SARA: Stochastic Adaption of Language Models to In-Domain RAG

Peiyu Xu[1], Naxin Chen[2], Xinyu Liu[3], Denghao Peng[4]
{xupeiyu0827@sjtu.edu.cn[1], chennx@mails.ccnu.edu.cn[2],
xinyu2121@mails.jlu.edu.cn[3], dc22843@umac.mo[4]}

UM-SJTU Joint Institute, Shanghai Jiao Tong University, Shanghai, China[1]
School of Computer Science, Central China Normal University, Wuhan, Hubei Province, China[2]
College of Computer Science and Technology, Jilin University, Changchun, Jilin Province, China[3]
Faculty of Science and Technology, University of Macau, Taipa, Macau, China[4]

Abstract. In the current landscape of pre-trained large language model (LLM) applications, various methods such as data augmentation, Residual Learning (RL), Curriculum Learning (CL), Low-Rank Adaptation (LoRA) and Retrieval-Augmented Generation (RAG), are commonly employed to integrate more targeted information into pre-trained models. However, these methods often fall short in terms of training costs or practical effectiveness. Therefore, finding a more effective approach to enhance the capabilities of large language models in downstream tasks is imperative. In this paper, we introduce Stochastic Adaption of Retrieval Augmentation (SARA), an integrated fine-tuning strategy that introduces chained adaption stages while incorporating features from Retrieval-Augmented Fine-Tuning (RAFT), in order to enhance the LLM's ability of effectively responding to domain-specific queries. By organizing the various adaptation stages in a stochastic manner, we achieve improved performance, robustness and training efficiency compared to existing fine-tuning methods.

Keywords: Stochastic Adaption, Retrieval Augmentation, Fine-tuning

1 Introduction

In recent years, the advancement of large language models (LLMs) have significantly transformed various applications, ranging from conversational agents [1] to automated content generation[2]. These models have shown remarkable capabilities in knowledge reasoning, making them increasingly useful in diverse fields[3]. However, in specialized domains, the ability to retrieve accurate information from specific documents becomes paramount [4]. This paper addresses the pressing need for enhanced information retrieval capabilities within LLMs, particularly in contexts that demand a higher degree of domain-specific knowledge.

CONF-MLA 2024, November 21, Adana, Turkey

DOI 10.4108/eai.21-11-2024.2354624

Current research has explored various methods[5], such as Data Augmentation[6], Residual Learning (RL), Curriculum Learning (CL) [7], Retrieval-Augmented Generation[8], Low-Rank Adaptation (LoRA) [9] and its variations, such as Chain of LoRA (CoLA) [10]. These methods have demonstrated effectiveness in improving model performance, yet there remains substantial room for enhancement, particularly concerning context-aware question-answering capabilities and training efficiency[11].

In this paper, a novel approach is proposed for fine-tuning LLMs by incorporating Retrieval-Augmented Fine-Tuning (RAFT) [12]. By utilizing Stochastic Adaption, we aim to improve the training efficiency and enhance the contextual understanding and robustness of LLMs, enabling them to provide more accurate responses based on specific user queries. We have named this method Stochastic Adaption of Retrieval Augmentation (SARA).

To evaluate its effectiveness, we implemented SARA across various benchmarks to assess its performance in diverse scenarios. The findings highlight SARA's potential to address limitations in existing fine-tuning methods, particularly in handling domain-specific queries.

Through the implementation of SARA, we successfully demonstrate a promising avenue for making LLMs more reliable and accurate for practical applications. This research contributes to the growing body of work that seeks to refine LLM fine-tuning processes, ensuring that these models not only possess expansive knowledge but also excel in contextually relevant information retrieval.

2 Related Work

2.1 RAFT

RAFT (Retrieval-Augmented Fine-Tuning) is a training strategy aimed at enhancing the performance of large language models (LLMs) in domain-specific, "open-book" question-answering tasks [12]. RAFT introduces three critical components that distinguish it from traditional training approaches, and Figure 1 shows RAFT's procedure to process the documents and generate the desired dateset.

