RefGen: Enhancing LLM Agents via Reinforced Reference Selection

Jiechao Gao[1,*], **Wei Zhu**[2,*], **Liz Li**[3]
[1]Stanford University, Stanford, CA, United States
[2]University of Hong Kong, Hong Kong, HK, China
[3]DataSelect AI, Shanghai, China

Abstract

Reference-augmented inference has emerged as an effective form of test-time scaling for large language models (LLMs), where selected referred demonstrations or referred cases help adapt reasoning to a given query. However, existing reference selection methods mainly rely on heuristic retrieval, independent scoring, or expensive LLM-based reranking, and therefore do not explicitly model how multiple references should be jointly composed and ordered under a limited context budget. We propose **RefGen**, a lightweight and modular framework that formulates reference selection as an autoregressive index generation problem over a retrieved candidate pool. Instead of generating new textual demonstrations, RefGen uses a compact Transformer encoder–decoder to produce an ordered sequence of candidate indices, enabling query-aware composition of referred demonstrations for downstream reasoning. To optimize this discrete selection policy, we combine supervised fine-tuning with reinforcement learning using verifiable rewards. Experiments on mathematical reasoning, scientific question answering, and visual question answering benchmarks show that RefGen consistently outperforms retrieval-based and learning-based baselines across both LLM and VLM backbones. RefGen is plug-and-play for frozen foundation models and introduces only minimal inference overhead, making it practical for real-world deployment.

1 Introduction

Recent advances in large language models (LLMs) and vision–language models (VLMs) have shown that increasing *test-time compute* can substantially improve reasoning performance without updating model parameters (Wei et al., 2022; Jaech et al., 2024; Guo et al., 2025). Beyond generating longer chains of thought or performing iterative self-correction, an increasingly important paradigm is to augment inference with *references*, such as previously solved cases, task-relevant exemplars, or external knowledge sources. When suitable references are provided, foundation models can better adapt their reasoning process to the current query, reuse useful solution patterns, and perform stronger few-shot inference at test time. This capability is particularly valuable for challenging reasoning tasks, where the quality of contextual examples often has a direct impact on final prediction accuracy.

Despite this promise, existing reference selection strategies remain limited. Classical sparse and dense retrieval methods (Chen et al., 2017; Karpukhin et al., 2020) typically score candidate references independently based on query similarity, which often produces redundant or weakly complementary referred demonstrations under a restricted context budget. More recent learning-based approaches (Zhou et al., 2025) can improve selection quality, but many of them still rely on one-shot scoring or expensive LLM-based reranking, and therefore do not explicitly model how multiple referred demonstrations should be *jointly composed* into an effective ordered context. In practice, high-quality references are rarely just a set of individually relevant examples. Instead, they are better viewed as a structured sequence of referred cases whose coverage, diversity, and ordering interact with one another and jointly affect downstream reasoning. Failing to model this compositional structure limits the effectiveness and efficiency of existing approaches.

To address these challenges, we propose *RefGen*, a lightweight and modular framework for adaptive reference composition. RefGen formulates reference selection as an *autoregressive index generation* problem over a retrieved candidate pool. Given an input query and a small set of candidate referred demonstrations, a compact Transformer encoder–decoder directly generates an ordered sequence of

* Corresponding author. For any inquiries, please contact: michaelwzhu91@gmail.com, jiechao@stanford.edu.

reference indices, rather than generating textual demonstrations themselves. This design enables RefGen to explicitly capture sequential dependencies among selected references while cleanly decoupling reference composition from the text generation process of the underlying LLM or VLM. As a result, RefGen can be attached to frozen backbone models in a plug-and-play manner, introduces only limited additional inference cost, and naturally extends to both language-only and multimodal settings. To learn this discrete and structured selection policy, we combine supervised fine-tuning with reinforcement learning using verifiable rewards.

We evaluate RefGen on a diverse set of benchmarks covering mathematical reasoning, scientific question answering, and visual question answering. Across multiple LLM and VLM backbones, RefGen consistently outperforms retrieval-based and learning-based baselines, with especially clear gains on more difficult tasks that require stronger compositional reasoning. In addition to accuracy improvements, RefGen preserves the practical advantages required by real-world deployment: it does not require modification of proprietary foundation models, can be integrated as an external module, and adds only modest computational overhead. These results suggest that learning to generate ordered references, rather than independently retrieving or reranking them, provides an effective and scalable route for enhancing test-time reasoning in modern LLM and VLM agents.

