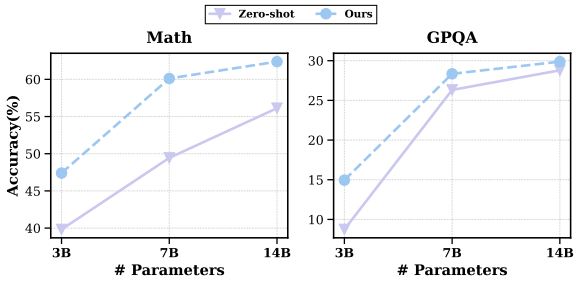


Figure 5: Performance on Models of Different Sizes. We evaluate the Qwen model family on Math and GPQA datasets.

4.5 Ablation Study

To comprehensively evaluate the effectiveness of our DS^2-INSTRUCT framework, we conduct systematic ablation studies analyzing three core components: (1) the keyword generation mechanism with bi-directional expansion and retrieval augmentation, (2) the cognitive level integration based on Bloom's Taxonomy, and (3) the self-consistency filtering approach. These studies are performed across four diverse datasets using `Mistral-7B` to understand each component's individual contribution to overall performance.

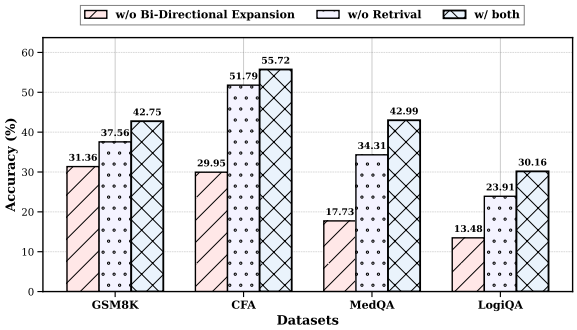


Figure 6: Ablation study evaluating the effectiveness of keyword generation components.

Effectiveness of Keywords Generation. To validate our keyword generation components, we conduct ablation studies across four datasets using `Mistral-7B`. We compare three configurations: (1) *w/o Bi-Directional Expansion*, which uses direct prompting to obtain seed keywords without iterative expansion, (2) *w/ Retrieval*, which expands keywords through LLM generation without Wikipedia-based retrieval, and (3) *w/ Both*, our complete framework with retrieval-augmented expansion. Figure 6 shows the critical importance of both components. Without bi-directional expansion, performance is limited across all datasets. Adding LLM-based expansion provides moderate improvements. However, incorporating retrieval-augmented expansion yields substantial gains, with the full framework outperforming the baseline. The synergistic effect of combining systematic keyword expansion with authoritative Wikipedia sources validates our integrated approach for comprehensive domain coverage.

Effectiveness of Cognitive Level Instruction Generation. To evaluate the impact of incorporating diverse cognitive levels in instruction generation, we compare two configurations using `Mistral-7B` across four datasets: (1) *w/o Cognitive Levels*, which directly prompts the model to generate instructions by providing keywords without any cognitive complexity guidance, and (2) *w/ Cognitive Levels*, our complete framework that systematically incorporates Bloom's Taxonomy to ensure instructions span different cognitive complexities. Figure 7 demonstrates the effectiveness of cognitive-level integration. The baseline approach without cognitive levels relies on simple keyword pairing, allowing the model to generate instructions without explicit guidance on the depth of reasoning required. In contrast, our cognitive-level framework explicitly structures instruction generation around Bloom's Taxonomy, ensuring a balanced distribution across different thinking pro-

cesses, from basic recall to higher order analysis and creation.

