KBAlign: Efficient Self Adaptation on Specific Textual Knowledge Bases

Zheni Zeng[1*], Yuxuan Chen[2*], Shi Yu[3], Ruobing Wang[6], Yukun Yan[3✉], Zhenghao Liu[7], Shuo Wang[3], Xu Han[3], Zhiyuan Liu[3,4,5], Maosong Sun[3,4,5]
[1]Nanjing University [2]Peking University [3]Tsinghua University
[4] Beijing National Research Center for Information Science and Technology
[5] Institute of Artificial Intelligence, Tsinghua University
[6]University of Chinese Academy of Sciences [7]Northeastern University
yanyk.thu@gmail.com

Abstract

Although retrieval-augmented generation (RAG) remains essential for knowledge-based question answering (KBQA), current paradigms face critical challenges under specific domains. Existing methods struggle with targeted adaptation on small-scale KBs: vanilla unsupervised training exhibits poor effectiveness, while fine-tuning incurs prohibitive costs of external signals. We present KBAlign, a self-supervised framework that enhances RAG systems through efficient model adaptation. Our key insight is to leverage the model's intrinsic capabilities for knowledge alignment through two innovative mechanisms: multi-grained self-annotation that captures global knowledge for data construction, and iterative tuning that accelerates convergence through self verification. This framework enables cost-effective model adaptation to specific textual KBs, without human supervision or external model assistance. Experiments demonstrate that KBAlign can reserve 90% of the performance gain obtained through GPT-4-supervised adaptation, while relying entirely on self-annotation of much smaller models. KBAlign significantly improves downstream QA accuracy across multiple domains with tiny costs, particularly benefiting scenarios requiring deep knowledge integration from specialized corpora. We release our experimental data, models, and process analyses to the community for further exploration [1].

1 Introduction

Large language models (LLMs) have demonstrated their general capabilities across a wide range of downstream tasks (Achiam et al., 2023), and the factual reliability of the models could be enhanced with common techniques such as retrieval augmented generation (RAG) (Lewis et al., 2020).

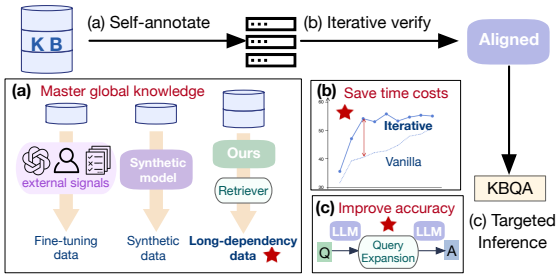

Figure 1: KBAlign schematic. We design special self-annotation methods to help master global KB knowledge, conduct iterative verifying to save training time costs, and adopt targeted inference to improve accuracy.

When applied in specific domains, however, the adaptation of models to knowledge base (KB) materials remains a crucial strategy to further improve performance (Ling et al., 2023). Intuitively, adapted models align with the knowledge scope of the given KBs, and can directly memorize some domain information or better utilize the retriever through query rewriting. For example, both general QA model and the retriever may regard "LLM" as an AI term, while the legal aligned model can generate disambiguated context in which "LLM" stands for master of laws.

Existing adaptation methods usually construct domain models with large-scale training data (Zhao et al., 2024), while there are quite specific needs in real-world scenarios corresponding to small-scale textual KBs, such as providing customized services based on user-specific document repositories, or plug-and-play integration of modules. In these cases, simple LM training on raw data may degrade performance, necessitating alternative approaches to align with specific domains (Cheng et al., 2023). Targeted fine-tuning, on the other hand, typically involves the incorporation of external knowledge signals (Tan et al., 2024) to transform data into more structured tasks. When faced with constraints such as confidentiality, convenience, and limited

[1] https://github.com/thunlp/KBAlign

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 13519–13532
November 4-9, 2025

computational resources, involving human annotation or relying on online large models becomes unpractictible. Therefore, a low-cost adaptation to small-scale KBs that does not rely on external supervisions is urgently needed.