2 Related works

In-context learning and reference selection Reference-based test-time adaptation is closely connected to the literature on in-context learning (ICL), where model performance is highly sensitive to the choice and arrangement of demonstrations. A major line of work studies retrieval-based demonstration selection, typically by encoding the input query and candidate examples into a shared embedding space and then retrieving the most relevant items using similarity search. Representative approaches include embedding-based configuration methods (Li et al., 2024), EPR (Rubin et al., 2021), and UDR (Li et al., 2023), which improve retrieval quality by training the retriever with task-aware objectives. Other methods further refine the retrieval signal by aligning demonstrations with target responses (Hu et al., 2022), or by introducing learned selectors on top of a candidate pool. Lever-LM (Yang et al., 2024), for example, employs a smaller language model to select demonstrations, while VICL (Zhou et al., 2024) adopts a retrieve-and-rerank pipeline and uses a large model for reranking in multimodal settings. SURf (Sun et al., 2024) improves in-context demonstration construction through self-refinement, and (Qin et al., 2024) iteratively builds diverse demonstration sets for test predictions. Despite their effectiveness, most existing methods still rely on independent relevance scoring, heuristic reranking, or direct language-model decoding over textual demonstrations. As a result, they do not explicitly model the sequential and combinatorial dependencies among multiple referred demonstrations under a limited context budget. In contrast, RefGen formulates reference composition as autoregressive index generation over a retrieved candidate pool, allowing the model to learn how an ordered sequence of referred demonstrations should be jointly composed for downstream reasoning, without generating additional textual demonstrations.

Reinforcement learning for reasoning and discrete decision optimization Reinforcement learning has recently become a central technique for improving reasoning behavior in foundation models, especially in settings where the quality of a generated trajectory can be evaluated through verifiable outcomes (Jaech et al., 2024; Guo et al., 2025). In particular, Group Relative Policy Optimization (GRPO) (Shao et al., 2024) has emerged as an influential paradigm for reinforcement learning with verifiable rewards, replacing a learned value function with group-wise relative advantage estimation and thereby improving optimization stability for difficult reasoning tasks. This line of research has motivated a series of open efforts, including Open-Reasoner-Zero (Hu et al., 2025), Reinforce++ (Hu, 2025), VinePPO (Kazemnejad et al., 2024), RLEF (Gehring et al., 2024), and DAPO (Yu et al., 2025), which investigate more stable policy optimization, better credit assignment, and more effective use of execution or outcome feedback. Although these works mainly target long-form reasoning, code generation, or mathematical problem solving, they provide an important foundation for our setting, where the action space is also discrete and the final utility of a selected reference sequence is naturally measured by downstream task performance. Different from prior RL studies that optimize textual reasoning trajectories, RefGen applies reinforcement learning with verifi-

able rewards to the structured problem of reference index generation, enabling direct optimization of ordered referred-demonstration selection according to end-task outcomes.

3 RefGen Framework

We now describe the proposed **RefGen** framework in detail, including the construction of the reference bank, the autoregressive reference generation architecture, and the training strategy used to optimize the reference selection policy.

3.1 Formulation of reference bank

Given a frozen LLM/VLM agent, denoted by $\mathcal{M}$ (e.g., DeepSeek-R1 (Wu et al., 2024)), our goal is to select a query-dependent ordered list of references that can improve downstream inference. We maintain a *reference bank* $\mathcal{R} = \{r_1, \ldots, r_N\}$, where each reference r_j is a previously observed referred case or a referred demonstration that can be inserted into the context of the backbone model. Depending on the task, a reference may contain text only or multimodal content. For example, in visual question answering, a referred case can be written as $r_j = (I_j, T_j, A_j)$, where I_j is the image, T_j is the question or instruction, and A_j is the corresponding answer. For a test input x, RefGen first retrieves a candidate pool $\mathcal{C}(x) = \{c_1, \ldots, c_n\} \subseteq \mathcal{R}$ using an embedding model such as BGE (Xiao et al., 2024) for text or CLIP (Radford et al., 2021) for multimodal inputs, together with a vector search engine such as Faiss (Douze et al., 2024). RefGen then generates an ordered index sequence $\mathbf{i} = (i_1, \ldots, i_k)$ with $i_t \in \{1, \ldots, n\}$, which specifies the final referred demonstrations selected from $\mathcal{C}(x)$. The resulting reference sequence $\mathcal{Z}(x) = [c_{i_1}, \ldots, c_{i_k}]$ is serialized and concatenated with the test input before being passed to the backbone model, yielding the final prediction

$$\hat{y} = \mathcal{M}(\mathcal{Z}(x), x). \tag{1}$$

This formulation explicitly separates coarse candidate retrieval from fine-grained reference composition: retrieval ensures efficiency over a large reference bank, while RefGen learns how to compose a compact and ordered set of referred demonstrations tailored to the current query.