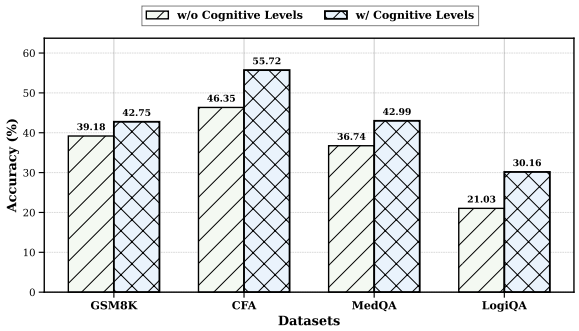


Figure 7: Ablation study evaluating the effectiveness of cognitive level integration in instruction generation.

Effectiveness of Self-Consistency Approach. Our analysis compares performance with and without the self-consistency response filtering component described in Eq. 2. The results, presented in Figure 8, demonstrate substantial and consistent improvements across all evaluated domains. The substantial improvements across all datasets confirm that response consistency serves as an effective proxy for instruction quality, enabling our framework to automatically identify and retain only the most reliable instruction-response pairs for downstream fine-tuning.

Figure 8: Ablation study comparing model performance with and without self-consistency filtering.

5 Conclusion

This paper introduces DS^2-INSTRUCT, a zero-shot framework for generating high-quality, domain-specific instruction datasets without human supervision or seed examples. Our approach generates task-informed keywords for domain coverage, applies Bloom's Taxonomy for cognitive diversity, and uses self-consistency validation for quality filtering. Extensive experiments across seven specialized domains demonstrate that models fine-tuned on DS^2-INSTRUCT generated data substantially outperform existing synthetic instruction methods. Our ablation studies validate the necessity of each component, while scalability experiments across different data sizes and model scales confirm the robustness and generalizability of our approach. These results highlight DS^2-INSTRUCT's effectiveness in enhancing domain-specific reasoning capabilities without requiring expensive human annotation.

Limitations

We identify three primary limitations in our work. First, our framework relies on static knowledge bases for keyword expansion, which may introduce coverage bias toward well-documented topics while potentially overlooking emerging or niche subfields. Incorporating dynamic knowledge retrieval or domain-specific corpora could help address this limitation. Second, the output quality depends on the generator LLM. Any biases or knowledge gaps in the generator can propagate to the synthesized data. Third, we evaluate our method only on English benchmarks, so its multilingual performance remains untested.

References

AI@Meta. 2024. The llama 3 herd of models. Technical report, Meta AI. Https://ai.meta.com/research/publications/the-llama-3-herd-of-models/.

Stella Biderman, Kieran Bicheno, and Leo Gao. 2022. Datasheet for the pile. *arXiv preprint arXiv:2201.07311*.

Benjamin S Bloom and 1 others. 1956. *Taxonomy of educational objectives: The classification of educational goals*. Longmans, Green.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, and 1 others. 2022. Improving language models by retrieving from trillions of tokens. In *ICML*.

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

Mitchell Conover, Siyan Liu, Binh Do, Mahdi Shokr, Sijia Tang, An-Ting Yu, and Andrew Dai. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, Singapore. Association for Computational Linguistics.

Mary Forehand. 2010. Bloom's taxonomy. *Emerging perspectives on learning, teaching, and technology*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, and 1 others. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Yiduo Guo, Zhen Guo, Chuanwei Huang, Zi-Ang Wang, Zekai Zhang, Haofei Yu, Huishuai Zhang, and Yikang Shen. 2025. Synthetic data rl: Task definition is all you need. *arXiv preprint arXiv:2505.17063*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Arjun Agarwal, Steven Levi, Kevin Ellis, and 1 others. 2021. Measuring mathematical problem solving with the math dataset. In *NeurIPS*.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *ACL*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. In *JMLR*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Yuxin Jiang, Yufei Wang, Chuhan Wu, Xinyi Dai, Yan Xu, Weinan Gan, Yasheng Wang, Xin Jiang, Lifeng Shang, Ruiming Tang, and Wei Wang. 2025. Instruction-tuning data synthesis from scratch via web reconstruction. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6603–6618, Vienna, Austria. Association for Computational Linguistics.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019a. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W Cohen, and Xinghua Lu. 2019b. Pubmedqa: A dataset for biomedical research question answering. In *EMNLP*.