The first crucial element of RAFT is the usage of **distractor** documents during training. These documents, irrelevant to the query, are included along with the **golden document**, which is the relevant source in the training set. By exposing the model to a mixture of relevant and irrelevant information, RAFT emulates real-world retrieval conditions, where retrieved documents may contain both useful and non-useful content. This strategy improves the model's ability to differentiate between relevant and irrelevant information, which is an essential skill for better performance in retrieval-augmented generation (RAG) systems.

Another key innovation in RAFT is the introduction of the **P%** hyperparameter, the proportion of training data that includes the golden document. Contrary to common conception, the researchers have found that limiting the inclusion of golden documents to a subset of the training data can enhance model performance. This encourages the model to balance memorization with contextual reasoning, and to develop robustness when extracting relevant information from incomplete or imperfect contexts.

The final core component of RAFT is the incorporation of **Chain-of-Thought (CoT)** reasoning

in the training process [13]. The model is trained to generate a logical reasoning sequence that leads to the answer rather than producing a direct response. The inclusion of CoT significantly improves the model's problem-solving capabilities and generates more accurate and interpretable outputs. This approach helps guide the model by mitigating the chances of overfitting to straightforward answers and ensures it can handle more complex queries with increased reliability.

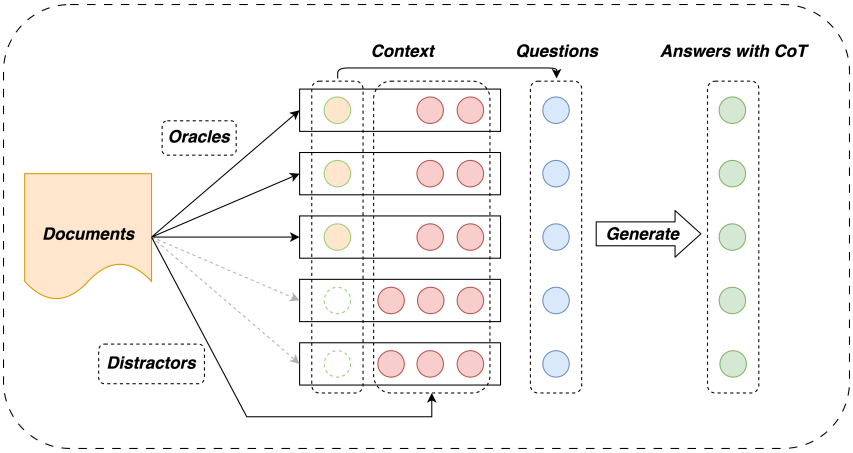

Fig. 1. RAFT procedure: The document is split into chunks (oracles), and questions are generated correspondingly. Oracles appear in context with probability $p\%$, alongside distractors (oracles unrelated to the current question). Answers with CoT reasoning are generated based on the questions and context.

2.2 LoRA and Chain of LoRA

Low Rank Adaptation (LoRA) is a widely used fine-tuning method, whose key idea is to introduce low-rank matrices to make possible parameter-efficient fine-tuning [14]. Based on this, Chain of LoRA (CoLA) tries to diminish the generalization error and improve models' performance by learning a low-rank adaptive matrices repeatedly to form an augmentation in high rank for the LLM weights [10]. Additionally, the procedure of CoLA draws inspiration from the concept of residual learning and extends in the form of a chain structure, which is shown in Figure 2.

Additionally, since RAFT is a fine-tuning method that requires carriers for implementation, instead of full parameter fine-tuning, we will utilize LoRA and CoLA for efficiency.

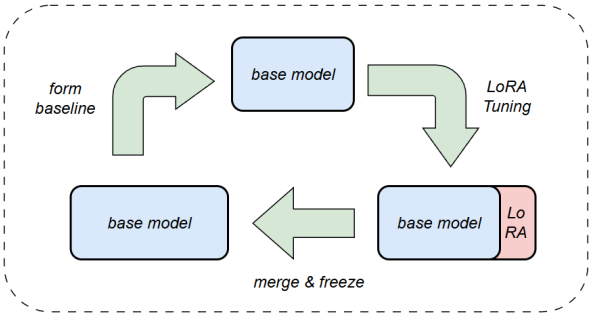

Fig. 2. Iterative framework of Chain of LoRA. CoLA starts with the LoRA tuning on a frozen LLM, then merges the LoRA weights into the model to form a new base model, and introduces new LoRA weights to that model.

