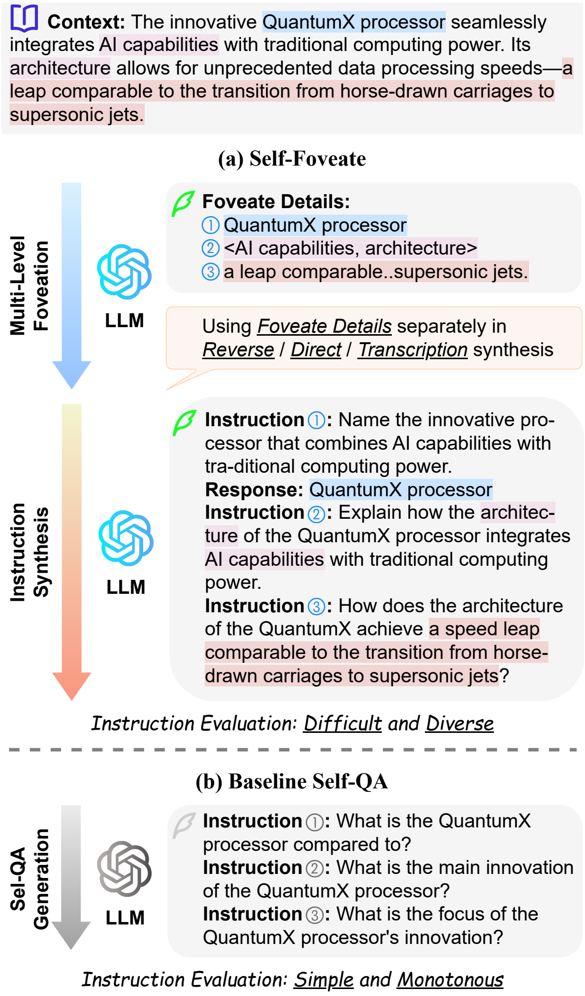
arXiv:2507.23440v1 [cs.AI] 31 Jul 2025

SELF-FOVEATE: Enhancing Diversity and Difficulty of Synthesized Instructions from Unsupervised Text via Multi-Level Foveation

Mingzhe Li, Xin Lu, Yanyan Zhao[*]
Research Center for Social Computing and Interactive Robotics
Harbin Institute of Technology, China
{mzli, xlu, yyzhao}@ir.hit.edu.cn

Abstract

Large language models (LLMs) with instruction following capabilities have demonstrated impressive problem-solving abilities. While synthesizing instructional data from unsupervised text has become a common approach for training such models, conventional methods rely heavily on human effort for data annotation. Although existing automated synthesis paradigms have alleviated this constraint, they still exhibit significant limitations in ensuring adequate diversity and difficulty of synthesized instructions. To address these challenges, we propose SELF-FOVEATE, an innovative LLM-driven method for instruction synthesis. This approach introduces a "Micro-Scatter-Macro" multi-level foveation methodology that effectively guides the LLM to deeply excavate fine-grained information embedded in unsupervised text, thereby enhancing both the diversity and difficulty of synthesized instructions. Comprehensive experiments across multiple unsupervised corpora and diverse model architectures validate the effectiveness and superiority of our proposed method. We publicly release our data and codes: https://github.com/Mubuky/Self-Foveate



Context: The innovative QuantumX processor seamlessly integrates AI capabilities with traditional computing power. Its architecture allows for unprecedented data processing speeds—a leap comparable to the transition from horse-drawn carriages to supersonic jets.

(a) Self-Foveate

Foveate Details:
① QuantumX processor
② <AI capabilities, architecture>
③ a leap comparable..supersonic jets.

Using *Foveate Details* separately in *Reverse* / *Direct* / *Transcription* synthesis

Instruction ①: Name the innovative processor that combines AI capabilities with tra-ditional computing power.
Response: QuantumX processor
Instruction ②: Explain how the architecture of the QuantumX processor integrates AI capabilities with traditional computing power.
Instruction ③: How does the architecture of the QuantumX achieve a speed leap comparable to the transition from horse-drawn carriages to supersonic jets?

Instruction Evaluation: Difficult and Diverse

(b) Baseline Self-QA

Instruction ①: What is the QuantumX processor compared to?
Instruction ②: What is the main innovation of the QuantumX processor?
Instruction ③: What is the focus of the QuantumX processor's innovation?

Instruction Evaluation: Simple and Monotonous

Figure 1: Illustration of (a) SELF-FOVEATE in contrast with (b) Baseline Self-QA. For SELF-FOVEATE, the Multi-Level Foveation enables the LLM to extract details (highlighted in distinct colors) of the text, subsequently synthesizing instructions with diversity and difficulty via distinct synthesis paradigms. In comparison, Self-QA employs single-step generation that produces instruction candidates with simplicity and monotony.

1 Introduction

Large language models (LLMs), such as GPT-4o (OpenAI, 2024), Claude 3.5 Sonnet (Anthropic, 2024), and Llama 3.1 (Meta, 2024), have garnered significant attention due to their exceptional instruction-following capabilities (Zhou et al., 2023b), with their continuously enhanced problem-solving abilities (Cobbe et al., 2021; Dua et al., 2019) being increasingly recognized. A critical component in training such models typically involves fine-tuning with extensive supervised question-answering instruction data (Wang et al., 2023b). However, substantial challenges persist in constructing large-scale, high-quality supervised fine-tuning (SFT) data for instruction tuning (Zhang et al., 2024). The prohibitive costs associated with human annotation (Wang et al., 2023a;

[*] Email corresponding.

Related Work	Data Utilization	*w/o* Human Annotation	Correctness Guarantee	Diversity Augment	Difficulty Augment
Self-Instruct (Wang et al., 2023a)	Seed QA examples	❌	❌	✅	❌
Self-Align (Sun et al., 2023)	Seed QA examples	❌	❌	✅	✔
Self-Chat (Xu et al., 2023)	Dialogue	✅	❌	❌	❌
Self-QA (Zhang and Yang, 2023)	Unsupervised Knowledge	✅	✅	✔	❌
LongForm (Köksal et al., 2024)	Web dataset	✅	✅	✅	❌
Humpback (Li et al., 2024b)	Web dataset	❌	✅	❌	✔
ISARA (Guo et al., 2024)	Seed QA examples	✅	✅	✅	❌
Wiki2023 (Jiang et al., 2024)	Unsupervised Text	✔	✅	❌	❌
Self-Foveate (Ours)	Unsupervised Text	✅	✅	✅	✅

Table 1: A comparative analysis of various task generation methodologies or frameworks. The gray checkmark symbol denotes that the work may partially accomplish specific objectives (though not comprehensively). The red cross marker indicates either the work's failure to achieve the stated objective or absence of explicit documentation regarding this goal.