Daniel Martin Katz, Michael James Bommarito, Shang Gao, and Pablo Arredondo. 2023. Gpt-4 passes the bar exam. *Available at SSRN 4389233*.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagy, and 1 others. 2023. Openassistant conversations–democratizing large language model alignment. *arXiv preprint arXiv:2304.07327*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, and 1 others. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, and 1 others. 2022. Competition-level code generation with alphacode. *Science*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In *IJCAI*.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. In *ICML*.

Deepanway Mehrabi, Harsh Bhargava, and Raghav Goyal. 2023. Flacuna: Unleashing the problem solving power of vicuna using flan fine-tuning. *arXiv preprint arXiv:2307.02053*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773*.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*.

Nihal V. Nayak, Yiyang Nan, Avi Trost, and Stephen H. Bach. 2024. Learning to generate instruction tuning datasets for zero-shot task adaptation. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Archiki Prasad, Weizhe Yuan, Richard Yuanzhe Pang, Jing Xu, Maryam Fazel-Zarandi, Mohit Bansal, Sainbayar Sukhbaatar, Jason Weston, and Jane Yu. 2025. Self-consistency preference optimization. In *ICML*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, and 1 others. 2022. Multitask prompted training enables zero-shot task generalization. In *ICLR*.

Freda Shi, Daniel Fried, Marjan Ghazvininejad, Luke Zettlemoyer, and Sida I. Wang. 2022. Natural language to code translation with execution. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3533–3546, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, and 1 others. 2023. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

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. *arXiv preprint arXiv:2211.09085*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Fanqi Wan, Xinting Huang, Tao Yang, Xiaojun Quan, Wei Bi, and Shuming Shi. 2023. Explore-instruct: Enhancing domain-specific instruction coverage through active exploration. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9435–9454, Singapore. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023a. Self-consistency improves chain of thought reasoning in language models. In *ICLR*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023b. Self-instruct: Aligning language model with self generated instructions. In *ACL*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023c. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, and 16 others. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. *International Conference on Learning Representations*.

Qianqian Xie, Weiguang Han, Xiao Zhang, Yanzhao Lai, Min Peng, Alejandro Lopez-Lira, and Jimin Huang. 2023. Pixiu: A large language model, instruction data and evaluation benchmark for finance. *Preprint*, arXiv:2306.05443.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin

Jiang. 2024. Wizardlm: Empowering large language models to follow complex instructions. In *ICLR*.

Fei Xu, Qingyan Wang, Jiuding Li, Wenjie Zhong, Zhaoxiang Zhang, and Qi Liu. 2023a. Data advisor: Dynamic data curation for safety alignment of large language models. In *EMNLP*.

Yixin Xu, Siyuan Chen, Yutong Wu, Chenguang Li, and Cheng Li. 2023b. Instruction-tuned language models are better listeners. *arXiv preprint arXiv:2308.12038*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, and 1 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. 2023. Fingpt: Open-source financial large language models. *arXiv preprint arXiv:2306.06031*.

Shusheng Yin, Mo Al-Khamissi, Marwan Al-Khamissi, Feras El-Kurdi, Keng-Hwee Lee, Jungo Kim, and Graham Neubig. 2023. Instruction-tuning with synthesized data. *arXiv preprint arXiv:2304.06029*.

Ping Yu, Jack Lanchantin, Tianlu Wang, Weizhe Yuan, Olga Golovneva, Ilia Kulikov, Sainbayar Sukhbaatar, Jason Weston, and Jing Xu. 2025. Cot-self-instruct: Building high-quality synthetic prompts for reasoning and non-reasoning tasks. *arXiv preprint arXiv:2507.23751*.

Zhen Yuan, Hong Liu, Dong Zhao, and Rui Yan. 2024. Self-alignment of large language models with instruction-reward. *arXiv preprint arXiv:2401.10353*.