3 Our Method

In this section, we present the details of our method, Stochastic Adaptation of Retrieval Augmentation (SARA). Apart from stochasticity, SARA incorporates RAFT's approach to dataset handling and the chain structure from CoLA. Based on this architecture, we further purpose two proven frameworks: SARA-D and SARA-P.

3.1 Algorithm of SARA

In SARA, We partition the entire tuning process into three adaption stages, or layers: **Ada1**, **Ada2**, and **Ada3**, and model weights are merged between stages. Each stage focuses on a distinct yet interconnected purpose, with stochastic elements being utilized in Adas. In the stochastic process, the stochastic decider s and the threshold T play a crucial rule. Furthermore, we will demonstrate that residual learning, curriculum learning and data augmentation are naturally integrated into our method. In the explanation of our algorithm, we will use distraction as the metric.

3.1.1 Ada1

Ada1 refers to the first adaptation layer of the language model, which functions as a layer that allows the model to quickly become familiar with and learn the fundamental aspects of the task. The difficulty level of the data points is reduced, and stochastic elements are applied to augment them back to their original complexity. In the case of augmented data points, the distractors can vary, which aligns with the concept of **data augmentation** by introducing variability.

Algorithm 1 SARA-Ada1 (Stochastic)

Input: original training data points D with number N
Initialize the stochastic decider s_1 and threashold T_1
for $i = 1, \ldots, N$ **do**
    choose a distractor d in $D[i]$
    Update: $D[i] \Leftarrow D[i] - d$
    sample the stochastic decider s_1 from the range $(0, 1)$
    if $s_1 < T_1$ **then**
        select a unique distractor d_u from the rest $N-1$ data points
        Inject: $D[i] \Leftarrow D[i] + d_u$
    end if
    forward pass
    backward pass and update parameters
end for

The Ada1 layer processes each data point in a stochastic manner and feeds it into the forward and backward passes, as depicted in Algorithm 1.

3.1.2 Ada2

In Ada2, the stochastic threshold $T_2 = 0$ and each individual data point remains unchanged, with only the order of training being resampled. The objective of Ada2 is to consolidate the model's knowledge and establish a solid baseline.

Since each data point in Ada1 is processed in a stochastic manner, we can express the relationship between the data points D_{Ada1} and D_{Ada2} as:

$$D_{Ada1}[i] = \begin{cases} D_{Ada2}[i] & \text{with probability } T_1 \\ D_{Ada2}[i] - Distractor & \text{with probability } 1 - T_1 \end{cases} \tag{1}$$

Then we show how residual learning (RL) and curriculum learning (CL) are naturally integrated into our method.

Nature of Residual Learning: We denote the optimal weights tailored for the specific task as W. Then we have

$$W = W_{pretrained} + \Delta W, \tag{2}$$

where ΔW stands for the optimal weight update. For the data points $D_{Ada1}[i] = D_{Ada2}[i]$, since the model has already learned in Ada1 and the weights have been merged, in Ada2, it can be considered as learning the residual of $\Delta W - B_1 A_1$, where B_1 and A_1 represent the low-rank matrices that were learned and merged in Ada1. This transforms the problem into a simpler optimization task, as it no longer involves learning ΔW from the ground up.

Nature of Curriculum Learning: We denote the set of data points $D_{Ada1}[i] = D_{Ada2}[i] - Distractor$ as I, i.e.,

$$I = \{D_{Ada2}[i] \,|\, D_{Ada1}[i] = D_{Ada2}[i] - Distractor\} \tag{3}$$

and the difficulty of each data point as D_i. As the model learns, it progressively focuses on data points with increasing difficulty, represented as:

$$W = W_{pretrained} + \sum_{i \in I} \Delta W_i \quad \text{where } D[i] \text{ increases for } i \in I, \tag{4}$$

where I is the index set of data points corresponding to increasing difficulty.

3.1.3 Ada3

In Ada3, the objective is to enable the model to refine its learned knowledge while also enhancing its robustness when dealing with more challenging examples. Therefore, we introduce difficulty to the data in a stochastic manner in this adaption layer.