3.2 RefGen

We now present the design of RefGen, as illustrated in Figure **??**. RefGen operates as a lightweight external module on top of a frozen backbone model. Given an input query x, it first retrieves a candidate pool $\mathcal{C}(x) = \{c_1, \ldots, c_n\}$ from the reference bank $\mathcal{R}$. It then predicts an ordered sequence of k indices over the candidate pool, corresponding to the referred demonstrations that will be inserted into the final prompt. In contrast to conventional retrieval pipelines that rank candidates independently or select the top-k items in a single step, RefGen models reference composition as a sequential decision process, allowing the choice at step t to depend on both the query and the previously selected references.

Model architecture RefGen contains two components: a frozen embedder $E(\cdot)$ and a trainable Transformer encoder–decoder $G(\cdot)$. The embedder maps the input query and each candidate reference into a shared fixed-dimensional representation space. For language-only tasks, $E(\cdot)$ can be instantiated by BGE (Xiao et al., 2024); for multimodal tasks, we use CLIP (Radford et al., 2021). Concretely, for a candidate referred case $c_j = (I_j, T_j, A_j)$, we encode the visual content and the concatenated textual fields $[T_j; A_j]$ and obtain its representation $h_j = E(I_j, [T_j; A_j])$. For the query $x = (I_x, T_x)$, the corresponding representation is $h_x = E(I_x, T_x)$. Candidate embeddings can be precomputed when building the vector index, which keeps online inference efficient. The trainable generator $G(\cdot)$ takes as input the sequence $[h_x; h_1; \ldots; h_n]$, where $[\cdot;\cdot]$ denotes concatenation along the sequence dimension. Its encoder produces contextualized representations over the query and all retrieved candidates, while its decoder autoregressively predicts a sequence of reference indices

$$\mathbf{i} = (i_1, \ldots, i_k), \qquad i_t \in \{1, \ldots, n\}. \tag{2}$$

Importantly, the decoder outputs indices rather than natural-language tokens. This design makes RefGen substantially lighter than methods that generate new demonstrations in free-form text, while still enabling it to model interdependence, complementarity, and ordering among referred demonstrations.

Inference procedure At inference time, RefGen first performs dense retrieval to obtain the candidate pool $\mathcal{C}(x)$. The encoder–decoder then generates the index sequence autoregressively according to

$$P(\mathbf{i} \mid x, \mathcal{C}(x)) = \prod_{t=1}^{k} P(i_t \mid i_{<t}, x, \mathcal{C}(x); \theta), \tag{3}$$

where θ denotes the parameters of RefGen. The ordered references selected by $\mathbf{i}$ are then assembled

into the final context in the same order as generated. This ordering is important in practice, since large models are known to be sensitive not only to which demonstrations are provided, but also to how they are arranged in context. By generating indices sequentially, RefGen can learn to avoid redundancy, promote coverage across different reasoning patterns, and construct a more effective few-shot context under a limited budget.

3.3 Training methods for RefGen

Since RefGen produces an autoregressive sequence of discrete indices, it can be optimized using training strategies similar to those used for sequence models. Following recent advances in LLM post-training and reinforcement learning with verifiable rewards (Guo et al., 2025; Ouyang et al., 2022), we consider supervised fine-tuning (SFT), reinforcement learning with verifiable rewards (RLVR), and a staged pipeline that first performs SFT and then applies RLVR.

SFT In the SFT stage, RefGen is trained with teacher forcing to maximize the likelihood of an oracle index sequence $\mathbf{i}^* = (i_1^*, \ldots, i_k^*)$. The objective is the standard negative log-likelihood:

$$\mathcal{L}_{\text{SFT}} = -\sum_{t=1}^{k} \log P(i_t^* \mid i_{<t}^*, x, \mathcal{C}(x); \theta)\,. \quad (4)$$

This stage teaches the model the output format and provides a strong initialization for reference composition. In particular, it allows RefGen to learn basic patterns of useful referred-demonstration ordering before moving to direct end-task optimization.

