TED-RAG: A Tree-Structured Multi-Expert Framework for Domain-Specific Data Synthesis in Retrieval-Augmented Generation

Sheng Li†, Lei Pan‡, Shengjie Sun‡, Haodong Zhou‡ and Shuai Fan‡*
†Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China
chant_li@sjtu.edu.cn
‡AISpeech Technology Co., Ltd. Suzhou, China
{lei.pan,shengjie.sun,haodong.zhou,shuai.fan}@aispeech.com

***Abstract*—We propose a framework named TED-RAG for generating high-quality, domain-specific question-answer (QA) data using collaboration among multiple expert large language models (LLMs), aiming to train specialized QA models. The framework follows a tree-structured generation approach and includes three core components: Generator, which produces QA pairs through multi-turn dialogues; Rewarder, which evaluates the quality of generated questions; and Prompter, which provides feedback to guide iterative optimization. We employ models such as Qwen2.5-72B-Instruct along with well-crafted prompt templates to generate diverse QA pairs and automate the refinement of outputs through the Rewarder and Prompter. We conducted experiments on DragonBall and ChatRAG. Results show that TED-RAG significantly outperforms traditional single-model approaches in both diversity and quality of generated QA data, leading to improved performance in downstream tasks. We further validate the effectiveness of our approach through comprehensive ablation studies.**

***Index Terms*—RAG, Large Language Model, Data Synthesis, Knowledge Distillation.**

I. Introduction

Recent advances in Large Language Models (LLMs) have significantly enhanced their capabilities in open-domain question answering, mathematical reasoning, and path planning. This enhancement is largely due to the integration of reinforcement learning techniques during the training process [25] [3]. However, previous research has highlighted the issue of the scarcity of training data [33]. Consequently, synthesizing training data using large models has garnered significant attention [32] [31] [22] [5]. At the same time, with the increasing demand for privatization, LLMs are increasingly being used with specialized private documents, such as government policy papers or vehicle manuals, for tasks like question answering [30]. In these application scenarios, the evaluation of models primarily focuses on the accuracy of responses within these specific-domain documents rather than their general reasoning abilities. Nevertheless, current data synthesis methods based on large models still struggle to address this issue effectively.

* Corresponding author.

In this work, we propose a specific-domain text QA data generation framework with multi-agent collaborative, multi-expert retrieval augmented generation (RAG) scenarios (the overall architecture of our method is illustrated in Figure 1). The framework consists of three distinct types of agents: the Generator, the Prompter, and the Rewarder. This approach facilitates the creation of high-quality, specific-domain RAG training data using only background text as input. The process leverages tree-based generation, multi-expert decision-making, and automated dynamic enhancement. The framework is highly flexible and extensible: (1) the number of models in the Generator and the amount of generated data can be customized flexibly; (2) the method also supports the use of alternative models to serve as the Rewarder and Prompter.

Specifically, we evaluate our method on DragonBall [29] and ChatRAG [28], as these datasets provide a convenient testbed for assessing performance on downstream tasks. Starting from the background documents, we sequentially perform three tasks: question generation, question evaluation, and question refinement suggestions. We employ carefully designed prompts to guide the models in executing different tasks, while iteratively generating data across multiple rounds.

Finally, we fine-tune the Qwen2.5-14B-Instruct [27] model using the generated QA data and measure the accuracy of model responses on downstream tasks to validate the effectiveness of our approach. Experimental results show that our method achieves significant improvements over existing QA data synthesis approaches. Moreover, we analyze how the depth and breadth of the generation tree influence the overall performance of QA data synthesis.

In summary, our contributions are as follows:

- A fully automated framework for high-quality, domain-specific QA pair synthesis without human intervention.
- State-of-the-art performance on DragonBall and significant gains over SOTA on ChatRAG.
- We compare our work with prior QA data synthesis approaches, showing that our method generates the highest-quality data with minimal human intervention.