Sun et al., 2023; Xu et al., 2023), coupled with difficulties in ensuring data diversity and quality control (Ge et al., 2024), continue to impede technological advancements. Given the demonstrated excellence and ongoing improvements in LLMs' instruction-following and generative capabilities, researchers are actively exploring effective methodologies to leverage these models for synthetic data generation (Wang et al., 2023a; Zhang and Yang, 2023; Nayak et al., 2024b; Wu et al., 2024). The primary objectives are to produce high-quality, cost-efficient datasets that reduce reliance on expensive human annotation (Ling et al., 2024), while enhancing the diversity and difficulty of automatically synthesized instructions to improve the performance of fine-tuned models on downstream tasks.

A promising recent paradigm in synthetic instruction synthesis is *unsupervised text-based instruction synthesis*. The advantages of this paradigm become particularly pronounced when handling massive unsupervised text corpora – a ubiquitous resource containing rich world knowledge and linguistic patterns. By leveraging LLMs' intrinsic capabilities in contextual understanding and logical reasoning, this paradigm eliminates the need for manual annotation while confining the scope of instruction synthesis to the given unsupervised textual materials. The pioneering work by Zhang and Yang (2023) has demonstrated the feasibility of extracting instructions from unsupervised textual data.

Despite advancements in synthesizing instruction data from unsupervised text and the proposal of automated LLM-based methods like Self-QA (Zhang and Yang, 2023) to mitigate these issues – approaches that have significantly optimized the data generation pipeline while reducing human labor costs – as shown in Table 1, the following challenges remain unresolved: **(1) Diversity of Instructions:** While existing frameworks continuously refine data generation strategies and enhance post-synthesis filtering for instruction data, limitations persist in the diversity of synthesized instructions. Models trained on such synthetic datasets frequently exhibit insufficient generalization capabilities and may even suffer from performance degradation. **(2) Difficulty of Instructions:** Current synthesis methodologies generally lack emphasis on instruction complexity and depth. For instance, Self-QA (Zhang and Yang, 2023) directly acquires instructions through single-step generation without guaranteed difficulty levels (as illustrated in Figure 1). Synthesized instructions often demonstrate simplistic structures and inadequate comprehension of inter-entity relationships, failing to effectively stimulate models to produce nuanced responses. This deficiency becomes particularly pronounced when handling complex queries or generating contextually rich responses. The absence of high-difficulty instruction design constrains model performance in real-world applications requiring advanced cognitive engagement and problem-solving capabilities. This collective evidence indicates that unsupervised text data still harbors substantial untapped potential for instruction synthesis.

To address these challenges, we attempt to leverage unsupervised text itself, observing that textual data inherently contains abundant detailed information encompassing entities (*e.g.*, *QuantumX proces-*

sor), attributes, relations, and writing techniques – such as the metaphorical comparison of "*speed*" to "*not just a luxury*" in the statement *"In the realm of technology, speed is not just a luxury"*, which implicitly analogizes speed as a fundamental necessity through *subtle analogy*, thereby emphasizing its critical importance (as multi-color annotated in Figure 1). This information remains underutilized in existing methods like SELF-QA.

Motivated by these observations, this paper proposes SELF-FOVEATE, a comprehensive LLM-based methodology designed to automatically synthesize instructions from unsupervised text. Diverging from prior research, as illustrated in Figure 1, SELF-FOVEATE introduces a "Micro-Scatter-Macro" multi-level foveation methodology to comprehensively excavate detailed information from raw text, subsequently synthesizing instructions with enhanced diversity and difficulty through three synthesis paradigms. Furthermore, SELF-FOVEATE integrates a data regeneration module to improve the fidelity and quality of instructions to source text.

To summarize, the key contributions of this paper are as follows:

▷ We focus on unsupervised text-based instruction synthesis tasks, revealing limitations in existing works regarding diversity and difficulty.

▷ We propose SELF-FOVEATE, a method that synthesizes instructions with diversity and difficulty through LLMs based on unsupervised text.

▷ We conducted extensive experiments to evaluate SELF-FOVEATE, covering diversity and difficulty analysis, downstream task capabilities, and data scale trend analysis. The results demonstrate SELF-FOVEATE's superiority in unsupervised text-based instruction synthesis and suggest promising directions for future research.

2 Related Work

Instruction Tuning Multitask instruction fine-tuning (Wei et al., 2022) of language models significantly enhances their ability to follow instructions and generalize to new unseen tasks (Sanh et al., 2022; Mishra et al., 2022; Chung et al., 2022; Longpre et al., 2023; Zhou et al., 2023a; Li et al., 2024b). In our work, we utilize data from unsupervised text synthesis to conduct instruction tuning, enabling the model to better adapt to specific domain tasks.

Synthetic Data Generation LLMs have showcased remarkable capabilities in data synthesis (Long et al., 2024), facilitating the creation of extensive synthetic datasets for pretraining and fine-tuning, thereby progressively supplanting labor-intensive manual data scraping and selection (Liu et al., 2024), and mitigating the constraints that data imposes on model capability growth (Villalobos et al., 2024).

Distinct from earlier approaches centered on traditional language models (Schick and Schütze, 2021), LLMs present enhanced potential for generating high-quality synthetic data across diverse applications, including online translation (Oh et al., 2023), named entity recognition (Xiao et al., 2023), benchmark creation (Wang et al., 2024; Wei et al., 2024), and data diversity enhancement (Dai et al., 2023; Chung et al., 2023; Hong et al., 2024).

The concept of synthetic input-output pairs for instruction tuning advances by requiring that the data generated by LLMs be diverse, accurate, and difficult, often leveraging LLMs on a set of seed task demonstrations or user-provided unsupervised text to create new synthetic tasks (Wang et al., 2023a; Honovich et al., 2023; Zhang and Yang, 2023; Taori et al., 2023; Peng et al., 2023; Yuan et al., 2024; Li et al., 2024a).

Our work advances the generation of synthetic input-output pairs by developing a paradigm that integrates multi-level foveation into the creation of instruction tuning datasets, forming a comprehensive framework without any human annotation.

3 SELF-FOVEATE

In this section, we will introduce the proposed SELF-FOVEATE, denoted as $\mathcal{F}$, a multi-level automated method for synthesizing instructions that leverages only unsupervised text without the need for human-annotated samples. SELF-FOVEATE consists of three levels and a re-synthesis module. Formally, consider an original unsupervised text set $\mathcal{D}$, the proposed method SELF-FOVEATE operates by transforming each element d_i from $\mathcal{D}$ through a multi-level synthesis path $\mathcal{F}_j$. In practice, SELF-FOVEATE is an operation applied independently to each document d_i ($d_i \in \mathcal{D}$). The final generated dataset, $\mathcal{D}_{\text{gen}}$, is accumulated from the data subsets generated by all elements:

$$\mathcal{D}_{\text{gen}} = \mathcal{F}(\mathcal{D}) = \sum_{d_i \in \mathcal{D}} \sum_{\mathcal{F}_j \in \mathcal{F}} \mathcal{F}_j(d_i) \qquad (1)$$

During the synthesis process, the objectives of SELF-FOVEATE emphasize maximizing the diversity and difficulty of the generated instructions.