Drawing an analogy to the human learning process, RAG is similar to open-book tests in which students could query KB materials. If they conduct self-study in advance, quickly grasping the fundamental content of the books themselves, the effectiveness and efficiency in tests can be improved. Based on this idea, we purpose KBAlign, a highly efficient self-adaptation approach tailored to specific textual KBs comprising self-improvement learning. Generally, we align the model with the small domain in a highly efficient and completely self-supervised manner. As shown in Fig. 1, for **self annotation**, we organize the original KB materials in multiple grains and conduct self annotation to get instruction-response pairs that can cover various downstream task scenarios. For **iterative tuning**, we require the model to check its own responses and help modify common mistakes in the current stage for a faster convergence. Meanwhile, we conduct **targeted inference** in which strategies such as query expansion and confidence verification are adopted to refine the response.

Experiments on fact QA, long-form QA, and professional field test have shown the effectiveness of our method across different backbone models. With a low cost, KBAligned models generally master the domain knowledge, memorize part of detailed information in KB, and also obtain some capability of better utilizing retrieved context. Side experiments including ablation studies and performance curve analyses identify the most efficient self-annotated data amount and optimal training volumes, offering valuable guidance for effectively applying KBAlign in practical scenarios.

Our main contributions are as follows: (1) We propose KBAlign, a novel method for autonomous LLM adaptation tailored to textual KBs. It helps LLMs perform KB adaptation relying entirely on self annotations; (2) We provide empirical insights into efficient self-adaptation to KBs, offering practical parameters and settings for deploying KBAlign; (3) We conduct a comprehensive analysis of the proposed self-adaptation framework. Through a range of evaluation metrics and case studies, we identify the effectiveness of KBAlign and discuss the current limitations of our approach, highlighting areas for future improvement.

2 Related Work

Domain Adaptation.Though LLMs have shown their impressive capabilities in various scenarios (Jablonka et al., 2023), training methods for LLMs to adapt to certain domains still emerge in endlessly, due to the vertical application requirements. For domains with plenty of data resources, researchers directly take domain materials in pretraining (Wang et al., 2023a; Madani et al., 2023). In more cases, they continue to train based on general LLMs or mix domain data with the general corpus (Wu et al., 2023). To adopt domain knowledge in a more efficient way, format conversion and annotation are often performed (Zhang et al., 2024a) for fine-tuning. Some works focus on different settings for synthetic generation of QA data (Heydar et al., 2024; Ushio et al., 2023), while with the development of annotation model capabilities, the impact of specific synthetic strategies diminishes significantly. More crucially, existing approaches predominantly focus on local information and ignore global knowledge in synthesis.

Knowledge Enhancement. For some specific downstream requirements, there often exist high-quality knowledge materials (e.g., domain KBs, personal documents or records), of which the data amount is not enough for model tuning, and knowledge enhancement methods can help improve the performance. There are two mainstream solutions. The first one is to rely on the strong in-context learning capability of LLMs (Dong et al., 2022), and adopt RAG (Lewis et al., 2020) to enhance the model. Apart from textual materials such as Internet passages, it is proven that integrating special KBs and tools is also a good approach to improve the model performance with specific knowledge (Cui et al., 2023; Jin et al., 2024; Qin et al., 2023). To provide more useful information in context, strategies including designing better queries for retrieval are proposed (Wang et al., 2023b; Qian et al., 2024). The second solution is to augment training data based on knowledge materials. LLMs can help synthesize data in more styles (Sun et al., 2023) or convert the original data into formats more suitable for training (Cheng et al., 2023). Our method is special, emphasizing the self annotation instead of introducing new LLMs into the system.