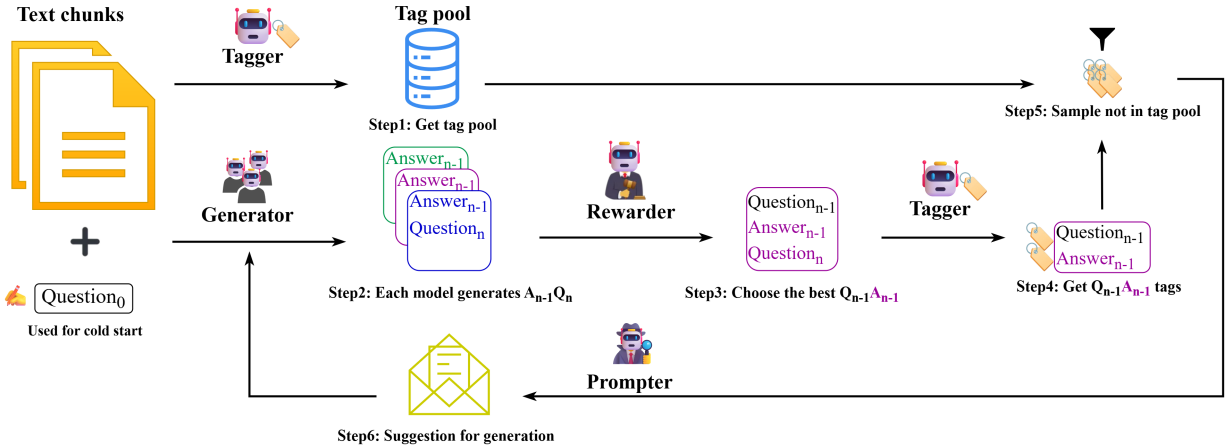


Fig. 1: Detailed TED-RAG architecture: Tagger extracts tags; Generator produces candidates/new questions; Rewarder scores QA pairs; Prompter samples uncovered tags for guidance.

- The experimental findings also validate the effectiveness of reinforcement learning and the multi-expert paradigm in QA data synthesis.

II. Related Work

Retrieval-Augmented Generation (RAG). RAG [26] enhances LLMs by integrating an external knowledge base or leveraging information retrieved from web searches. This approach indirectly expands the model's context size, significantly reducing hallucinations. Furthermore, RAG has been shown to improve the performance of LLMs across various NLP tasks, such as specific-domain question answering [24] [23] and reasoning tasks [21] [20].

Fine-tuning RAG. Recent work has focused on fine-tuning different components of RAG to achieve higher-quality outputs from the RAG pipeline [14]. At their core, these approaches aim to improve the end-to-end generation quality of RAG. Our work specifically focuses on the generator component.

LLMs-Driven Data Synthesis and Data Augmentation. There have been a significant amount of previous works leveraging LLMs for data synthesis [31] [22] [15]. These synthesized datasets have been applied to various task scenarios, such as instruction following [19] [18], privacy and security [32], or serving as evaluation data [29]. Additionally, previous works have shown that tree-based structures can effectively stimulate the creativity of large language models [18] [17]. However, some works have proposed a new application scenario in which LLMs are able to incorporate specific-domain knowledge while improving the performance of RAG for specific-domain tasks [30]. This scenario has seen limited exploration in terms of data synthesis, and our work focuses precisely on this area.

Knowledge distillation. Knowledge distillation [16] [13] usually involves using a high-performing large model to improve the performance of smaller models. While our approach shares a similar objective with knowledge distillation, it differs in a key aspect. Most existing distillation methods primarily align the behaviors of large and small models based on available task-specific datasets. In contrast, our method focuses on utilizing the generative capabilities of large models to enhance smaller models in a target domain, without depending on pre-existing task datasets.

LLMs-Driven Multi-Agent Collaboration. Multi-agent systems are an active research area in NLP, aiming to solve complex tasks through the collaboration of multiple agents. These systems have demonstrated potential in tasks such as dialogue systems, question answering, and sentiment analysis [10]. The introduction of LLMs has significantly enhanced the capabilities of multi-agent systems, particularly in reasoning and planning [9] [8]. However, our work differs from previous efforts by focusing on stimulating creativity among multiple agents.

III. Methodology

A. Overall Structure

Building on prior work [18] [17], we adopt a tree-based generation framework (Figure 1 Step 2), where each node represents a QA pair. Starting from background knowledge as the root, our method expands nodes using diverse generation strategies. A Rewarder (Section III-C) evaluates candidates, enabling the selection of the optimal generation path.

An expanded view is shown in Figure 1.We employ an LLM to generate $Question_0$ from text chunks for cold start. Two generator groups interact via QA-based reasoning over the document. The Generator (Section III-B) produces diverse QA pairs using varied prompts. The Rewarder scores each pair, and the top result is fed to the Prompter (Section III-D), which generates adaptive prompts for subsequent rounds, incorporating QA history and scores. Inspired by INSTAG [7], we use tagging to track coverage and promote diversity.