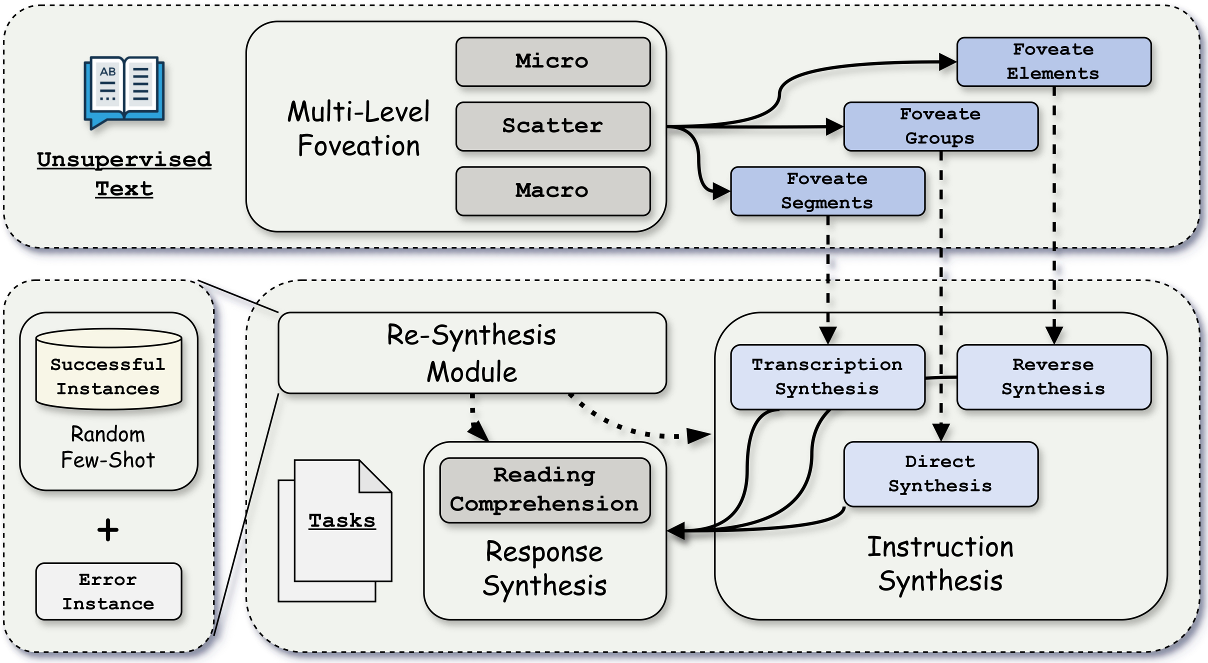


Figure 2: The SELF-FOVEATE workflow is designed for instruction synthesis based on unsupervised text. SELF-FOVEATE takes unsupervised text as input, extracts foveate elements, foveate groups, and foveate segments, then synthesizes instruction tuning data through these extracted details.

To ensure optimal functionality, SELF-FOVEATE incorporates three levels and one module as shown in Figure 2:

- **Micro-foveate Level:** This level captures the essential content within unsupervised text and acquires instructions through reverse synthesis. It plays a crucial role in maintaining the focus of synthesized instructions on the significant information in the text.
- **Scatter-foveate Level:** This level aims to combine key information scattered throughout the unsupervised text and uses direct synthesis to generate instructions. It stimulates LLMs to synthesize instructions that possess a profound understanding of the relationships between different contents in the text.
- **Macro-foveate Level:** This level focuses on and transcribes larger-grained information within the text. It plays a key role in enhancing the instructions' deep comprehension of the overall information in the text.
- **Re-synthesis Module:** This module is used to perform post-synthesis filtering on the generated instructions, excluding outliers in the instruction synthesis process and conducting reference-based re-synthesis for abnormally generated instructions to reduce information gaps in the instruction set and improve synthesis success rates.

3.1 Micro-foveate Level

General unsupervised text typically contains several primary entity types, a greater number of secondary entity types, and attributes of all these entity types. However, instruction sets directly synthesized by teacher LLMs may struggle to intentionally and comprehensively cover all the important content within such unsupervised text. This could lead to the synthesized instruction sets missing critical information from the original text, thereby resulting in weaker downstream model performance. To address this, we employ the micro-foveate mechanism and reverse synthesis to ensure that these key pieces of information are preserved in the synthesized instruction sets. Attention to a broader range of information enhances the diversity of the synthesized instruction sets, while the exploration of entity attributes also increases the difficulty.

Micro-foveate mechanism This mechanism aims to guide the teacher LLM to synthesize instructions from a fine-grained perspective of the unsupervised text. We introduce the concept of "*foveate elements*", which broadly encompass all entities and their attributes within a given text. We guide the large model to extract more foveate elements than needed and then select a certain number of foveate elements with the highest cosine similarity between the text and the embedding of each foveate element.

Reverse synthesis Based on the selected foveate elements, we employ the reverse synthesis method to generate instructions. Specifically, each foveate element is treated as a potential answer to an instruction, guiding the teacher LLM to synthesize instructions from the unsupervised text and then re-synthesize the answers. Even though the foveate elements can already serve as answers or parts of answers, to enhance the fluency, completeness, and accuracy of the answers, we choose to regenerate the answers in the reverse synthesis step rather than directly using the foveate elements as answers.

3.2 Scatter-foveate Level

Complete unsupervised texts often imply relationships between different entities or attributes within the text. However, instruction sets directly synthesized by teacher large language models (LLMs) may struggle to detect connections that require reasoning or deeper understanding in the text. This could result in the instruction sets synthesized by the teacher model missing deep-level information from the text, thereby affecting downstream model performance. To address this, we employ the scatter-foveate mechanism and direct synthesis method to ensure that these deep-level insights are reflected in the synthesized instruction sets. Attention to the deeper, less obvious implicit information enhances the diversity of the synthesized instruction sets, while insights into the relationships between entities or attributes increase the difficulty of the instructions.

Scatter-foveate mechanism This mechanism aims to guide the teacher LLM to synthesize instructions from a deeper perspective of the unsupervised text. We extract foveate elements more broadly from the text and randomly combine them into a certain number of *foveate groups* based on an empirical distribution.

Direct Synthesis Based on the formed foveate groups, we synthesize instructions using the direct synthesis method. The direct synthesis method treats each element in the foveate group as an indispensable part of the instruction to be synthesized, actively guiding the teacher LLM to consider the deep-level connections between different entities or attributes and solidifying such reasoning-based or deep-thinking connections in the synthesized instructions. After the instructions are synthesized, the answers are regenerated based on the text.

3.3 Macro-foveate Level

Unsupervised texts often contain content requiring focused attention, such as figurative devices and rhetorical exaggerations. Notably, teacher large language models (LLMs) may require additional prompting to specifically emphasize the critical information embedded within these writing techniques. To capture information more comprehensively from a global perspective, we employ the macro-foveate mechanism and transcription synthesis method to guide teacher LLMs in developing a profound understanding of unsupervised texts, ensuring effective extraction of key information conveyed through literary devices.