Yue Zhang, Michel Galley, Jianfeng Gao, and Bill Dolan. 2024. Targen: Targeted data generation with large language models. In *COLM*.

A Dataset Details

We evaluate our DS^2-INSTRUCT framework across seven challenging domain-specific benchmarks spanning mathematics, finance, science, logical reasoning, and biomedicine. These benchmarks were selected to test diverse reasoning capabilities and domain-specific knowledge across multiple disciplines.

The datasets encompass the following domains and characteristics:

- **CFA** (Xie et al., 2023): Chartered Financial Analyst exam questions testing asset valuation, portfolio management, wealth planning, and ethical standards. Questions require fundamental knowledge understanding, quantitative analysis, and practical application of financial concepts.
- **PubMedQA** (Jin et al., 2019b): Biomedical research questions requiring yes/no/maybe answers based on corresponding abstracts from PubMed. This task demands strong comprehension and interpretation of biomedical literature and scientific reasoning.
- **MedQA** (Jin et al., 2021): Medical board exam questions testing professional-level knowledge across physiology, pathology, pharmacology, and clinical reasoning. Questions simulate real medical licensing examinations.
- **GPQA** (Rein et al., 2024): Graduate-level questions in Physics, Chemistry, and Biology requiring deep subject-matter knowledge, complex reasoning, calculation, and synthesis of advanced scientific concepts.
- **LogiQA** (Liu et al., 2020): Logical reasoning problems with multiple-choice questions that test the ability to understand logical structures, identify fallacies, and make valid inferences from given premises.
- **GSM8K** (Cobbe et al., 2021): Grade school math word problems involving basic arithmetic, algebra, and geometry. Problems require step-by-step reasoning and mathematical problem-solving skills.
- **MATH** (Hendrycks et al., 2021): Competition-level mathematics problems covering algebra, geometry, number theory, and calculus. These problems require sophisticated mathematical reasoning and step-by-step solution derivation.

B Additional Data Statistics

Table 5 shows statistics of data generated by Self-Instruct (Wang et al., 2023b).

C Implementation Details

For fair comparison across all methods, we generate 6,000 instruction-response pairs for each baseline and task. Our DS^2-INSTRUCT framework begins with 50 initial seed keywords and performs bi-directional expansion, generating 5 new keywords in each direction per iteration over 100 total iterations.

C.1 Training Configuration

We employ LoRA for efficient fine-tuning of pre-trained LLMs. Table 6 presents the complete training hyperparameters used across all experiments.

Table 4: Overview of benchmark datasets used for evaluation.

Dataset	Domain	Test Size	Question Type	Evaluation Metric
CFA (Xie et al., 2023)	Finance	1,506	Multiple Choice	Accuracy
PubMedQA (Jin et al., 2019b)	Biomedical	1,000	Yes/No/Maybe	Accuracy
MedQA (Jin et al., 2021)	Medical	1,273	Multiple Choice	Accuracy
GPQA (Rein et al., 2024)	Graduate Science	448	Multiple Choice	Accuracy
LogiQA (Liu et al., 2020)	Logical Reasoning	651	Multiple Choice	Accuracy
GSM8K (Cobbe et al., 2021)	Mathematics	1,319	Problem-Solving	Exact Match
MATH (Hendrycks et al., 2021)	Competition Math	5,000	Problem-Solving	Exact Match

Table 5: Statistical analysis of Self-Instruct generated datasets across different domains, including verb-noun (V-N) pair diversity metrics.

Metric	CFA	PubMedQA	MedQA	GPQA	LogiQA	GSM8K	MATH
Avg. Length (words) ↑	85.65	53.33	91.92	41.98	79.91	46.39	37.69
Unique Verb-Noun Pairs ↑	505	45	333	287	196	117	68
Avg. Occurrences ↓	28.56	63.60	12.36	3.96	18.60	27.73	8.00
Std. Dev. Occurrences ↓	160.31	129.81	49.70	8.66	88.05	115.53	8.77

Table 6: Fine-tuning hyperparameters for LoRA adaptation.