Self Improvement. There are some works exploring the self-improve capability of LLMs, most of which focus on the automatic generation and selection of reasoning steps for existing answers,

being helpful in tuning (Huang et al., 2023a) and inference (Jiang et al., 2023). Self-play fine-tuning in an iterative manner (Chen et al., 2024) also unlocks the full potential of golden data. Even without the ground-truth answers, intern consistency of LLMs can be adopted as an important supervision signal that can achieve improvement (Liang et al., 2024). Nevertheless, challenge remains for human-like self improvement, such as how to self correct the reasoning process (Huang et al., 2023b). We observe the human learning process and design corresponding self-improvement methods.

3 Methodology

3.1 Task Setup

We define KB adaptation as the process in which, given a knowledge base K (textual materials in our case), an original generative model M, and a retriever R for RAG, the goal is to efficiently align the models with the information in K without any external signals, thus improving the knowledge-intensive tasks based on K. The optimization objectives are to maximize the performance scores in downstream tasks while minimizing training costs.

There are two common approaches utilizing K to enhance model performance: tuning-based and inference-based methods. Tuning-based methods involve generating tuning data of M from K using unsupervised techniques, or designing specific R for the current domain which is not covered in this work. Inference-based methods, on the other hand, focus on optimizing the retrieved content in the basic RAG setups, or post-processing the generated results to enhance relevance and accuracy. We now introduce our method which combines unsupervised tuning and RAG improvement, optimizing both the tuning and inference approaches. Examples are shown in Fig. 2.

3.2 Self Annotation

To learn the knowledge from KBs without any supervised data, we conduct self annotation with the backbone model M on the K text. To be specific, we choose a paragraph of golden context C_g as the knowledge source and require M to raise a set of questions Q. We then supplement the related context C_R by the retriever R, and ask M to annotate the answers A based on $C_g + C_R$. When answering the questions with RAG, M sometimes fails due to the vague context provided by R; while in the annotation process, A is comparably precise because of the ensured existence of C_g and our handcrafted keyword filters (e.g., questions should not mention pronouns such as "in this paragraph").

Owing to the diverse forms and attributes of K and associated downstream tasks, we propose multi-grained annotation corresponding to different organization strategies for C. The detailed process is shown in algorithm 1.

Short-dependency Annotation. For downstream tasks that prioritize precise fact knowledge expressed in one specific paragraph, we employ this approach to simply divide K into fixed-length chunks, each with no more than $1,024$ words while keeping continuity of information across boundaries. One chunk is adopted directly as the annotation context C_g.

Long-Dependency Annotation. Considering that real-world tasks often require a comprehensive understanding of multiple pieces of information at long distances in text, we design long-dependency annotation methods that split K into shorter segments with less than 256 words. Several segments $S_{1,\dots,n}$ with the same hierarchical directory, or with the highest embedding similarities across different directories, are concatenated as C_g. When generating Q, the model is required to raise questions that: (1) involve knowledge from different segments to emphasize the multi-hop reasoning capability; (2) are as vague as possible, corresponding to a series of information $I_{1,\dots,n}$ annotated on S, based on which a refined long-form answer A is generated to improve the integration capability.

3.3 Iterative Tuning

Apart from summarizing and self-questioning to help deep understanding, human students also take tests at each learning stage and strengthen the knowledge they have not yet mastered by correcting their answers. Similarly, we hope that the model can improve itself through self-verification in addition to understanding. The detailed process is also provided in algorithm 1.

Initial Tuning. With the self-annotation <Q, A> data, we tune M to get an initially adapted version. Due to the limitations of the retriever, the retrieved C_R in test scenario may differ from the annotation context C (in which the golden paragraph C_g must be included). Therefore, we randomly concatenate either C or C_R with the question Q as the input. This mixed paradigm aims to bridge the gap between tuning and inference.