Macro-foveate mechanism This mechanism aims to guide teacher LLMs in comprehending unsupervised texts from a holistic perspective. We specifically highlight and extract text segments employing writing techniques – including metaphor, hyperbole, contrastive foil, rhetorical questioning, and citation – which are extracted as *foveate segments*.

Transcription synthesis Based on the identified writing technique segments, we employ transcription synthesis to convert each foveate segment into an instructional format. This process transforms declarative foveate segments into interrogative or imperative forms. Subsequently, corresponding answers are synthesized according to the content of the unsupervised text.

3.4 Re-synthesis Module

Due to limitations such as the capabilities of the teacher LLM, not all instructions in a directly synthesized instruction set may be answerable by the referenced unsupervised text. To ensure the synthesized instructions remain faithful to the referenced unsupervised text, we employ a re-synthesis module to repeatedly re-synthesize these failed instructions over multiple iterations, aiming to generate instructions that can be fully answered based on the referenced unsupervised text.

Single Sample Reference-synthesis During re-synthesis, we process only one failed instruction at a time and randomly sample a subset of successfully synthesized instructions as reference examples for the teacher LLM in each iteration. By providing the teacher LLM with different batches of reference samples during each re-synthesis attempt, we enhance the success rate of re-synthesis.

Hyperparameter Configuration The outputs of large language models (LLMs) are significantly influenced by hyperparameters such as temperature, top-p, and frequency penalty. To improve re-synthesis success rates, we meticulously calibrated these hyperparameters to define a high-creativity mode, enabling the teacher LLM to synthesize instructions more creatively based on reference samples.

4 Experiment

In this section, our experiments focus on three critical research questions: (Q1) How effective is Self-Foveate in enhancing instruction diversity? (Q2) How effective is Self-Foveate in enhancing instruction difficulty? (Q3) How effective is Self-Foveate in improving the model's problem-solving capabilities during instruction fine-tuning?

4.1 Experimental Setup

Datasets We employ three independent datasets, including the training set of the FilmWiki dataset containing 2,385 unsupervised texts with corresponding question-answer pairs. Additionally, we sample two widely-used benchmark QA datasets from the MRQA 2019 shared task (Fisch et al., 2019): SQuAD (Rajpurkar et al., 2016) (following Bonito (Nayak et al., 2024a)) and HotpotQA (Yang et al., 2018). To maintain comparable data scale and computational costs with the FilmWiki dataset, we extract 2,500 unsupervised texts with corresponding QA pairs from each dataset's training split as substitutes for the complete collections. Further implementation details are provided in Table 2.

Baselines We consider three key baselines: zero-shot, Self-QA, Bonito and Wiki2023. For the zero-shot baseline, we prompt models for evaluation without leveraging any unsupervised texts from the target task (**None**). The Self-QA baseline employs an unsupervised knowledge-guided method for extracting instruction-question-answer triples (**Self-QA**) (Zhang and Yang, 2023). The Bonito baseline utilizes a 7B-sized specialized model to generate various types of questions from unsupervised text. We configure it to produce questions of the "question answering without choices" type and obtain a sufficient number of samples through multiple sampling iterations (**Bonito**). The Wiki2023 baseline implements a text-based QA pair extraction methodology (**Wiki2023**). All baselines use identical unsupervised texts as our method, as specified in Section 4.1.

Dataset	Source		# Test Examples
	Question	Context	
SQuAD	Crowdsourced	Wikipedia	11639
HotpotQA	Crowdsourced	Wikipedia	2500
FilmWiki	LLM	Wikipedia	7398

Table 2: Statistics for the evaluation datasets from our experiments.

Datasets	Methods	Diversity Metrics	
		SelfBLEU Div.	Embedding Div.
SQuAD	Self-QA	0.593	0.838
	Bonito	0.494	0.838
	Wiki2023	0.550	0.842
	Self-Foveate	**0.665**	**0.851**
	Test Questions	0.695	0.840
HotpotQA	Self-QA	0.463	0.823
	Bonito	0.371	0.769
	Wiki2023	0.554	0.822
	Self-Foveate	**0.607**	**0.835**
	Test Questions	0.634	0.786
FilmWiki	Self-QA	0.406	0.687
	Bonito	0.197	0.677
	Wiki2023	0.341	0.664
	Self-Foveate	**0.563**	**0.706**
	Test Questions	0.316	0.618

Table 3: Comparison of diversity metrics across different methods and datasets. The table presents SelfBLEU Diversity (SelfBLEU Div.) and Embedding Diversity (Embedding Div.) scores for various methods on the datasets. The diversity of the test questions from each dataset is also provided as a reference.

Instruction Synthesis As described in Section 3, we process unsupervised texts through Self-Foveate using GPT-4o-mini and DeepSeek-V3 to generate instructional data. We implement two distinct hyperparameter configurations emphasizing high stability and high creativity, respectively, with detailed specifications in Appendix B.

Base Models We select three state-of-the-art open-source foundation models as our base models prior to instruction tuning: Meta-Llama-3.1-8B (Meta, 2024), Qwen2.5-7B (Team, 2024), and Gemma-2-9B. These decoder-only language models employ next-word prediction objectives and

Model	Settings	GPT-4o mini						DeepSeek-V3					
		SQuAD		HotpotQA		FilmWiki		SQuAD		HotpotQA		FilmWiki	
		Rec.	Acc.	Rec.	Acc.	Rec.	Acc.	Rec.	Acc.	Rec.	Acc.	Rec.	Acc.
Llama-3.1-8B	None*	0.309	0.202	0.244	0.160	0.212	0.082	0.309	0.202	0.244	0.160	0.212	0.082
	Self-QA	0.367	0.384	0.372	0.358	0.328	0.201	0.389	0.412	0.399	0.378	0.370	0.239
	Wiki2023	0.327	0.361	0.338	0.322	0.333	0.235	0.342	0.370	0.340	0.328	0.349	0.244
	Bonito*	0.386	0.405	0.360	0.372	0.219	0.153	0.386	0.405	0.360	0.372	0.219	0.153
	SELF-FOVEATE	**0.484**	**0.490**	**0.507**	**0.486**	**0.512**	**0.367**	**0.481**	**0.491**	**0.525**	**0.501**	**0.548**	**0.397**
Qwen2.5-7B	None*	0.251	0.300	0.266	0.234	0.139	0.032	0.251	0.300	0.266	0.234	0.139	0.032
	Self-QA	0.249	0.232	0.276	0.246	0.206	0.082	0.119	0.125	0.102	0.106	0.111	0.056
	Wiki2023	0.215	0.221	0.135	0.112	0.192	0.093	0.170	0.083	0.197	0.203	0.202	0.136
	Bonito*	0.143	0.109	0.212	0.199	0.168	0.098	0.143	0.109	0.212	0.199	0.168	0.098
	SELF-FOVEATE	**0.408**	**0.414**	**0.372**	**0.329**	**0.283**	**0.140**	**0.388**	**0.389**	**0.342**	**0.331**	**0.261**	**0.140**
Gemma-2-9B	None*	0.224	0.121	0.175	0.078	0.211	0.099	0.224	0.121	0.175	0.078	0.221	0.099
	Self-QA	0.383	0.409	0.408	0.389	0.429	0.315	0.402	0.435	0.424	0.408	0.509	0.386
	Wiki2023	0.336	0.378	0.361	0.352	0.478	0.384	0.364	0.399	0.373	0.365	0.494	0.401
	Bonito*	0.411	0.457	0.366	0.373	0.255	0.196	0.411	0.457	0.366	0.373	0.255	0.196
	SELF-FOVEATE	**0.507**	**0.525**	**0.537**	**0.520**	**0.672**	**0.528**	**0.499**	**0.514**	**0.552**	**0.525**	**0.697**	**0.581**