Hyperparameter	Value
LoRA Rank (r)	8
LoRA Alpha (α)	16
Learning Rate	2e-5
Training Epochs	5
Batch Size	16
Weight Decay	0.01
Optimizer	AdamW

C.2 Data Generation Configuration

Our data generation pipeline integrates Retrieval-augmented keywords expansion and self-consistency filtering to ensure high-quality instruction-response pairs.

Keyword Generation Configuration. Our keyword expansion strategy initializes with 50 seed keywords and generates 5 new keywords in each direction (forward and backward) per iteration. The expansion process runs for 100 iterations. For retrieval-augmented generation, we construct a diverse corpus by sampling from five complementary datasets in the Pile (Gao et al., 2020): ArXiv, FreeLaw, StackExchange, Wikipedia, and Github. This multidomain approach ensures comprehensive coverage across different writing styles and domain-specific terminology. We employ BM25 (Robertson and Zaragoza, 2009) as our ranking function. For each retrieval query, we retrieve the top-5 most relevant documents from the corpus.

Self-Consistency Filtering. To ensure response quality, we apply self-consistency filtering with a threshold $\tau = 3/5$. For each generated instruction, we sample $N = 5$ response generations with temperature 0.7 and select responses that achieve majority agreement. The maximum generation length is set to 2048.

D Detailed Analysis of Cognitive Levels

D.1 Case Study

To illustrate how DS^2-INSTRUCT generates cognitively diverse instructions across different levels of Bloom's Taxonomy, we present a detailed case study from the logical reasoning domain. We demonstrate how our framework systematically creates questions ranging from basic recall to complex creative tasks, all centered on domain-specific concepts. Table 7 presents these generated instruction-response pairs, showcasing the cognitive diversity achieved by our approach.

D.2 Bloom Level Distribution Analysis

To systematically evaluate cognitive demands across reasoning benchmarks, we randomly sampled 500 samples and conducted human annotation following Bloom's Taxonomy classification guidelines. Table 8 presents distributions across GPQA, GSM8K, and MedQA, revealing distinct cognitive profiles.

Cognitive Level	Generated Instruction
Remembering	What is the definition of satisfiability in the context of logic and problem-solving? **Options:** A) The ability to prove that a statement is true in all possible scenarios. B) The process of finding a solution that meets all specified constraints. C) The method of determining the truth value of a logical expression. D) The technique of simplifying complex logical expressions to their basic components.
Understanding	Which statement best describes the roles of formal proof techniques and symbolic logic in constructing logical arguments? **Options:** A) Formal proof techniques and symbolic logic are both essential tools in constructing logical arguments, each offering unique advantages in terms of clarity and rigor. B) Symbolic logic is the preferred method for constructing logical arguments, as it eliminates the need for formal proofs. C) Formal proof techniques are less precise than symbolic logic, but they are faster and easier to use in logical arguments. D) Neither formal proof techniques nor symbolic logic provide significant value in constructing logical arguments.
Applying	In a logical reasoning test, you are given the statements: 'If the system is secure, then the data is protected' ($P \rightarrow Q$) and 'The data is not protected' ($\neg Q$). Using modus tollens, what can you conclude about the security of the system? **Options:** A) The system is secure. B) The data is protected. C) The system is not secure. D) The system's security status cannot be determined from the given statements.
Analyzing	In the context of logical reasoning, compare and contrast the application of "basic set operations" and "verified observation" in constructing a valid argument. How do these concepts interact when evaluating the strength and validity of an argument? **Options:** A) Basic set operations are used to verify observations, ensuring that the logical structure of an argument is sound. B) Verified observation is a prerequisite for applying basic set operations, as it provides the necessary empirical data. C) Basic set operations and verified observation are independent processes that do not influence each other in logical reasoning. D) Basic set operations can be used to manipulate sets of verified observations to test the consistency and validity of an argument.
Evaluating	Evaluate the following statement: "All bachelors are unmarried men, which is a truth known a priori." How does this statement relate to the concept of synthetic a priori knowledge, and what is the nature of the conclusion drawn? **Options:** A) The statement is an example of analytic a priori knowledge, not synthetic a priori, because the predicate 'unmarried' is contained within the subject 'bachelor.' B) The statement is an example of synthetic a priori knowledge because it combines a priori reasoning with empirical facts about bachelors. C) The statement is neither analytic nor synthetic a priori because it relies on empirical observation to be true. D) The statement is an example of synthetic a posteriori knowledge because it requires empirical verification to confirm that all bachelors are indeed unmarried.
Creating	Design a logical reasoning problem where two events, A and B, always occur together (constant conjunction), but it is not clear whether one causes the other. Present a scenario and ask students to identify which additional information would best help determine if there is a causal relationship between A and B, or if their conjunction is coincidental. **Example Scenario:** In a small town, every time the town clock strikes 12, a flock of birds flies over the town square. This has been observed consistently for the past year. **Options:** A) The type of birds observed flying over the town square. B) The time of day when the birds are observed flying over the town square. C) Whether the birds still fly over the town square when the clock is temporarily stopped. D) The weather conditions when the birds are observed flying over the town square.