Algorithm 1 TED-RAG Algorithm

Input: Document D, Generator List G_L, Rewarder R, Prompter P, Tagger T, Rounds N_r, samples per model $N_{q\_per\_model}$
Output: QA Set S_{QA}

$S_{QA} \leftarrow \emptyset$; $r \leftarrow 0$
$S_{DocTag} \leftarrow T(D)$; $L_{QA} \leftarrow []$
$Q_{best} \leftarrow G(D)$; $S_r \leftarrow \emptyset$
while $r < N_r$ **do**
  for $G \in G_L$ **do**
    for $i = 0$ to $N_{q\_per\_model}$ **do**
      $A_r Q_{r+1} \leftarrow G(D, Q_{best}, S_r)$
      L_{QA}.append$(A_r Q_{r+1})$
    end for
  end for
  $R_{QA} = R(D, Q_{best}, A_r)$ for $A_r Q_{r+1} \in L_{QA}$
  Choose best QA in L_{QA} based on R_{QA}
  S_{QA}.add(Q_{best}, A_{best})
  $Q_{best} \leftarrow Q_{r+1}$ which A_r is A_{best}
  $S_S = sample(S_{DocTag} - T(Q_{best}, A_{best})$
  $S_r \leftarrow P(Q_{best}, A_{best}, R_{QA}, S_S))$
  $r \leftarrow r + 1$
end while
return S_{QA}

Algorithm 1 outlines the full procedure.

B. Generator

The Generator consists of multiple LLMs and is responsible for generating data based on background knowledge and prompts. Its process can be simplified into the following formula (1).

$$A_{n-1}Q_n = G(D, A_{n-2}Q_{n-1}, S_{n-1}) \tag{1}$$

Where, G represents a generator composed of multiple models, and n denotes the current generation round. In the generation results, A_{n-1} indicates an answer to Q_{n-1}, while Q_n signifies the formulation of a new question. S refers to the generation suggestions provided by the Prompter (Section III-D). Additionally, we impose constraints on the number of historical dialogue rounds that G can reference, as well as the amount of data generated in each round.

C. Rewarder

The Rewarder consists of a single model and is responsible for evaluating the data generated by the Generator and selecting the optimal QA data. Its process can be simplified into the following formula(2):

$$Score_n = R(D, Q_n, A_n) \tag{2}$$

where R represents the Rewarder.

D. Prompter

The Prompter is implemented as a single model, tasked with providing suggestions for subsequent generation based on the QA data produced by the Generator and the scores assigned by the Rewarder. Its process can be simplified into the following formula (3):

$$S_n = P(D, Q_n, A_n, Score_n, tags_n) \tag{3}$$

where P represents the Prompter. Additionally, to introduce diversity, we draw inspiration from the work of INSTAG [7] by tagging the text and QA data. During each iteration, the Prompter randomly samples tags from the text's tag pool that are not yet covered by the QA tags.

IV. Experiments

We conducted experiments to evaluate several key components of the entire data generation process. First, we describe the preliminary setup for our experiments, followed by the details of the experiments themselves.

A. Experimental Setup

After obtaining the generated QA data, we proceeded to train the Qwen2.5-14B-Instruct model. Specifically, our evaluation of each method was conducted on the fine-tuned model, with a focus on its performance in downstream tasks. For training, we uniformly employed the DeepSpeed framework [6] to perform full-parameter fine-tuning of the model. During the training process, we used the default hyperparameters, and the number of training epochs was set to 3 for all experiments.

B. Baselines

Base Models. We selected various existing base models to compare with our method, aiming to verify that our approach can effectively integrate the strengths of different models in QA data generation. The chosen base models include DeepSeek-V3 [25], Qwen2.5-72B-Instruct [27], DeepSeek-R1-Distill-Qwen-32B, and Qwen2.5-14B-Instruct. These models are all publicly available and have demonstrated strong performance across a wide range of tasks. We considered both general-purpose generation models (e.g., DeepSeek-V3) and reasoning-optimized models (e.g., DeepSeek-R1-Distill-Qwen-32B) in order to comprehensively evaluate the performance of different model types. Notably, Qwen2.5-14B-Instruct, which serves as the target model for subsequent fine-tuning, is also included in this evaluation.