Table 4: Recall (Rec.) and LLM Accuracy (Acc.) on downstream tasks: SELF-FOVEATE vs. baselines. Results include models fine-tuned with instructions synthesized by GPT-4o mini or DeepSeek-V3, as well as reference non-instruction-tuned models (None). * Indicates that the base model was not fine-tuned using instructions synthesized by GPT-4o mini or DeepSeek-V3.

were pretrained on trillions of tokens without any instruction-based fine-tuning.

4.2 Diversity Discussion

To explicitly evaluate the diversity of the instructions in a metric-driven manner, we follow established practices recommended in Zhu et al. (2018); Perez et al. (2022); Tevet and Berant (2021) and employ two metrics: SelfBLEU score and Sentence-BERT embedding distances (Reimers and Gurevych, 2019a). These metrics capture different facets of diversity. SelfBLEU measures diversity in the form of text, while embedding distances measure diversity in the semantics of text. For SelfBLEU scores, we compute the average Self-BLEU scores using n-grams for $n \in \{2, 3, 4, 5\}$, following the approach suggested by Zhu et al. (2018). We use the implementation of the Self-BLEU metric in Alihosseini et al. (2019). Further details are available in Appendix C.

Table 3 demonstrates that SELF-FOVEATE-generated instructions achieve substantial diversity improvements in both textual and semantic dimensions, attaining or even surpassing the diversity level of crowdsourced test questions through a low-cost automated process.

4.3 Difficulty Discussion

To investigate the difficulty of the generated instructions, we designed prompts and employed GPT-4o to conduct a rigorous head-to-head comparison between the instructions generated by SELF-FOVEATE and those generated by baseline methods. Specifically, for each unsupervised text in each data subset, we used two contrasting methods to generate an equal number of instructions. We then provided GPT-4o with a set of instructions generated by both methods under the same unsupervised text, ensuring that the instruction sets were anonymous and their relative positions were randomized. Subsequently, we recorded GPT-4o's judgments and calculated the win rate for SELF-FOVEATE. The results of the head-to-head comparison, as shown in Table 5, demonstrate the outstanding difficulty of the instructions synthesized by SELF-FOVEATE. Further details can be found in Appendix D.

4.4 Problem-solving Capabilities

To evaluate the impact of instruction-tuning datasets on model performance in downstream tasks, we fine-tuned open-source models and assessed their performance on in-distribution problems. Table 4 illustrates the effects of instruction sets synthesized by two models (GPT-4o mini and DeepSeek-V3) during fine-tuning on open-source

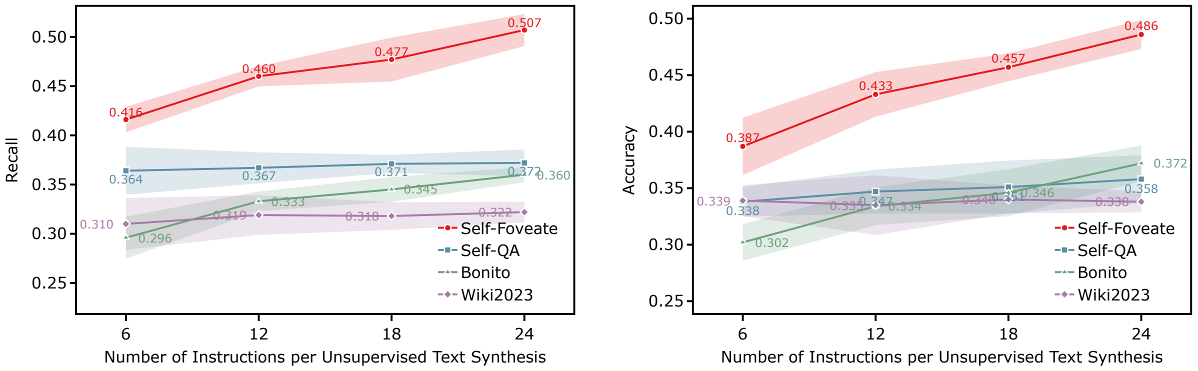


Figure 3: Impact of instruction set scale from SELF-FOVEATE and baselines on model fine-tuning performance.

model performance, with equal quantities of instructions per unsupervised text. As answers in test problems often consist of phrase-level or sentence-fragment extractions from source texts while large language models (LLMs) tend to generate more comprehensive responses, we selected both recall and LLM-evaluated accuracy as evaluation metrics. Here, higher recall indicates a greater probability of generated answers containing correct responses, while higher LLM-evaluated accuracy reflects stronger consistency between answers generated by DeepSeek-V3 and ground truth labels. To control potential confounding effects from the number of instructions synthesized per unsupervised text and examine the agreement between these two metrics, we investigated the influence of instruction set scale synthesized by SELF-FOVEATE and baseline methods on model fine-tuning performance using the HotpotQA dataset, as shown in Figure 3. Additional details regarding downstream task evaluations are provided in Appendix E.

Dataset	Baseline	SELF-FOVEATE WR.	
		Win	Lose
SQuAD	Self-QA	**70.64%**	29.36%
	Wiki2023	**80.83%**	19.17%
	Bonito	**99.96%**	00.04%
HotpotQA	Self-QA	**89.52%**	10.48%
	Wiki2023	**91.17%**	08.83%
	Bonito	**100.00%**	00.00%
FilmWiki	Self-QA	**85.12%**	14.88%
	Wiki2023	**95.08%**	04.92%
	Bonito	**96.31%**	03.69%

Table 5: Head-to-Head Comparison of Instruction Difficulty for SELF-FOVEATE Against Baselines Through Win Rates (WR.) Across Datasets.