RLVR We further optimize RefGen with reinforcement learning using verifiable downstream rewards. We adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which has recently shown strong empirical performance in reasoning-oriented policy optimization (Jaech et al., 2024; Guo et al., 2025). For a given query, the behavior policy $\pi_{\theta_{\text{old}}}$ samples G candidate index sequences $\{\mathbf{i}^{(g)}\}_{g=1}^{G}$. Each sequence determines an ordered set of referred demonstrations, which is then inserted into the prompt of the frozen backbone model $\mathcal{M}$ to produce an answer $\hat{y}^{(g)}$. The reward is computed from the final task outcome, and the group-normalized advantage for the g-th rollout is

$$\hat{A}^{(g)} = \frac{R^{(g)} - \text{mean}(\{R^{(j)}\}_{j=1}^{G})}{\text{std}(\{R^{(j)}\}_{j=1}^{G})}. \quad (5)$$

GRPO optimizes a clipped policy objective with KL regularization:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\, \{\mathbf{i}^{(g)}\}_{g=1}^{G}\sim\pi_{\theta_{\text{old}}}} \left[\frac{\text{Obj}}{G} - \text{KL}\right], \quad (6)$$

where

$$\text{Obj} = \sum_{g=1}^{G} \frac{1}{|\mathbf{i}^{(g)}|} \sum_{t=1}^{|\mathbf{i}^{(g)}|} \min\Big(r_{g,t}(\theta)\hat{A}^{(g)}, \text{clip}(r_{g,t}(\theta), \varepsilon)\hat{A}^{(g)}\Big), \quad (7)$$

$$\text{KL} = \beta D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{ref}}), \quad (8)$$

and

$$r_{g,t}(\theta) = \frac{\pi_\theta(i_t^{(g)} \mid x, \mathcal{C}(x), i_{<t}^{(g)})}{\pi_{\theta_{\text{old}}}(i_t^{(g)} \mid x, \mathcal{C}(x), i_{<t}^{(g)})}. \quad (9)$$

Compared with pure supervised learning, RLVR directly optimizes the usefulness of the generated reference sequence with respect to the final task result, which is particularly important because the quality of a referred-demonstration set depends on its joint effect on downstream reasoning rather than on local token-level matching.

Note that the outcome reward is computed from the final prediction of the frozen backbone model:

$$R(\hat{y}, y) = \text{Score}(\hat{y}, y), \quad (10)$$

where y is the gold answer and $\hat{y}$ is the model prediction obtained after conditioning on the references selected by RefGen. For mathematical reasoning and question answering tasks, $\text{Score}(\cdot,\cdot)$ is the instance-level accuracy or exact-match score. In addition, we incorporate a simple structural reward to encourage valid outputs from the index generator:

$$R_{\text{format}}(\mathbf{i}) = \text{CorrectFormat}(\mathbf{i}), \quad (11)$$

where $\text{CorrectFormat}(\cdot)$ returns 1 if the generated sequence is valid—for example, all indices fall within the candidate range and do not contain illegal repetitions—and 0 otherwise. This auxiliary signal improves training stability by encouraging RefGen to produce well-formed ordered reference sequences throughout optimization.

4 Experiments

4.1 Datasets

Vision-language tasks We evaluate RefGen on three visual question answering benchmarks:

VQAv2 (Goyal et al., 2017), VizWiz (Gurari et al., 2018), and OK-VQA (Marino et al., 2019).

Mathematical reasoning tasks We consider three math benchmarks: Math-500 (Lightman et al., 2023), AIME-24 (Zhang and Math-AI, 2024), and AIME-25 (Zhang and Math-AI, 2025).

Science question answering We further evaluate on two challenging science QA benchmarks: MMLU-Pro (Wang et al., 2024) and GPQA (Rein et al., 2024).

4.2 Reference construction

For the VQA benchmarks, we use the validation splits of VQAv2 and VizWiz to construct the reference data. We randomly partition the combined set into two disjoint halves: one half is used as the reference bank, and the other half is used to train RefGen.

For the mathematical and science reasoning tasks, we use the Mixture-of-Thoughts data from OPEN-R1 (Face, 2025), which provides large-scale tuples of problems, reasoning traces, and answers generated by DeepSeek-R1. From this resource, we build the reference bank using 5,000 math samples and 5,000 science samples covering domains such as physics, chemistry, and biology. We additionally sample another 5,000 instances for training RefGen.