Self-Verify Tuning. We divide the annotated

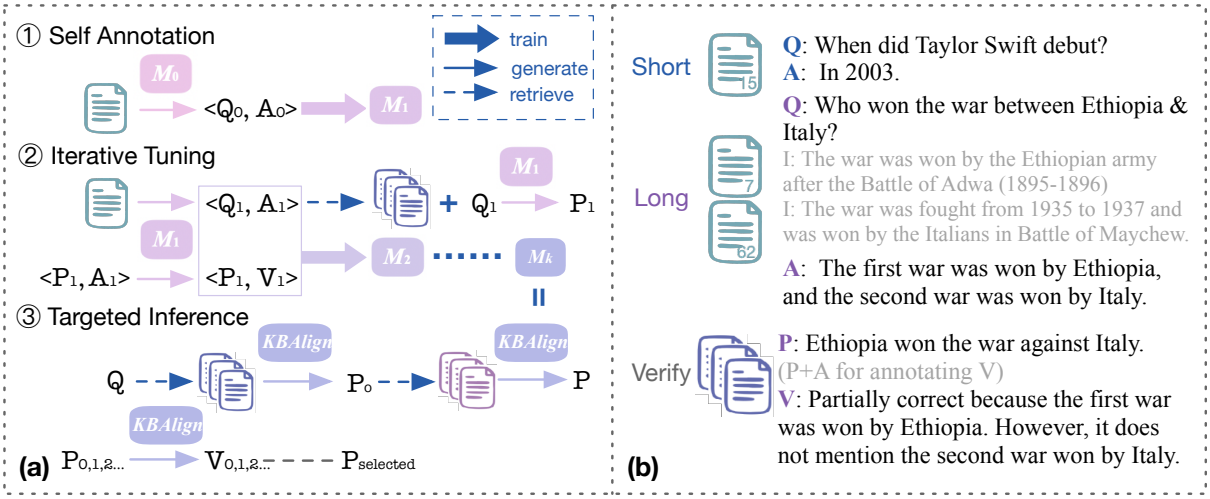

Figure 2: (a) Details for the KBAlign framework; (b) Instances for different annotation strategies and tasks.

data into k parts <$Q_{1,2,...k}$, $A_{1,2,...k}$>, and adapt the model with the first part <Q_1, A_1> to get the initial version M_1. Using this model, we perform RAG inference in the second part Q_2 to obtain the predicted answer P_2 reflecting current capability of M_1. Given the ground-truth answer A_2, the model verifies its own prediction and analyzes the wrong reason, which we name V_2. In the next stage, we can then use <Q_2, P_2> as input and V_2 as output to continue tuning M_0. And so on, we generate the verification data based on current performance, and conduct the Q&A task and verify task at the same time in an iterative manner. In experiments, we use 25% of verify and 75% of Q&A, and implement 2-3 iterations.

3.4 Targeted Inference

We improve the downstream performance mainly by training the model to learn more specific knowledge. We also employ **Query Expansion** (QE) to refine the retrieval results in reference stage. To be specific, directly applying Q as the search query may miss useful information due to the short expression and the limitation of the retriever R. Considering that our model has memorized the overall knowledge, it can provide a prediction P that is relevant to the KB content. We then expand the search query as Q+P, and this may help make the retrieval results much better.

The other strategy that can be used in reference is **Self Verification**, which is based on the capabilities learned in iterative verifying. For the generated P, the model can check the correctness by itself. It should be emphasized that this is not the standard strategy setting in subsequent experiments, because it will increase the time cost, and it is also difficult for the model to correct the error after realizing it. However, the model can at least provide an uncertainty warning, or sample a new response when the confidence score is low when needed, which helps improve reliability.

4 Experiment

4.1 Datasets and Models

In order to evaluate the effectiveness of our method as comprehensively as possible, we use four datasets in the experiment, each could form a corresponding KB (from 0.41 to 21 M tokens). Details are displayed in section A.

LooGLE (Li et al., 2023). This is a long-text dataset, with textual materials that can be regarded as KBs and high-quality questions. We evaluate the specific knowledge memorizing capability of the model in this dataset.

ASQA (Stelmakh et al., 2022). This is a long-form QA dataset. We evaluate the knowledge recall and organizing capability of the model in test set, and do not use any training data from it.