Table 4 demonstrates that two state-of-the-art (SOTA) models achieve optimal performance across all downstream tasks in both metrics when using instructions synthesized via SELF-FOVEATE. Notably, during the fine-tuning process of the Qwen2.5-7B model, we observed that instruction data synthesized by baseline methods may lead to performance degradation compared to the pre-fine-tuned model, further demonstrating the advancement of the SELF-FOVEATE method in instruction synthesis. Our analysis presented in Figure 3 reveals that as the number of instructions synthesized per unsupervised text by SELF-FOVEATE increases, the performance of downstream fine-tuning tasks improves significantly, with an expanding performance gap compared to other baseline methods. Figure 3 also visually demonstrates the consistency between recall and LLM-evaluated accuracy, validating the appropriateness of these two metric selections.

4.5 Ablation Studies

To validate the effectiveness and necessity of each component in SELF-FOVEATE, we conduct comprehensive ablation studies focusing on two key aspects: (1) the contribution of each core component in the multi-level foveation framework, and (2) the impact of the answer regeneration mechanism in reverse synthesis.

4.5.1 Component-wise Analysis

We evaluate the necessity of each core component by systematically removing individual modules from the complete SELF-FOVEATE framework.

Table 6 presents the ablation results on the SQuAD dataset using Llama-3.1-8B as the base model. The results demonstrate that each component contributes significantly to the overall performance.

Setting	Recall	LLM Acc.
w/o Micro-Foveate	0.283	0.277
w/o Scatter-Foveate	0.274	0.260
w/o Macro-Foveate	0.344	0.339
Self-Foveate (Full)	**0.484**	**0.490**

Table 6: Ablation study results showing the contribution of each core component in Self-Foveate on the SQuAD dataset with Llama-3.1-8B.

The removal of any single component leads to performance degradation, validating the complementary necessity of all three core components.

4.5.2 Answer Regeneration Analysis

In our reverse synthesis paradigm, although foveate elements (e.g., "QuantumX processor") can directly serve as answers, we regenerate answers to enhance fluency, completeness, and semantic coherence. To validate this design choice, we compare the quality of answers with and without the regeneration mechanism.

Table 7 presents the comparative analysis on the SQuAD dataset, evaluating both fluency and completeness of generated answers through GPT-4o evaluation. The results demonstrate significant improvements in both aspects when employing answer regeneration.

Setting	High (%)	Medium (%)	Low (%)
	Fluency		
w/o Regeneration	64.9	19.5	15.6
Reverse Synthesis	**93.0**	**6.8**	**0.2**
	Completeness		
w/o Regeneration	36.7	27.5	35.7
Reverse Synthesis	**76.6**	**19.7**	**3.7**

Table 7: Comparison of answer quality with and without regeneration mechanism in reverse synthesis on the SQuAD dataset. Fluency and completeness are evaluated on a three-point scale (High/Medium/Low).

The results reveal that answer regeneration dramatically improves both fluency and completeness. This validates our methodological choice to maintain consistency across all synthesis paradigms by generating contextually appropriate and semantically coherent answers from the source text, rather than directly using extracted foveate elements.

5 Conclusion

In this paper, we proposed Self-Foveate, an unsupervised text-based instruction generation method powered by LLMs, which further explores key challenges in diversity and difficulty. Its innovative multi-level foveation and Re-synthesis Module ensure the acquisition of high-quality, text-faithful instructions from unsupervised text. Extensive experiments, including performance evaluations on downstream tasks, demonstrate the effectiveness of Self-Foveate. Our study has obtained many insightful findings, laying the foundation for future research in areas such as instruction difficulty assessment.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) via grant 62441614 and 62176078. We thank the anonymous reviewers for their insightful comments and suggestions.

Limitations

Although Self-Foveate provides significant improvements in generating diverse and difficulty-oriented fine-tuning instruction data based on unsupervised text data, several limitations must be acknowledged. First, the computational cost of processing large-scale unsupervised text data using closed-source SOTA large language models (LLMs) remains substantial. Ensuring data quality requires multi-step reasoning and iterative processing on unsupervised text data, which further escalates computational demands. Second, although Self-Foveate aims to fully synthesize data from unsupervised text and incorporates verification mechanisms, the inherent tendency of LLMs to hallucinate or generate erroneous information persists as a challenge. While precise prompting strategies and higher-quality unsupervised text data can mitigate these inaccuracies, they cannot be entirely eliminated.

Ethical Consideration

We state that any research or application arising from this study is strictly authorized solely for research purposes. In our work, any unsupervised text datasets used are from public sources and do not contain any private information. In this paper, we have fully presented the prompts used by

SELF-FOVEATE in the Appendix. All synthesized instructions rely on the provided unsupervised text and are inspected by relevant modules. Therefore, our method strives to minimize potential safety and ethical risks as much as possible. However, during the process of synthesizing fine-tuning instruction data, maliciously provided unsupervised text data can lead the model to produce harmful or inappropriate outputs, which is a shared problem. Additionally, potential unfairness and discrimination present in the unsupervised text data might be amplified by LLMs during the instruction generation process. Ensuring the quality of generated fine-tuning instruction data in a safe and highly controllable manner is crucial. The application of these techniques should be guided by ethical considerations, with safeguards in place to prevent misuse and reduce the likelihood of producing harmful outcomes.

References

Danial Alihosseini, Ehsan Montahaei, and Mahdieh Soleymani Baghshah. 2019. Jointly measuring diversity and quality in text generation models. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 90–98, Minneapolis, Minnesota. Association for Computational Linguistics.

Anthropic. 2024. Claude. `https://www.anthropic.com/claude`.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *Preprint*, arXiv:2210.11416.

John Chung, Ece Kamar, and Saleema Amershi. 2023. Increasing diversity while maintaining accuracy: Text data generation with large language models and human interventions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 575–593, Toronto, Canada. Association for Computational Linguistics.

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*.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, Hongmin Cai, Lichao Sun, Quanzheng Li, Dinggang Shen, Tianming Liu, and Xiang Li. 2023. Auggpt: Leveraging chatgpt for text data augmentation. *Preprint*, arXiv:2302.13007.

Michael Han Daniel Han and Unsloth team. 2023. Unsloth.

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*.

DeepSeek-AI. 2024. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proc. of NAACL*.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

Yuan Ge, Yilun Liu, Chi Hu, Weibin Meng, Shimin Tao, Xiaofeng Zhao, Hongxia Ma, Li Zhang, Boxing Chen, Hao Yang, Bei Li, Tong Xiao, and Jingbo Zhu. 2024. Clustering and ranking: Diversity-preserved instruction selection through expert-aligned quality estimation. *Preprint*, arXiv:2402.18191.

Hongyi Guo, Yuanshun Yao, Wei Shen, Jiaheng Wei, Xiaoying Zhang, Zhaoran Wang, and Yang Liu. 2024. Human-instruction-free llm self-alignment with limited samples. *Preprint*, arXiv:2401.06785.

Zhang-Wei Hong, Idan Shenfeld, Tsun-Hsuan Wang, Yung-Sung Chuang, Aldo Pareja, James Glass, Akash Srivastava, and Pulkit Agrawal. 2024. Curiosity-driven red-teaming for large language models. *Preprint*, arXiv:2402.19464.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14409–14428, Toronto, Canada. Association for Computational Linguistics.