4.3 Baselines

We compare RefGen with the following baselines: (a) IO, which directly outputs the final answer; (b) CoT (Wei et al., 2022); and (c) ComplexCoT (Fu et al., 2022). These methods do not use external references. We further include retrieval-based and learning-based reference selection methods: (d) Sparse-Retrieval with BM25 (Chen et al., 2017); (e) Dense-Retrieval (Karpukhin et al., 2020); (f) Reranker; (g) UDR (Li et al., 2023); (h) Lever-LM (Yang et al., 2024); and (i) MOMENTO (Zhou et al., 2025).

4.4 Experimental setups

Devices All experiments are conducted on a single server equipped with NVIDIA A800 GPUs (80GB). **Backbone models** For language-only tasks, we use Qwen3-8B and Qwen3-32B (Yang et al., 2025), as well as Qwen-turbo[1]. For vision-language tasks, we use Qwen2.5-VL 3B[2] and Deepseek-VL2 2.8B (Wu et al., 2024). Unless otherwise specified, all backbone models use greedy decoding at inference time. **RefGen configuration** For language-only tasks, the embedder $E(\cdot)$ is initialized with BGE-en (Xiao et al., 2024). For vision-language tasks, $E(\cdot)$ is initialized with CLIP ViT-B/32 (Radford et al., 2021). On top of the frozen embedder, RefGen uses a lightweight Transformer encoder–decoder with one encoder layer and one decoder layer. The hidden size is matched to that of the corresponding embedder. **Reference selection settings** We first apply dense retrieval to obtain $n = 64$ candidate references from the reference bank. RefGen then autoregressively selects the final $k = 4$ referred demonstrations from this candidate pool. **Training protocol** We implement training with the OPEN-R1[3] framework. RefGen is trained with supervised fine-tuning followed by reinforcement learning with verifiable rewards.

4.5 Main results

Results on mathematical reasoning and science QA Table 1 reports results on the math and science benchmarks with Qwen-turbo as the backbone. RefGen consistently achieves the best performance across all datasets. On Qwen-turbo, it reaches 95.3% on Math-500, 86.7% on AIME-24, and 81.1% on AIME-25, outperforming the strongest baseline, MOMENTO (Zhou et al., 2025), by 1.7, 5.5, and 4.4 points, respectively. Similar improvements are observed on MMLU-Pro and GPQA, where RefGen also surpasses both retrieval-based methods and learned selectors such as UDR (Li et al., 2023) and Lever-LM (Yang et al., 2024). The same trend holds across Qwen3-8B and Qwen3-32B, indicating that the gains of RefGen are robust across backbone scales. The improvements are especially clear on the more difficult benchmarks, suggesting that autoregressive reference composition is more effective than independent retrieval or reranking when the context budget is limited.

Results on visual question answering Table 2 reports results on VQAv2 (Goyal et al., 2017), VizWiz (Gurari et al., 2018), and OK-VQA (Marino et al., 2019) with Deepseek-VL2 2.8B and Qwen2.5-VL 3B. RefGen again delivers the best performance on all benchmarks and backbones. On Deepseek-VL2, it improves over Lever-LM (Yang et al., 2024) by 2.8, 3.2, and 4.4 accuracy points

[1] https://help.aliyun.com/zh/model-studio/qwen-api-reference/

[2] https://huggingface.co/Qwen/Qwen2.5-VL-3B-Instruct

[3] https://github.com/huggingface/open-r1

Method	Math-500 pass@1	AIME-24 pass@4	AIME-25 pass@4	MMLU-Pro acc	GPQA acc	Avg.
IO	86.2	0.0	0.0	60.6	68.1	43.0
COT	91.1	74.4	68.4	72.6	71.6	75.6
ComplexCoT	90.5	66.9	64.2	73.3	71.0	73.2
Sparse-Retrieval	91.6	77.8	77.0	78.2	73.5	79.6
Dense-Retrieval	92.3	79.8	72.9	78.6	74.1	79.5
Reranker	92.1	76.5	69.1	78.9	72.7	77.9
UDR	92.6	70.2	68.8	78.3	74.4	76.9
Lever-LM	93.0	76.5	73.8	78.7	74.9	79.4
MOMENTO	93.4	81.0	76.5	79.2	75.5	81.1
Ours	**95.3**	**86.7**	**81.1**	**81.6**	**77.4**	**84.4**

Table 1: Overall performance on the math solving and QA benchmarks, on the Qwen-turbo LLM backbone.