Algorithm 2 SARA-Ada3 (Stochastic)

Input: original training data points D with number N
Initialize the stochastic decider s_3 and threashold T_3
for $i = 1, \ldots, N$ **do**
    sample the stochastic decider s from the range $(0, 1)$
    if $s_3 < T_3$ **then**
        select an extra distractor d_e
        Inject: $D[i] \Leftarrow D[i] + d_e$
    end if
    forward pass
    backward pass and update parameters
end for

As depicted in Algorithm 2, each data point has a certain probability of having an extra distractor added. Denoting the stochastic threshold in Ada3 as T_3, we have:

$$D_{\text{Ada3}}[i] = \begin{cases} D_{\text{Ada2}}[i] & \text{with probability } 1 - T_3 \\ D_{\text{Ada2}}[i] + Extra\ Distractor & \text{with probability } T_3 \end{cases} \tag{5}$$

Using the same reasoning as in Ada2, we conclude that each data point now has a probability of $1 - T_3$ of being classified as a residual learning data point. For these points, the model learns the residual of $\Delta W - \sum_{i=1}^{2} B_i A_i$ to make optimal weight updates, based on the previously learned low rank matrices $B_1 A_1$ and $B_2 A_2$ from Ada1 and Ada2, respectively. Data points with additional distractors injected, which effectively create new training scenarios, inherently incorporate elements of **data augmentation**, and can therefore be classified as augmented data points.

3.2 Framework of SARA-D.

SARA-D is a framework that we propose based on SARA, which uses **distraction** as the metric. The upper portion of Figure 3 stands for the architecture of RAFT, while the lower sections correspond to our proposed Stochastic Adaption framework.

In RAFT, each chain is trained on the complete training set, utilizing a fixed number of distractors. In the SARA-D framework, stochastic elements are introduced in both Ada1 and Ada3. Weight merging between Ada layers is also employed to consolidate knowledge and facilitate further residual learning.

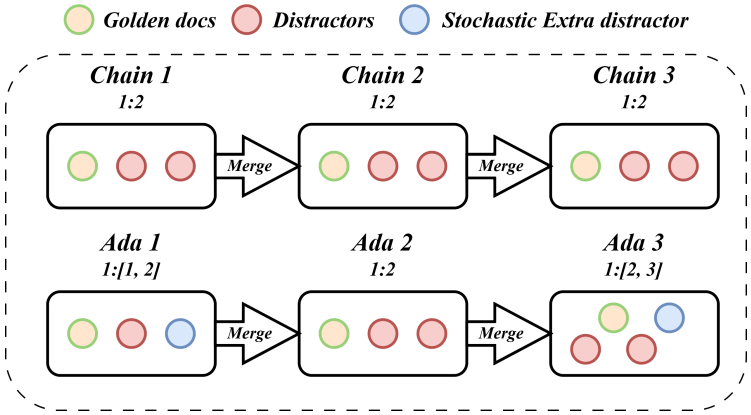

Fig. 3. Framework of SARA-D. Green circles indicate the golden documents, red circles indicate distractors, and blue circles represent extra distractors chosen in a stochastic manner. From our experiments, the reasonable values for the stochastic thresholds are $T_1 = 0.5$, $T_2 = 0$ and $T_3 = 0.15$. In Ada1 and Ada3, the notation $[1,2]$ and $[2,3]$ represents all potential values within those ranges, reflecting the inherent stochasticity of the process.

3.3 Framework of SARA-P

As mentioned earlier in RAFT, the proposed feature p (the **probability** of golden documents appearing in the context) can effectively help LLMs learn to filter out irrelevant information, preventing the model from generating arbitrary answers when the correct one is not available. We have fully considered and leveraged the role of p, using its value as a measure of the difficulty. Based on this, we construct another kind of SARA framework, **SARA-P**.

As shown in Figure 4, for the stochastic adaption layers under the SARA-P architecture, we apply an increasing difficulty with p values of 1, 0.6, and 0.2, with the stochastic threshold $T_2 = 0.4$ and $T_3 = 0.65$ in Ada2 and Ada3 respectively. For each Ada layer, the golden context of each data point is either kept or discarded in a stochastic manner. In contrast, for the control group RACoLA, we use fixed p = 0.6 for all three chains.