RAFT. RAFT [30] generates multiple questions for each document and uses LLMs to complete their chain of thought and answers. We re-implemented their method using the prompt provided in the original paper, with DeepSeek-V3 as the backbone model. Specifically, we generated 5 questions per document and completed their corresponding reasoning chains. During training, we further introduced 3 randomly selected distractor documents for each background article to increase the difficulty of retrieval.

Multi-hop QA. Multi-Hop QA [5] provides a method for constructing QA pairs that require multi-step reasoning. However, our target task focuses on single-step reasoning. Therefore, we adapted the original setup of the method to generate QA data that only involves one-hop reasoning.

Method	SacreBLEU	R1	R2	RL	METEOR	BERTScore	Acc.
Qwen2.5-14B-Instruct [27]	21.24	50.74	32.38	43.45	47.36	69.22	16.80
Qwen2.5-72B-Instruct [27]	24.01	56.55	37.78	48.87	50.19	72.07	17.63
DeepSeek-V3 [25]	17.25	47.95	29.13	40.15	45.01	63.45	13.11
DeepSeek-R1 [3]	17.59	46.03	28.16	38.00	46.83	64.45	16.02
RAFT [30]	7.84	43.65	27.81	39.88	32.04	63.32	13.53
CRAFT [4]	10.53	42.16	22.31	33.64	41.97	61.37	15.44
Multi-Hop QA [5]	0.10	24.85	16.14	24.66	17.34	52.34	13.29
TED-RAG (Our Method)	25.47	**56.67**	**37.91**	**49.64**	**50.58**	**72.97**	18.01
TED-RAG + distractor data	**25.49**	53.88	35.23	45.89	48.54	70.63	17.78
TED-RAG + COT	25.11	54.16	35.47	46.97	50.25	70.85	**18.19**
TED-RAG + distractor data + COT	24.48	53.35	34.65	46.01	48.98	69.86	17.75

TABLE I: The end-to-end RAG evaluation results of our method on the DragonBall dataset are presented, where accuracy is measured as whether the model's generated response fully covers the golden answer. R denote ROUGE. The **bolded** values indicate the best results under the corresponding metrics, underline indicates the second-best result, and double underline indicates the third-best result.

Method	QuAC	QReCC	TopiOCQA	INSCIT	CoQA	ConvFinQA	Avg.
Qwen2.5-14B-Instruct [27]	30.08	34.26	26.61	33.88	10.39	42.50	29.62
Qwen2.5-72B-Instruct [27]	32.69	34.67	27.56	33.77	10.53	47.00	31.04
RAFT [30]	21.56	31.42	27.17	**34.10**	8.95	44.50	27.95
CRAFT [4]	25.56	32.77	26.25	33.94	9.22	**69.00**	32.79
Multi-Hop QA [5]	17.98	19.75	17.42	22.22	9.45	48.00	22.47
TED-RAG (Our Method)	**33.71**	**37.61**	**32.27**	32.10	**48.04**	65.50	**41.54**

TABLE II: The Performance of Our Method on the ChatRAG Test Set.

CRAFT. CRAFT [4] introduces a few-shot-based QA data generation framework. We adopted the prompt templates provided in the original paper and randomly sampled 3 prompts during each generation step to form the few-shot context.

C. Workflow Configuration

We configured the workflow using Qwen2.5-72B-Instruct and DeepSeek-V3 as the expert models in the Generator responsible for QA data generation. In each round, each model generates four candidate outputs. DeepSeek-R1 serves as the Rewarder, with prompting strategies partially inspired by QGEval [2] to score and select high-quality generations. Additionally, we employed DeepSeek-V3 as the Prompter, which draws on part of the prompting design from WizardLM [18] to provide adaptive guidance for subsequent generations, aiming to improve the overall quality of the generated data.

D. Results

We evaluated our method on the DragonBall dataset [29] and ChatRAG [28] datasets, with results summarized in table I and table II.

For the DragonBall dataset, we designed four training settings: (1) using the document from which the QA pairs were generated as the background context; (2) adding three noise documents to setting (1); (3) incorporating chain of thought (COT) during inference on top of setting (1); (4) adding three noise documents to setting (3). During evaluation, we used the top-5 most relevant text chunks retrieved by BGE-M3 [12] based on user queries as the background context for model generation.