Table 7: Case study demonstrating DS^2-INSTRUCT's generation of cognitively diverse questions across all six levels of Bloom's Taxonomy for the logical reasoning domain.

Bloom's Taxonomy Level	GPQA (%)	GSM8K (%)	MedQA (%)
Remembering	5.5	9.2	7.8
Understanding	36.0	16.0	18.3
Applying	20.8	20.4	32.9
Analyzing	24.0	25.2	23.3
Evaluating	9.7	15.8	10.4
Creating	4.0	13.4	7.3
Aggregate Cognitive Categories			
Lower-Order (Remember + Understand)	41.5	25.2	26.1
Middle-Order (Apply + Analyze)	44.8	45.6	56.2
Higher-Order (Evaluate + Create)	13.7	29.2	17.7

Table 8: Bloom's Taxonomy distribution across benchmarks based on human annotation of 500 randomly sampled questions. GPQA emphasizes conceptual understanding (36.0%), GSM8K shows balanced higher-order demands (29.2%), and MedQA concentrates on procedural application (32.9%).

E Prompt Templates

Cognitive Levels for Instruction Generation

Remembering: Create instructions that emphasize recall of factual knowledge, definitions, basic concepts, recognition tasks, and core terminology related to the keyword.
Understanding: Design instructions that require conceptual understanding, explanation of relationships, interpretation, illustrative examples, and meaningful comparisons involving the keyword.
Applying: Formulate instructions that demand practical use of methods, implementation of procedures, execution of calculations, and real-world application of the keyword.
Analyzing: Develop instructions that involve breaking down complex ideas, identifying patterns, examining relationships, and conducting comparative or structural analysis of the keyword.
Evaluating: Construct instructions that involve critical judgment, validation of techniques, assessment of alternatives, justification of decisions, and critique of methods related to the keyword.
Creating: Design instructions that foster original thinking, synthesis of ideas, problem innovation, creative design, and novel applications of the keyword.