Method	VQAv2 acc	VizWiz acc	OK-VQA acc	Avg.
IO	66.6	39.9	36.6	47.7
Sparse-Retrieval	68.0	50.5	37.3	51.9
Dense-Retrieval	73.1	51.5	37.7	54.1
Reranker	74.0	52.7	38.2	55.0
UDR	74.2	55.2	43.4	57.6
Lever-LM	75.1	57.0	44.1	58.7
MOMENTO	76.0	57.6	46.6	60.1
Ours	**79.4**	**59.7**	**49.6**	**62.9**

Table 2: Overall performance on the visual question answering benchmarks.

on VQAv2, VizWiz, and OK-VQA, respectively; similar gains are observed on Qwen2.5-VL. The largest improvements appear on OK-VQA, a benchmark that requires stronger knowledge grounding, which is consistent with the benefit of composing complementary referred cases rather than selecting references independently.

Method	MMLU-Pro acc	VQAv2 acc
SFT	79.0	75.2
PPO	79.7	77.2
REINFORCE	78.6	76.5
GRPO	79.8	77.7
DAPO	**81.6**	**79.4**

Table 3: Ablation on the training method for RefGen.

4.6 Ablation studies and further analysis

Ablation on the training method for RefGen We compare several optimization strategies for RefGen, including supervised fine-tuning (SFT), REINFORCE, PPO (Schulman et al., 2017), GRPO (Shao et al., 2024), and DAPO (Yu et al., 2025). As shown in Table 3, SFT gives the weakest results, indicating that fixed oracle sequences are insufficient for learning high-utility reference combinations. Reinforcement learning with verifiable rewards consistently improves performance, with PPO and GRPO outperforming REINFORCE due to more stable policy updates. DAPO achieves the strongest results in this comparison and is therefore adopted in our main setting.

Embedder model	MMLU-Pro
all-mpnet-base-v2	80.8
contriever	81.0
GTE-base	81.3
GTE-large	81.6
BGE-base-en	81.5
BGE-large-en	81.8

Table 4: Ablation on the embedder model $E()$.

Ablation on the embedder model $E(\cdot)$ To examine the effect of the pretrained embedder, we compare six text embedding models on MMLU-Pro: all-mpnet-base-v2 (Reimers and Gurevych, 2019), Contriever (Izacard et al., 2021), GTE-

base and GTE-large[4], BGE-base-en (Xiao et al., 2024), and BGE-large-en (Xiao et al., 2024). Table 4 shows that RefGen performs strongly with all of them, ranging from 80.9% to 81.8% accuracy. BGE-large-en gives the best result, but the overall variation is small, suggesting that RefGen is not highly sensitive to the specific embedding model. Larger embedders consistently perform slightly better than their base variants, indicating that stronger semantic representations can modestly improve reference selection.

Although larger embedders yield slightly better results, they also increase computational cost. In practice, the choice of embedder can therefore be made according to the target trade-off between accuracy and efficiency.

5 Conclusion

We introduced *RefGen*, a lightweight and modular framework that formulates reference selection as autoregressive index generation over a retrieved candidate pool. By generating an ordered sequence of reference indices, RefGen explicitly models dependencies among referred demonstrations while avoiding free-form demonstration generation. The framework can be attached to frozen LLMs and VLMs in a plug-and-play manner, and can be optimized with supervised fine-tuning and reinforcement learning using verifiable rewards. Experiments on mathematical reasoning, scientific question answering, and visual question answering benchmarks show that RefGen consistently outperforms retrieval-based and learning-based baselines, with especially clear gains on more challenging tasks. Overall, the results indicate that learned reference composition is an effective and scalable approach for improving test-time reasoning in foundation models such as Qwen (Yang et al., 2025) and DeepSeek (Wu et al., 2024; Guo et al., 2025).

Limitations

Despite the consistent gains across multiple benchmarks and backbone models, RefGen has several limitations.

First, RefGen depends on a predefined reference bank and an initial retrieval stage to form the candidate pool. Its effectiveness is therefore bounded by the coverage and quality of the available references. If relevant referred demonstrations or referred cases are missing from the reference bank, RefGen can only compose from suboptimal candidates. In addition, building and maintaining a large, high-quality reference bank introduces extra storage and system overhead in practical deployments.