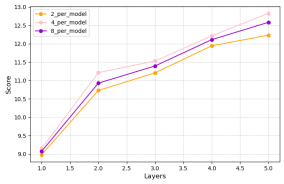
For ChatRAG, we selected six test datasets (Quac, Qrecc, Topiocqa, Inscit, Coqa, Convfinqa) and randomly sampled 200 instances from each to validate the effectiveness of our method. During training, we only used the specific text chunk involved in QA generation as the background document.

Experimental results show that our method achieves superior performance across all evaluation metrics on the DragonBall dataset compared to other approaches.

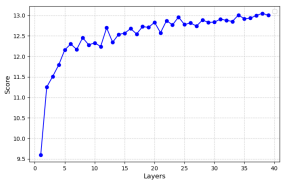
However, we observe that incorporating COT and distractor documents into the training data does not yield significant improvements in performance—in fact, it leads to worse results on several metrics compared to using only the text chunks from which the QA pairs were generated as context during training.

Additionally, it is worth noting that models from the DeepSeek series did not perform as well, likely due to the severe hallucination issues of the R1 model and the use of an INT8 quantized version of the V3 model.

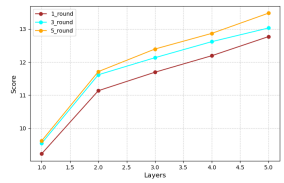
For the ChatRAG test set, our method achieves the best performance on all datasets except INSCIT and ConvFinQA. Moreover, the average performance across all datasets is approximately 10% higher than that of the second-best approach. In addition, on INSCIT and ConvFinQA, the performance gap



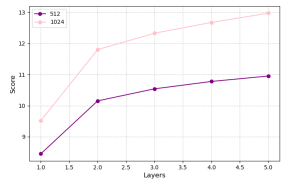
(a) The trend of QA data quality across different generation depths, under varying branching factors.



(b) The trend of model generation quality as depth increases.



(c) The impact of the number of dialogue history turns on generation performance.



(d) The impact of background text size on data generation quality in QA generation.

Fig. 2: Parameter trends with generation depth. Scores computed using INF-ORM-Llama3.1-70B; low-quality samples (score < 0) excluded.

Chunk Method	BERTScore	Acc.
512		
Constant length	66.49	15.41
Slide Window	66.51	15.97
Semantic	66.55	16.21
1024		
Constant length	68.60	17.04
Slide Window	68.62	17.66
Semantic	**68.66**	**17.92**

TABLE III: The end-to-end RAG generation evaluation results of the model fine-tuned on data generated by our method, under different chunking strategies and chunk sizes.

between our method and the best-performing method is only 2% and 4%, respectively, demonstrating the effectiveness and competitiveness of our approach.

V. Hyperparameter Analysis

We evaluate key hyperparameters using INF-ORM-Llama3.1-70B [1] to score generated QA pairs, retaining only positivescore samples. Downstream valuation is performed on the DragonBall test set.

A. Number of Generation Branches

The branching factor controls candidate outputs per generation round. As shown in Figure 2a, values below 4 limit diversity, while values above 4 degrade selection quality due to rewarder overload. We set it to 4.

B. Depth of the Generation Tree

Depth determines the number of iterations for optimization. When the depth is low, performance increases rapidly and then fluctuates at a slower rate (Figure 2b). Considering computational costs and the stability of the method. We have fixed the depth to 5.

C. Number of Historical Dialogue Turns

Retaining more than 3 prior turns yields diminishing returns (Figure 2c), with marginal gains outweighed by increased context and memory costs. We use the last 3 turns.

D. Size of the Text in QA Generation

Longer texts generally contain richer information. Given our experimental setup (e.g., generation depth), we aim for the model to make full use of the background text during generation. However, we observe that when the background text is too short, the model quickly covers all available information in early generations. As a result, the quality of the generated data drops significantly in later rounds (as illustrated in Figure 2b as well).

Experimental results also show (see Figure 2d) that longer background texts can improve generation quality by providing more diverse and comprehensive information for the model to draw upon.

E. Chunking Strategy

We compare constant, sliding, and semantic chunking at 512 and 1024 tokens. Semantic splitting with 1024-token chunks achieves the highest BERTScore and accuracy (Table III), preserving coherence and minimizing noise. This strategy is used in all experiments.