We draw an analogy to the human learning process in order to explain SARA-P. For beginners in a new field, we ensure that all the learning materials contain the correct answers, preventing them from

getting overwhelmed in early stages. Since the materials guarantee the inclusion of correct answers, this learning stage is relatively easy and requires only a small amount of study. In the first layer of SARA-P, the LLM is trained on the p = 1 materials just once. Then, the difficulty gradually increases with p values shifting to 0.6 and 0.2. As the model learns from these materials, its ability to filter out irrelevant information must strengthen. However, considering that in real-world scenarios, situations where correct answers are unavailable do exist but are not the norm, high-difficulty materials with p = 0.2 are studied less frequently. This ensures the model develops its question-answering capabilities more effectively in common scenarios where p = 0.6 is typical.

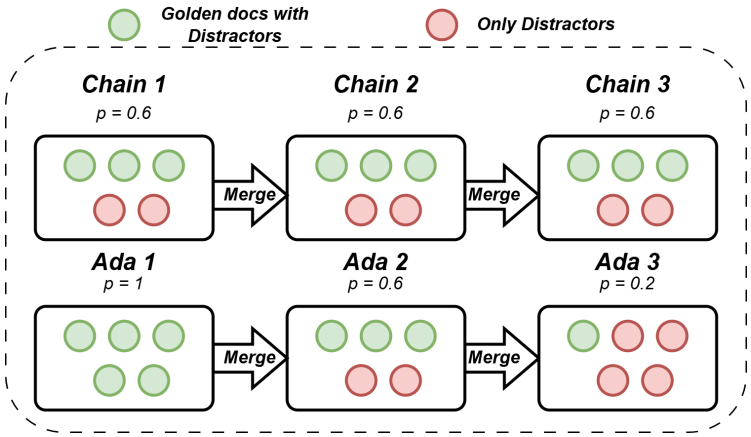

Fig. 4. Framework of SARA-P. Green circles indicate contexts containing golden documents, while red circles represent distractors. The color proportion reflects the corresponding p value. From our experiments, the reasonable values for the stochastic thresholds are $T_1 = 0$, $T_2 = 0.4$ and $T_3 = 0.65$.

4 Experimental Setup

4.1 Models and tasks

Models: We experiment with SARA to fine-tune Llama3.1-8B, a new and powerful model.

Tasks: We assess the effectiveness of our approach using PubMed [15], Hotpot [16] and the Huggingface API dataset [17]. Pubmed is a domain-specific (biomedical) context-based question answering dataset[18], while Hotpot is relatively general domain. The Huggingface dataset measure the model's ability to generate correct, functional and executable API calls based on given API documents.

Methods compared: In the current write-up, we primarily compare our approach with RAFT, which greatly enhances the model's retrieval ability. For future research, we plan to include additional baselines.

4.2 Implementation Details

We implement our method SARA as well as all experiments utilizing PyTorch, Transformers library [19] and Unsloth framework. The experiments are carried out on NVIDIA A100(40G) GPU, and the scores are compared within the same seed.

The training data size is chosen from {1k, 8k, 20k}, while the validation and test data sizes are each set to 25%. The p-value is chosen from {0.2, 0.6, 1}. We use AdamW_8bit as our base optimizer, and adopt a linear learning rate schedule with the initial learning rate chosen from {1e-5, 5e-5, 1e-4}. LoRA dropout, weight decay and early-stopping are enabled during all training process, aiming to keep the models with best performance, prevent over-fitting and record accurate training time for further comparison.

5 Results and Analysis

5.1 RAFT

Table 1: RAFT + LoRA compared with Llama3.1 and LoRA. With the assistance of retrieval information and LoRA, RAFT + LoRA obtains enormous improvement relative to the original Llama 3.1 and LoRA.

Models	F1	Gains relative to Llama 3.1
Llama3.1	66.3	—
LoRA	70.7	+ 6.6%
RAFT + LoRA	**74.4**	**+ 12.2%**

To evaluate the effectiveness and feasibility of RAFT, we compare it against the LLaMA 3.1 model fine-tuned using LoRA. We also employ the LoRA fine-tuning method when validating RAFT to ensure horizontal comparability. The results demonstrate a remarkable 12.2% improvement in performance for RAFT + LoRA compared to LLaMA 3.1, and LoRA also has a 6.6% performance boost over the Llama 3.1 baseline, as depicted in Table 1.