Second, RefGen currently operates with a fixed candidate pool size and a fixed output length (e.g., $n = 64$ and $k = 4$). This design simplifies training and inference, but it may be restrictive for tasks with highly variable context budgets or more complex dependency structures among references. Extending RefGen to adaptive-length selection or more structured reference composition is an important direction for future work.

Third, our training setup relies on verifiable downstream rewards, such as exact-match accuracy for mathematical reasoning and question answering. This assumption is suitable for benchmark tasks with clear automatic evaluation, but it is less applicable to open-ended generation, subjective judgments, or safety-sensitive settings, where reliable reward design is more difficult. As a result, the current optimization strategy may not transfer directly to all application scenarios.

Fourth, although RefGen adds only modest overhead relative to full LLM/VLM inference, it still requires extra computation for embedding, retrieval, and autoregressive index decoding. This cost may matter in latency-sensitive settings. Moreover, our experiments focus on a limited set of model families and benchmarks, so broader evaluation on additional domains, longer-context settings, and larger-scale workloads is still needed to fully assess generalization and scalability.

Finally, RefGen is designed to compose retrieved references rather than discover entirely new knowledge. Its benefit is therefore strongest when useful referred demonstrations already exist and can be effectively reused. Future work should study how reference composition can be combined with stronger retrieval, dynamic reference bank updates, and broader forms of external knowledge augmentation.

[4]https://huggingface.co/thenlper/gte-large

References

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé

Jégou. 2024. The faiss library. *arXiv preprint arXiv:2401.08281*.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2022. Complexity-based prompting for multi-step reasoning. *arXiv preprint arXiv:2210.00720*.

Jonas Gehring, Kunhao Zheng, Jade Copet, Vegard Mella, Quentin Carbonneaux, Taco Cohen, and Gabriel Synnaeve. 2024. Rlef: Grounding code llms in execution feedback with reinforcement learning. *arXiv preprint arXiv:2410.02089*.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. 2018. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3608–3617.

Jian Hu. 2025. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*.

Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A Smith, and Mari Ostendorf. 2022. In-context learning for few-shot dialogue state tracking. *arXiv preprint arXiv:2203.08568*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. 2024. Vineppo: Refining credit assignment in rl training of llms. *arXiv preprint arXiv:2410.01679*.

Li Li, Jiawei Peng, Huiyi Chen, Chongyang Gao, and Xu Yang. 2024. How to configure good in-context sequence for visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26710–26720.

Xiaonan Li, Kai Lv, Hang Yan, Tianya Lin, Wei Zhu, Yuan Ni, Guo Tong Xie, Xiaoling Wang, and Xipeng Qiu. 2023. Unified demonstration retriever for in-context learning. *ArXiv*, abs/2305.04320.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. 2019. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pages 3195–3204.

Long Ouyang, Jeffrey 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.

Chengwei Qin, Aston Zhang, Chen Chen, Anirudh Dagar, and Wenming Ye. 2024. In-context learning with iterative demonstration selection. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7441–7455.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

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*.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Jiashuo Sun, Jihai Zhang, Yucheng Zhou, Zhaochen Su, Xiaoye Qu, and Yu Cheng. 2024. Surf: Teaching large vision-language models to selectively utilize retrieved information. *arXiv preprint arXiv:2409.14083*.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Huai hsin Chi, F. Xia, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903.

Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, and 1 others. 2024. Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. *arXiv preprint arXiv:2412.10302*.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 641–649.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Xu Yang, Yingzhe Peng, Haoxuan Ma, Shuo Xu, Chi Zhang, Yucheng Han, and Hanwang Zhang. 2024. Lever lm: configuring in-context sequence to lever large vision language models. *Advances in Neural Information Processing Systems*, 37:100341–100368.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Yifan Zhang and Team Math-AI. 2024. American invitational mathematics examination (aime) 2024.

Yifan Zhang and Team Math-AI. 2025. American invitational mathematics examination (aime) 2025.

Huichi Zhou, Yihang Chen, Siyuan Guo, Xue Yan, Kin Hei Lee, Zihan Wang, Ka Yiu Lee, Guchun Zhang, Kun Shao, Linyi Yang, and 1 others. 2025. Memento: Fine-tuning llm agents without fine-tuning llms. *arXiv preprint arXiv:2508.16153*.

Yucheng Zhou, Xiang Li, Qianning Wang, and Jianbing Shen. 2024. Visual in-context learning for large vision-language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 15890–15902.