Zhengbao Jiang, Zhiqing Sun, Weijia Shi, Pedro Rodriguez, Chunting Zhou, Graham Neubig, Xi Lin, Wen-tau Yih, and Srini Iyer. 2024. Instruction-tuned language models are better knowledge learners. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5421–5434, Bangkok, Thailand. Association for Computational Linguistics.

Abdullatif Köksal, Timo Schick, Anna Korhonen, and Hinrich Schütze. 2024. Longform: Effective instruction tuning with reverse instructions. *Preprint*, arXiv:2304.08460.

Haoran Li, Qingxiu Dong, Zhengyang Tang, Chaojun Wang, Xingxing Zhang, Haoyang Huang, Shaohan Huang, Xiaolong Huang, Zeqiang Huang, Dongdong Zhang, Yuxian Gu, Xin Cheng, Xun Wang, Si-Qing Chen, Li Dong, Wei Lu, Zhifang Sui, Benyou Wang, Wai Lam, and Furu Wei. 2024a. Synthetic data (almost) from scratch: Generalized instruction tuning for language models. *Preprint*, arXiv:2402.13064.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer, Jason Weston, and Mike Lewis. 2024b. Self-alignment with instruction backtranslation. *Preprint*, arXiv:2308.06259.

Chen Ling, Xujiang Zhao, Jiaying Lu, Chengyuan Deng, Can Zheng, Junxiang Wang, Tanmoy Chowdhury, Yun Li, Hejie Cui, Xuchao Zhang, Tianjiao Zhao, Amit Panalkar, Dhagash Mehta, Stefano Pasquali, Wei Cheng, Haoyu Wang, Yanchi Liu, Zhengzhang Chen, Haifeng Chen, Chris White, Quanquan Gu, Jian Pei, Carl Yang, and Liang Zhao. 2024. Domain specialization as the key to make large language models disruptive: A comprehensive survey. *Preprint*, arXiv:2305.18703.

Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, and Andrew M. Dai. 2024. Best practices and lessons learned on synthetic data. *Preprint*, arXiv:2404.07503.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. *Preprint*, arXiv:2406.15126.

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. *Preprint*, arXiv:2301.13688.

Meta. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3470–3487, Dublin, Ireland. Association for Computational Linguistics.

Nihal Nayak, Yiyang Nan, Avi Trost, and Stephen Bach. 2024a. Learning to generate instruction tuning datasets for zero-shot task adaptation. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 12585–12611, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Nihal V. Nayak, Yiyang Nan, Avi Trost, and Stephen H. Bach. 2024b. Learning to generate instruction tuning datasets for zero-shot task adaptation. *Preprint*, arXiv:2402.18334.

Seokjin Oh, Su Ah Lee, and Woohwan Jung. 2023. Data augmentation for neural machine translation using generative language model. *Preprint*, arXiv:2307.16833.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

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

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. 2022. Red teaming language models with language models. *Preprint*, arXiv:2202.03286.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019a. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019b. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Kuniaki Saito, Kihyuk Sohn, Chen-Yu Lee, and Yoshitaka Ushiku. 2024. Where is the answer? investigating positional bias in language model knowledge extraction. *Preprint*, arXiv:2402.12170.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin

Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization. *Preprint*, arXiv:2110.08207.

Timo Schick and Hinrich Schütze. 2021. Generating datasets with pretrained language models. *Preprint*, arXiv:2104.07540.

Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan. 2023. Principle-driven self-alignment of language models from scratch with minimal human supervision. *Preprint*, arXiv:2305.03047.

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. `https://github.com/tatsu-lab/stanford_alpaca`.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Guy Tevet and Jonathan Berant. 2021. Evaluating the evaluation of diversity in natural language generation. *Preprint*, arXiv:2004.02990.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2024. Will we run out of data? limits of llm scaling based on human-generated data. *Preprint*, arXiv:2211.04325.

Siyuan Wang, Zhuohan Long, Zhihao Fan, Zhongyu Wei, and Xuanjing Huang. 2024. Benchmark self-evolving: A multi-agent framework for dynamic llm evaluation. *Preprint*, arXiv:2402.11443.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023a. 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.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023b. Aligning large language models with human: A survey. *Preprint*, arXiv:2307.12966.

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. *Preprint*, arXiv:2109.01652.

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Hu, Jie Huang, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, and Quoc V. Le. 2024. Long-form factuality in large language models. *Preprint*, arXiv:2403.18802.

Siyuan Wu, Yue Huang, Chujie Gao, Dongping Chen, Qihui Zhang, Yao Wan, Tianyi Zhou, Xiangliang Zhang, Jianfeng Gao, Chaowei Xiao, and Lichao Sun. 2024. Unigen: A unified framework for textual dataset generation using large language models. *Preprint*, arXiv:2406.18966.

Ruixuan Xiao, Yiwen Dong, Junbo Zhao, Runze Wu, Minmin Lin, Gang Chen, and Haobo Wang. 2023. FreeAL: Towards human-free active learning in the era of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14520–14535, Singapore. Association for Computational Linguistics.

Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. 2023. Baize: An open-source chat model with parameter-efficient tuning on self-chat data. *Preprint*, arXiv:2304.01196.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *Preprint*, arXiv:2401.10020.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2024. Instruction tuning for large language models: A survey. *Preprint*, arXiv:2308.10792.

Xuanyu Zhang and Qing Yang. 2023. Self-qa: Unsupervised knowledge guided language model alignment. *Preprint*, arXiv:2305.11952.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023a. Lima: Less is more for alignment. *Preprint*, arXiv:2305.11206.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023b. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. *Preprint*, arXiv:1802.01886.

A Datasets

A.1 Dataset Details

This section provides comprehensive details regarding the datasets utilized in this study.

SQuAD(Rajpurkar et al., 2016) This Dataset serves as a benchmark for extractive question answering, containing over 100,000 human-generated question-answer pairs anchored to Wikipedia passages. In our work, we utilize this dataset from the MRQA 2019 shared task(Fisch et al., 2019). Considering computational costs, we employ 2,500 randomly selected articles or contexts from the training set.

HotpotQA(Yang et al., 2018) The original purpose of proposing this dataset was to challenge models with multi-hop reasoning across 113k Wikipedia-based QA pairs requiring synthesis of information from multiple documents. In our work, we utilize this dataset from the MRQA 2019 shared task(Fisch et al., 2019). Considering computational costs, we employ 2,500 randomly selected articles or contexts from the training set.

FilmWiki(Saito et al., 2024) This dataset was initially constructed to investigate a phenomenon called the perplexity curse, with all texts sourced from Wikipedia. Our study employs its film-themed subset containing the complete collection of 2,385 unsupervised texts and their corresponding questions.

A.2 Usage of Datasets

During the synthesis process, we use the articles or contexts from the dataset (excluding the questions) and generate instructions through the SELF-FOVEATE. In the evaluation of problem-solving capabilities, we assess the fine-tuned models using the corresponding questions in the dataset.