5.2 RACoLA

Table 2: Performance of RAFT + CoLA with different chain lengths. The score of RAFT + CoLA rises as the chain length increases.

Models	BLEU	Gains relative to RALoRA
RAFT + LoRA	47.09	—
RAFT+ CoLA (2 chains)	47.30	+ 0.45%
RAFT + CoLA (3 chains)	**47.46**	**+ 0.79%**

Based on the improvement achieved by RAFT, we move forward with integration of RAFT with Chain of LoRA (CoLA).

We examine how the number of chains in RAFT + CoLA affects the model's performance [10]. We reproduce the experiments in the CoLA paper and obtain consistent results that the chain length of CoLA is positively correlated with the model capability, even after RAFT is integrated. RAFT + CoLA with 3 chains outperforms RAFT + LoRA model by 0.79% and has the best performance, as displayed in Table 2.

5.3 SARA

In this section, we compare our method SARA with RAFT (on top of CoLA with three chains), the best previous baseline, to demonstrate further improvement in performance, robustness and training efficiency. We also provide Llama3.1 for reference and include ablation studies on hyperparameters of SARA. We use the harmonic mean of RougeLsum precision and recall as the evaluation metric for PubMed and HotPot, which we denote as F1. For API tasks, we compare the models' functionality accuracy and hallucination. We adopt the **oracle + distractors** pattern in all test datasets. In subsequent sections, any mention of RAFT will specifically refer to RAFT implemented using CoLA.

5.3.1 The SARA-D Framework

We first implement SARA using SARA-D framework. All the training data are designed as $p = 1$ for fair comparisons.

Table 3: Performance comparison of Llama3.1, RAFT, and SARA across PubMed, HotPot, and Huggingface datasets. SARA consistently outperforms RAFT accross various tasks.

	PubMed	HotPot	Huggingface	
Models	F1	F1	Accuracy	Hallucination
Llama3.1	47.11	40.25	37.28	0.3042
RAFT	71.58	51.24	76.43	0.0638
SARA	**72.46**	**51.37**	**80.31**	**0.0542**
SARA vs. Llama3.1	+ 53.81%	+ 27.63%	+ 115.42%	- 82.18%
SARA vs. RAFT	+ 01.23%	+ 00.25%	+ 05.08%	- 15.05%

As shown in Table 3, using the SARA-D framework, our SARA model consistently surpasses the baseline models. Compared to the base Llama 3.1 model, the improvements are substantial, reaching up to 53.81% on HotPot QA and 115.42% on Huggingface evaluation, along with an 82.18% reduction in hallucination.

Even against RAFT, SARA demonstrates superior performance, achieving a 5.08% increase in functionality accuracy and a 15.05% further reduction in hallucination on the Huggingface evaluation. Note that for HotPot QA, since SARA is not specifically optimized for chain-of-thought (CoT)

reasoning for the current write-up, we don't observe significant improvements when comparing our SARA model with RAFT.

5.3.2 The SARA-P Framework

We then implement SARA using SARA-P framework. All the training data are designed as $p = 0.6$ for fair comparisons.

Table 4: Implemented using SARA-P, SARA still consistently outperforms RAFT.

	PubMed	HotPot	Huggingface	
Models	F1	F1	Accuracy	Hallucination
Llama3.1	47.11	40.25	37.28	0.3042
RAFT	71.61	51.24	78.87	0.0542
SARA	**72.02**	**51.52**	**82.08**	**0.0442**
SARA vs. Llama3.1	+ 52.88%	+ 28.00%	+ 120.17%	- 85.47%
SARA vs. RAFT	+ 00.57%	+ 00.55%	+ 04.07%	- 18.45%

As shown in Table 4, the base Llama 3.1 model achieves scores of 47.11 on PubMed and 40.25 on HotPot. RAFT improves these results to 71.61 and 51.24, respectively. SARA further enhances performance and reaches a 52.88% improvement over Llama 3.1 on PubMed and a 28.00% increase on HotPot. When compared to RAFT, SARA shows a modest yet consistent improvement of 0.57% on PubMed and 0.55% on HotPot.

For the API dataset, SARA achieves a 120.17% increase over Llama3.1 and a 4.07% gain over RAFT on functionality accuracy. Hallucination also drops significantly, with SARA reducing hallucinations by 85.47% compared to Llama3.1 and 18.45% compared to RAFT (Table 4).