F. Human Consistency Evaluation

In our approach, the role of the rewarder is critical, as it directly determines the quality of the final QA data. We select the model to serve as the rewarder by evaluating its consistency with human judgments. The human evaluation protocol follows that of QGEval [2]. We consider several representative models as candidate rewarders and conduct a consistency analysis between their outputs and human assessments. The results are summarized in Table IV.

The results indicate that DeepSeek-R1 achieves a high level of agreement with human evaluators. This suggests that, under our specified evaluation criteria, DeepSeek-R1 can effectively approximate human judgment. Consequently, the findings not only demonstrate that DeepSeek-R1 is the most suitable general-purpose model to act as the rewarder but also indirectly confirm that the QA data generated by our method attains high scores under human evaluation.

VI. Conclusion

We propose a tree-based domain-specialized data synthesis method for RAG, using multi-expert dialogue and rewardguided refinement. The approach enhances small-model RAG performance on private datasets by leveraging large

Modelname	RMSE ↓	MAE ↓	Average kkapa ↑	Consistency rate ↑
QwQ-32B [11]	0.2625	0.1232	0.5927	86.78
Qwen2.5-72B-Instruct [27]	0.2426	0.1156	0.5887	91.02
DeepSeek-V3 [25]	0.2715	0.1211	0.5853	89.89
DeepSeek-R1 [3]	**0.2388**	**0.1088**	**0.5928**	**93.22**

TABLE IV: Evaluation of agreement between model-based and human assessments. The direction of the arrow indicates whether a higher or lower value of the metric is preferable, and **bolded** values denote the best performance under the corresponding metric.

language models. Experiments on DragonBall and ChatRAG validate its effectiveness. The framework is extensible—by modifying the generator or rewarder, it can adapt to diverse tasks.

VII. Acknowledgments

The authors used Qwen (Alibaba Cloud) solely as a writing assistant for language polishing and stylistic improvement. All experimental work, data analysis, and research conclusions were conducted independently by the authors without AI involvement.

References

[1] X. Yang, M. Tan, and C. Qu, "Inf-orm-llama3.1-70b," 2024. [Online]. Available: https://huggingface.co/infly/INF-ORM-Llama3.1-70B

[2] W. Fu, B. Wei, J. Hu, Z. Cai, and J. Liu, "Qgeval: Benchmarking multi-dimensional evaluation for question generation," arXiv preprint arXiv:2406.05707, 2024.

[3] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, et al., "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," arXiv preprint arXiv:2501.12948, 2025.

[4] I. Ziegler, A. Köksal, D. Elliott, and H. Schütze, "Craft your dataset: Task-specific synthetic dataset generation through corpus retrieval and augmentation," arXiv preprint arXiv:2409.02098, 2024.

[5] M. Chen, X. Chen, and W. Yih, "Few-shot data synthesis for open domain multi-hop question answering," arXiv preprint arXiv:2305.13691, 2023.

[6] J. Rasley, S. Rajbhandari, O. Ruwase, and Y. He, "Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters," in Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2020, pp. 3505–3506.

[7] K. Lu, H. Yuan, Z. Yuan, R. Lin, J. Lin, C. Tan, et al., "# instag: Instruction tagging for analyzing supervised fine-tuning of large language models," arXiv preprint arXiv:2308.07074, 2023.

[8] H. Li, Y. Chong, S. Stepputtis, J. Campbell, D. Hughes, M. Lewis, and K. Sycara, "Theory of mind for multi-agent collaboration via large language models," arXiv preprint arXiv:2310.10701, 2023.

[9] L. Xu, Z. Hu, D. Zhou, H. Ren, Z. Dong, K. Keutzer, et al., "Magic: Investigation of large language model powered multi-agent in cognition, adaptability, rationality and collaboration," arXiv preprint arXiv:2311.08562, 2023.

[10] M. Baroni, R. Desì, and A. Lazaridou, "Emergent language-based coordination in deep multi-agent systems," in Proc. 2022 Conf. Empirical Methods Natural Language Process.: Tutorial Abstracts, 2022, pp. 11–16.

[11] Qwen Team, "QwQ-32B: Embracing the power of reinforcement learning," Mar. 2025. [Online]. Available: https://qwenlm.github.io/blog/qwq-32b/

[12] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "BGE M3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation," arXiv preprint arXiv:2402.03216, 2024.