B Instruction Synthesis

Taking into account both performance and API costs, and to eliminate potential biases in experimental results caused by specific teacher models (such as prompt hacking), we employ two models, GPT-4o mini(gpt-4o-mini-2024-07-18)(OpenAI, 2024) and DeepSeek-V3(DeepSeek-AI, 2024), to generate instructions from unsupervised text using the same methodology and sampling hyperparameters. Detailed hyperparameters are provided in Table 8, where, as described in Section 3, the normal mode is utilized for the initial foveate synthesis, and the high-creativity mode is adopted for re-synthesis.

Hyperparameters	Values
Normal Mode	
frequency_penalty	0.5
max_completion_tokens	None
presence_penalty	0
temperature	0.5
top_p	1.0
High-creativity Mode	
frequency_penalty	0.5
max_completion_tokens	None
presence_penalty	0
temperature	1.2
top_p	1.0

Table 8: Hyperparameters of Synthetic Task Generation.

C Diversity Experiment

Mathematically, we define the diversity metrics as follows ('D' denotes 'Diversity', 'SB' denotes 'SelfBLEU', and 'EB' denotes 'Embedding'):

$$\mathrm{D}_{\mathrm{SB}} = 1 - \frac{1}{|\mathcal{X}_\tau|} \sum_{x_i \in \mathcal{X}_\tau} \sum_{n=2}^{5} \mathrm{SelfBLEU}_{\mathcal{X}_\tau}(x_i, n) \tag{2}$$

$$\mathrm{D}_{\mathrm{EB}} = 1 - \frac{1}{2|\mathcal{X}_\tau|} \sum_{x_i \in \mathcal{X}_\tau} \sum_{x_j \in \mathcal{X}_\tau} \frac{\phi(x_i) \cdot \phi(x_j)}{\|\phi(x_i)\|^2 \|\phi(x_j)\|^2} \tag{3}$$

For computing sentence embeddings, we employ sentence-transformers/paraphrase-multilingual-mpnet-base-v2(Reimers and Gurevych, 2019b), which maps sentences to a 768-dimensional dense vector space. Compared to other sentence embedding transformation models, it offers a balanced mapping dimension and accuracy, enabling the computation of cosine similarity matrices at an acceptable computational cost while obtaining relatively accurate diversity evaluation metrics.

D Difficulty Head-to-Head Experiment

We prompt GPT-4o (gpt-4o-2024-08-06)(OpenAI, 2024) to determine which method-synthesized instruction set is more challenging. The prompts are shown below.

Difficulty Head-to-Head Experiment System Prompt

Please compare two sets of instructions/questions for a given content to determine which set is more difficult for a language model to generate coherent and accurate responses. A set of instructions is considered more difficult if it requires:
1. More complex reasoning or multi-step analysis
2. Integration of multiple concepts or contexts
3. Explanation of nuanced relationships or processes
4. Handling of ambiguous or less-documented information
5. Synthesis of information from diverse sources or time periods

Please respond with '1' if the first set of instructions is more difficult, or '2' if the second set of instructions is more difficult.
Please respond with the number only, no other text or characters.

Difficulty Head-to-Head Experiment User Prompt

Content: {unsupervised text}
Instruction Set 1: {instruction set a}
Instruction Set 2: {instruction set b}

The hyperparameters set during sampling are listed in Table 9.

Hyperparameters	Values
frequency_penalty	0.0
max_completion_tokens	5
presence_penalty	0
temperature	0.5
top_p	1.0

Table 9: Hyperparameters of Difficulty Head-to-Head Experiment.

E Downstream Experiment

E.1 Software and Hardware Details

The implementation leverages the LLaMA-Factory framework (Zheng et al., 2024) with computational optimizations from FlashAttention2 (Dao et al., 2022; Dao, 2024) and Unsloth libraries (Daniel Han and team, 2023). For training, we employ NVIDIA A100, A800, and H800 GPUs based on availability within our computational cluster. The entire work required approximately 110 GPU hours.

E.2 Hyperparameters

The hyperparameters and technical configurations for the instruction tuning process are documented in Table 10.

Hyperparameters	**Values**
cutoff_len	2048
learning_rate	0.0001
num_train_epochs	5.0
effective_batch_size	16
lr_scheduler_type	cosine
max_grad_norm	1.0
warmup_steps	0
optim	adamw_torch
quantization_bit	4
quantization_method	bitsandbytes
lora_rank	8
lora_alpha	16
lora_dropout	0

Table 10: Hyperparameters of Instruction Tuning with Q-LoRA Quantization.

E.3 LLM Accuary Evaluation

LLM Accurary Evaluation System Prompt

> You are a fair judge. Your task is to determine if the generated answer correctly answers the question, even if it contains additional explanations. Rules:
> 1. The generated answer is correct if it contains the key information from the ground truth
> 2. Additional explanations or context in the generated answer should not make it incorrect
> 3. Only respond with 'Correct' or 'Incorrect'

LLM Accurary Evaluation User Prompt

> Compare the following answers:
> Question: {question}
> Ground Truth Answer: {ground_truth}
> Generated Answer: {generated}
>
> Is the generated answer correct, regardless of any additional explanation? Respond only with 'Correct' or 'Incorrect'.

F Answer Regeneration Evaluation

This section provides detailed documentation of the evaluation methodology used in the ablation study for answer regeneration analysis (Section 4.5). To assess the quality of answers generated with and without the regeneration mechanism, we employed GPT-4o (gpt-4o-2024-08-06) to evaluate fluency and completeness on a three-point scale.

F.1 Evaluation Prompt

The following system prompt was used to evaluate question-answer pairs:

Answer Regeneration Evaluation System Prompt

> Please evaluate the given question and answer pair based on two criteria:
>
> 1. Fluency: How well does the answer flow and connect with the question?
> - High: The answer naturally follows from the question
> - Medium: The connection is somewhat clear but could be improved
> - Low: The answer feels disconnected from the question
>
> 2. Completeness: How thoroughly does the answer address the question?
> - High: The answer fully addresses all aspects of the question
> - Medium: The answer covers most aspects but misses some points
> - Low: The answer only partially addresses the question
>
> Please respond with a JSON object in the following format:
> {
> "fluency": "high|medium|low",
> "completeness": "high|medium|low"
> }
>
> Do not include any other text or explanation.

F.2 Evaluation Hyperparameters

The hyperparameters used for the answer regeneration evaluation are provided in Table 11.

Hyperparameters	**Values**
frequency_penalty	0.0
max_completion_tokens	50
presence_penalty	0
temperature	0.2
top_p	1.0

Table 11: Hyperparameters for Answer Regeneration Evaluation.

G Use of AI Assistants

In this study, we utilized AI-powered tools, including ChatGPT and Grammarly, to enhance the linguistic accuracy of our manuscript through spell-checking and minor grammatical corrections. Additionally, the codebase was developed using Cursor to improve coding efficiency and accuracy.