E.1 Task-Specific Initial Keyword Generation Prompts

CFA (Finance) - Initial Keywords

Task Context: You are an expert in Finance and CFA.
Task Description: Your task is to answer CFA exam questions in a multi-choice form, you should select the correct answer choice (e.g., A, B, C). These questions are about asset valuation, applying investment tools and concepts to analyze various investments, portfolio management, wealth planning, ethical and professional standards. It requires skills about Fundamental Knowledge Understanding, Quantitative Analysis and Calculations, Application and Analysis, etc.
Instructions: Generate 50 core keywords that represent the most essential concepts for this task.
Requirements:

- List exactly 50 core concepts separated by commas
- Use underscores for multi-word concepts (e.g., asset_valuation)
- Single words are acceptable (e.g., portfolio)
- Provide only the comma-separated list without any other text

Core Keywords:

GSM8K (Mathematics) - Initial Keywords

Task Context: You are an expert in Mathematics.
Task Description: You are given a math word problem involving basic arithmetic, algebra, or geometry. Your task is to carefully read the problem and provide a step-by-step solution.
Instructions: Generate 50 core keywords that represent the most essential concepts for this task.
Requirements:

- List exactly 50 core concepts separated by commas
- Use underscores for multi-word concepts (e.g., word_problem)
- Single words are acceptable (e.g., addition)
- Provide only the comma-separated list without any other text

Core Keywords:

PubMedQA (Biomedical) - Initial Keywords

Task Context: You are an expert in Biomedical Research.
Task Description: Your task is to answer biomedical research questions based on corresponding scientific abstracts. You must determine whether the answer to the question is "yes," "no," or "maybe" by analyzing the provided scientific text. This requires strong comprehension and interpretation of biomedical literature.
Instructions: Generate 50 core keywords that represent the most essential concepts for this task.
Requirements:

- List exactly 50 core concepts separated by commas
- Use underscores for multi-word concepts (e.g., clinical_trial)
- Single words are acceptable (e.g., treatment)
- Provide only the comma-separated list without any other text

Core Keywords:

LogiQA (Logical Reasoning) - Initial Keywords

Task Context: You are an expert in Logical Reasoning.
Task Description: Your task is to solve logical reasoning problems multiple-choice questions. This involves analyzing a given logical puzzle or argument and choosing the correct option from a set of multiple-choice answers. These questions test your ability to understand logical structures, identify fallacies, and make valid inferences.
Instructions: Generate 50 core keywords that represent the most essential concepts for this task.
Requirements:

- List exactly 50 core concepts separated by commas
- Use underscores for multi-word concepts (e.g., logical_fallacy)
- Single words are acceptable (e.g., premise)
- Provide only the comma-separated list without any other text

Core Keywords:

GPQA (Graduate Science) - Initial Keywords

Task Context: You are an expert in Graduate-level Science.
Task Description: Your task is to answer challenging, graduate-level multiple-choice questions spanning Physics, Chemistry, and Biology, requiring deep subject-matter knowledge, complex reasoning, calculation, and synthesis of information.
Instructions: Generate 50 core keywords that represent the most essential concepts for this task.
Requirements:

- List exactly 50 core concepts separated by commas
- Use underscores for multi-word concepts (e.g., quantum_mechanics)
- Single words are acceptable (e.g., thermodynamics)
- Provide only the comma-separated list without any other text

Core Keywords:

MATH (Competition Mathematics) - Initial Keywords

Task Context: You are an expert in Competition Mathematics.
Task Description: You are given a challenging competition math problem. Your task is to carefully read the problem and provide a step-by-step solution for it.
Instructions: Generate 50 core keywords that represent the most essential concepts for this task.
Requirements:

- List exactly 50 core concepts separated by commas
- Use underscores for multi-word concepts (e.g., number_theory)
- Single words are acceptable (e.g., calculus)
- Provide only the comma-separated list without any other text

Core Keywords:

MedQA (Medical Board Exams) - Initial Keywords

Task Context: You are an expert in Medical Knowledge.
Task Description: Your task is to answer medical questions collected from the professional medical board exams. These questions test professional-level knowledge across a broad range of medical domains, including physiology, pathology, pharmacology, and clinical reasoning. You should select the correct answer choice (e.g., A, B, C, or D).
Instructions: Generate 50 core keywords that represent the most essential concepts for this task.
Requirements:

- List exactly 50 core concepts separated by commas
- Use underscores for multi-word concepts (e.g., differential_diagnosis)
- Single words are acceptable (e.g., pharmacology)
- Provide only the comma-separated list without any other text

Core Keywords:

E.2 Task-Specific Instruction Generation Templates

Instruction Generation Template

Task Description: [TASK_DESCRIPTION]
Keyword: [KEYWORD]
Question Type: [COGNITIVE_LEVEL] - [COGNITIVE_LEVEL_DESCRIPTION]
Generate a high-quality question that precisely targets the keyword and question type described above. The question should be clear, unambiguous, and suitable for instruction tuning.
Directly output the question. Do not include the answer or any other text.
Generated Question:

E.3 Task-Specific Response Suffixes

Response Format Suffixes by Task

Multiple-Choice Tasks (CFA, LogiQA, MedQA, GPQA): Return exactly two lines and nothing else: `Reason: <1-3 sentence explanation> Answer: <A|B|C|D>`

PubMedQA: Return exactly two lines and nothing else: `Reason: <1-3 sentence explanation> Answer: <yes|no|maybe>`

GSM8K: Provide a step-by-step reasoning process and then write the final numerical answer on a new line in the format: `final answer: <answer>`.

MATH: Provide a step-by-step reasoning process and then write the final answer in the LaTeX boxed tag: `$\boxed{<answer>}$`.

E.4 Universal Expansion and Generation Templates

Bi-Directional Keyword Expansion Template

Task Context: You are an expert in the domain related to: [Task description T_{desc}].
Sample Keywords: [Randomly sampled keywords from current pool].
Instructions: Based on the sample keywords, generate new concepts in two directions:
1. **Prerequisite Concepts**: What fundamental concepts, basic terminology, or foundational principles should learners understand BEFORE studying the sample keywords?
2. **Advanced Concepts**: What specialized subfields, cutting-edge developments, or expert-level topics BUILD UPON the sample keywords?
Requirements:

- Generate 5 concepts for each direction
- Use underscores for multi-word concepts
- Ensure concepts are different from existing keywords
- Provide comma-separated lists

Retrieval-Augmented Keyword Extraction Template

Task Context: You are an expert in the domain related to: [Task description T_{desc}].
Current Keywords: [List of expanded keywords].
Retrieved Passages: The following passages from authoritative sources (ArXiv, FreeLaw, StackExchange, Wikipedia, Github) contain comprehensive domain knowledge: [Top-k retrieved passages]
Instructions: Extract additional domain-specific keywords directly from the retrieved passages that are missing from the current list.
Extracted Keywords:

E.5 Zero-shot Evaluation Prompts

Zero-shot Evaluation Template

General Structure:
[Task Description T_{desc}]
[Question from benchmark]
[Task-specific response format suffix]

Example for Multiple-Choice Tasks:
You are tasked with solving finance/medical/science questions.
Question: [Test question with options A, B, C, D]
Return exactly two lines and nothing else:
`Reason: <1-3 sentence explanation>`
`Answer: <A|B|C|D>`

Example for Math Tasks (GSM8K/MATH):
You are tasked with solving mathematical problems.
Question: [Math problem]
Provide a step-by-step reasoning process and write the final answer in the format: `final answer: <answer>` (for GSM8K) or `$\boxed{<answer>}$` (for MATH).

F Ethics Statement

We recognize the importance of evaluating and mitigating potential biases in automatically generated instruction data. To maintain transparency, we provide comprehensive details about our methodology, data sources, and evaluation procedures to enable reproducibility and critical assessment by the research community. Privacy is preserved throughout our framework by relying exclusively on publicly available resources and avoiding the collection or processing of sensitive personal information. Additionally, we disclose that ChatGPT was used exclusively for minor grammatical improvements and formatting corrections in the final manuscript preparation, without contributing to the core research methodology, analysis, or conclusions presented in this work.