5.3.3 Training cost of SARA

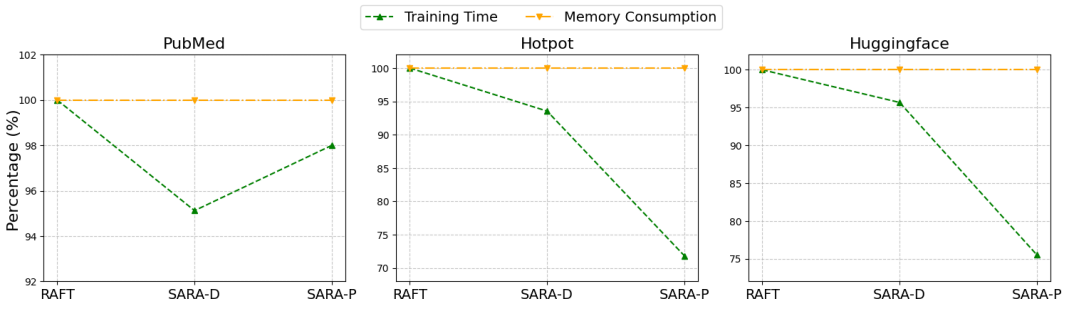

Fig. 5. Training time and memory consumption comparison between SARA frameworks and RAFT.

In addition to performance improvements, we also analyze the computational costs of SARA.

No additional memory needed: As illustrated in Figure 5, no extra memory is required during training, since the LoRA ranks r controls the tunable model parameters and are set the same.

Improved training efficiency: Both implementations of SARA demonstrate reduced training time. On PubMed, the reduction in training time is modest but consistent, with SARA-D requiring 95.12% and SARA-P 98% of the time needed by RAFT. On the HotPot QA and Huggingface API datasets, SARA-D demonstrates roughly a 5% reduction in training time. However, SARA-P significantly reduces training time, requiring only 71.83% on HotPot and 75.49% on Huggingface compared to RAFT, as early stopping is triggered during the last adaption stage Ada3.

These results highlight that SARA not only enhances model performance, but also accelerates convergence during the training process.

5.3.4 Ablation Study

The level of stochasticity to introduce: As described in Section 3.1, different stochastic thresholds $T \in [0,1]$ are adopted in different stages of Adas. Specifically, we study the stochastic threshold T_3 in SARA-D, which regulates the extent of augmentation in Ada3.

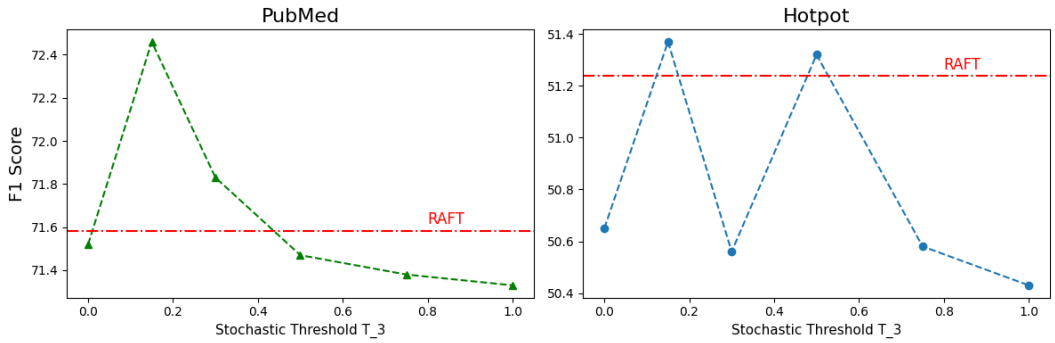

Fig. 6. SARA-D model performance under different values of T_3.

$T_3 = 0$ signifies that no stochasticity is introduced, while $T_3 = 1$ represents the scenario in which all data points are augmented. The horizontal baselines are set to represent the performance of RAFT. From Figure 6, we consider the optimal value for T_3 to be around 0.15, where the SARA model achieves its best performance on both the PubMed and Hotpot QA datasets. Ablation on HotPot QA also reveals a twin peak around $T_3 = 0.5$, suggesting that the model may benefit from the additional distraction, and potentially enhancing its ability to distinguish relevant information from distractors in highly noisy environments like HotPot QA.