[13] C. Hsieh, C. Li, C. Yeh, H. Nakhost, Y. Fujii, A. Ratner, et al., "Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes," arXiv preprint arXiv:2305.02301, 2023.

[14] B. Wang, W. Ping, L. McAfee, P. Xu, B. Li, M. Shoeybi, et al., "InstructRetro: Instruction tuning post retrieval-augmented pretraining," arXiv preprint arXiv:2310.07713, 2024.

[15] M. Schmidt, A. Bartezzaghi, and N. T. Vu, "Prompting-based synthetic data generation for few-shot question answering," arXiv preprint arXiv:2405.09335, 2024.

[16] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv preprint arXiv:1503.02531, 2015.

[17] Z. Wang, A. Liu, H. Lin, J. Li, X. Ma, and Y. Liang, "Rat: Retrieval augmented thoughts elicit context-aware reasoning in long-horizon generation," arXiv preprint arXiv:2403.05313, 2024.

[18] C. Xu, Q. Sun, K. Zheng, X. Geng, P. Zhao, J. Feng, et al., "Wizardlm: Empowering large language models to follow complex instructions," arXiv preprint arXiv:2304.12244, 2023.

[19] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi, "Self-instruct: Aligning language models with self-generated instructions," arXiv preprint arXiv:2212.10560, 2022.

[20] Z. Zhang, X. Zhang, Y. Ren, S. Shi, M. Han, Y. Wu, et al., "Iag: Induction-augmented generation framework for answering reasoning questions," arXiv preprint arXiv:2311.18397, 2023.

[21] Z. Wang, X. Pan, D. Yu, D. Yu, J. Chen, and H. Ji, "Zemi: Learning zero-shot semi-parametric language models from multiple tasks," arXiv preprint arXiv:2210.00185, 2022.

[22] N. Thakur, J. Ni, G. Hernández Ábrego, J. Wieting, J. Lin, and D. Cer, "Leveraging LLMs for synthesizing training data across many languages in multilingual dense retrieval," arXiv preprint arXiv:2311.05800, 2024.

[23] S. Siriwardhana, R. Weerasekera, E. Wen, T. Kaluarachchi, R. Rana, and S. Nanayakkara, "Improving the domain adaptation of retrieval augmented generation (RAG) models for open domain question answering," Trans. Assoc. Comput. Linguist., vol. 11, pp. 1–17, 2023.

[24] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-rag: Learning to retrieve, generate, and critique through self-reflection," in Proc. Int. Conf. Learn. Represent. (ICLR), 2024.

[25] DeepSeek-AI, "DeepSeek-V3 Technical Report," arXiv preprint arXiv:2412.19437, 2025.

[26] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, et al., "Retrieval-augmented generation for knowledge-intensive nlp tasks," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 33, 2020, pp. 9459–9474.

[27] A. Yang, B. Yu, C. Li, D. Liu, F. Huang, H. Huang, et al., "Qwen2.5-1m technical report," arXiv preprint arXiv:2501.15383, 2025.

[28] Z. Liu, W. Ping, R. Roy, P. Xu, C. Lee, M. Shoeybi, and B. Catanzaro, "Chatqa: Surpassing gpt-4 on conversational qa and rag," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 37, 2024, pp. 15416–15459.

[29] K. Zhu, Y. Luo, D. Xu, Y. Yan, Z. Liu, S. Yu, et al., "Rageval: Scenario specific rag evaluation dataset generation framework," arXiv preprint arXiv:2408.01262, 2024.

[30] T. Zhang, S. G. Patil, N. Jain, S. Shen, M. Zaharia, I. Stoica, and J. E. Gonzalez, "Raft: Adapting language model to domain specific rag," arXiv preprint arXiv:2403.10131, 2024.

[31] X. Li, X. Li, H. Zhang, Z. Du, P. Jia, Y. Wang, et al., "Syneg: Llm-driven synthetic hard-negatives for dense retrieval," arXiv preprint arXiv:2412.17250, 2024.

[32] S. Zeng, J. Zhang, P. He, J. Ren, T. Zheng, H. Lu, et al., "Mitigating the privacy issues in retrieval-augmented generation (rag) via pure synthetic data," arXiv preprint arXiv:2406.14773, 2024.

[33] N. Muennighoff, A. M. Rush, B. Barak, T. L. Scao, A. Piktus, N. Tazi, et al., "Scaling data-constrained language models," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 36, 2023, pp. 50358–50376.