6 Conclusions and Future Work

In this work, we present the Stochastic Adaptation of Retrieval-Augmented (SARA) to enable more efficient and effective fine-tuning of large language models, whose innovation is to introduce stochastic adaptation layers to enhance the retrieval ability of models. Current experimental results show that SARA consistently outperforms previous baselines with less training cost.

We are actively working on applying SARA across various domains of data and large language models (LLMs). We are also developing adaptation layers tailored to various metrics, including the complexity of chain-of-thought (CoT) reasoning.

Acknowledgments

Each co-author—Peiyu Xu, Naxin Chen, Xinyu Liu, and Denhao Peng—has significantly contributed to the conception, design, analysis, and writing of this paper. All five contributors should be considered co-first authors. This collective effort reflects our shared commitment to advancing research on enhancing the ability of LLMs to respond to domain-specific queries.

References

[1] Laranjo, L., Dunn, A. G., Tong, H. L., Kocaballi, A. B., Chen, J., Bashir, R., ... & Coiera, E. (2018). Conversational agents in healthcare: a systematic review. *Journal of the American Medical Informatics Association*, 25(9), 1248-1258.

[2] Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., ... & Wen, J. (2024). A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6), 186345.

[3] Min, B., Ross, H., Sulem, E., Veyseh, A. P. B., Nguyen, T. H., Sainz, O., ... & Roth, D. (2023). Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2), 1-40.

[4] Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., ... & Poon, H. (2021). Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1), 1-23.

[5] Patil, R., & Gudivada, V. (2024). A review of current trends, techniques, and challenges in large language models (LLMs). *Applied Sciences*, 14(5), 2074. MDPI.

[6] Ding, B., Qin, C., Zhao, R., Luo, T., Li, X., Chen, G., ... & Joty, S. (2024, August). Data augmentation using LLMs: Data perspectives, learning paradigms and challenges. In *Findings of the Association for Computational Linguistics ACL 2024* (pp. 1679-1705).

[7] Wang, X., Chen, Y., & Zhu, W. (2021). A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence, 44*(9), 4555-4576.

[8] Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., ... & Li, Q. (2024, August). A survey on RAG meeting LLMs: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (pp. 6491-6501).

[9] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... & Chen, W. (2021). Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

[10] Xia, W., Qin, C., & Hazan, E. (2024). Chain of lora: Efficient fine-tuning of language models via residual learning. *arXiv preprint arXiv:2401.04151*.

[11] Wu, T., He, S., Liu, J., Sun, S., Liu, K., Han, Q. L., & Tang, Y. (2023). A brief overview of ChatGPT: The history, status quo and potential future development. *IEEE/CAA Journal of Automatica Sinica*, 10(5), 1122-1136.

[12] Zhang, T., Patil, S. G., Jain, N., Shen, S., Zaharia, M., Stoica, I., & Gonzalez, J. E. (2024). RAFT: Adapting Language Model to Domain Specific RAG. *arXiv preprint arXiv:2403.10131v2*.

[13] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., ... & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems, 35*, 24824-24837.

[14] Ding, N., Qin, Y., Yang, G., Wei, F., Yang, Z., Su, Y., & Sun, M. (2023). Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence, 5*(3), 220-235.

[15] Jin, Q., Dhingra, B., Liu, Z., Cohen, W., & Lu, X. (2019, November). PubMedQA: A Dataset for Biomedical Research Question Answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (pp. 2567-2577).

[16] Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W. W., Salakhutdinov, R., & Manning, C. D. (2018). HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[17] Patil, S. G., Zhang, T., Wang, X., & Gonzalez, J. E. (2023). Gorilla: Large language model connected with massive APIs. *arXiv preprint arXiv:2305.15334*.

[18] Kaddari, Z., Mellah, Y., Berrich, J., Bouchentouf, T., & Belkasmi, M. G. (2020, October). Biomedical question answering: A survey of methods and datasets. In *2020 Fourth International Conference On Intelligent Computing in Data Sciences (ICDS)* (pp. 1-8). IEEE.

[19] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Rush, A. M. (2020, October). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations* (pp. 38-45).