JEC-QA (Zhong et al., 2020). This is a legal multiple choice dataset in Chinese. We evaluate the professional learning capability and instruction following in different inference formats.

BioASQ [2]. This is a biomedical question answering dataset. We evaluate the model's biomedical knowledge retrieval and reasoning capabilities.

We choose the following models as the backbone and comparison objects of the experiments:

[2] https://huggingface.co/datasets/kroshan/BioASQ

MiniCPM (Hu et al., 2024). This refers to MiniCPM-2B which is one of the backbone models in our experiment. It is an end-side LLM gaining the instruction following ability during pre-training, and has achieved the best performance among lightweight LLMs on several datasets. Therefore, we believe that it has wide personal applications and is suitable for efficient adaptation scenarios.

LLaMA-3.1. This refers to LLaMA-3.1-8B-Instruct [3] which is aligned from one of the most popular open-source model families. We choose it to evaluate whether our method is universally helpful when the backbone model becomes stronger.

GPT series. GPT-3.5 refers to GPT-3.5-turbo-0125 [4] which is a representative closed-source LLM with stable performance and comprehensive capability. GPT-4o is an even stronger LLM.

LM. This represents directly conducting the language modeling task to align the model with KBs. Knowledgeable text is segmented into 512-token-length paragraphs, and mixed with general instruction tuning data (Ding et al., 2023) to keep the instruction following capability.

RAFT (Zhang et al., 2024b). This represents adapting language models to domain-specific RAG. Follow this method, we use GPT-4o to annotate data from the KBs, generating both Chain-of-Thought(CoT) reasoning and final answers to help the model focus on useful information while disregarding distractors. The annotated data used for supervised fine-tuning (SFT). We adopt this method as one of the baselines in our experiments.

4.2 Evaluation Metrics

For LooGLE, ASQA and BioASQ, we consider the evaluation of the original dataset and decide to utilise the following metrics: (1) **Rule metrics**: F1 score, which measures the harmonic mean of precision and recall; Match score, which measures the recall of key elements in long-form answer; For JEC-QA, only precise prediction of options could be scored, regardless of whether the questions were single or multiple-choice.

(2) **Similarity metrics**: BERT score (Zhang* et al., 2020) calculates cosine similarity to assess semantic consistency, utilizing embeddings generated by the text2vec (Xu, 2023) model from sentences; BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), which are traditional text generation similarity metrics provided in ablation studies.

(3) **Intelligent metrics**: semantic judgment by the representative OpenAI LLM, GPT-4o, is used to evaluate the quality of responses further. Detailed prompts are provided in Section A.

4.3 Other Settings

We tune all parameters of MiniCPM, while conduct a parameter-efficient tuning for LLaMA-3.1, utilizing the LoRA (Hu et al., 2021) strategy to reduce the need for computing power costs. In the test scenario, the chunks of KB materials are divided with less than 128 tokens, and the top 8 relevant chunks are provided.

Hyper-parameters, retrieval and speed-up settings are provided in Section A.

4.4 Result Analysis

Time Costs. We first estimate the time cost to prove the efficiency of our method. We provide the result on ASQA after scaling to the capacity of an A100 GPU: short-dependency annotation for 1k data items takes 30 min, long-dependency annotation for 1k data items takes 140 min, and iterative tuning process takes 160 min. Comparably, direct language modeling training takes 480 min, which is longer than the whole process of KBAlign. As for RAFT, it involves larger models and longer CoT responses requiring more annotation time, and the tuning time is controlled to be the same with us.

Main Experiments. Results for our experiments are shown in Table 1. We provide the GPT-series results, the initial version and the self-adapted version of both MiniCPM-2B and LLaMA-3.1-8B-Instruct on the four dataset. Overall, comparing the "Ours" lines with the corresponding vanilla RAG, we can see that KBAlign ensures an obvious improvement on most of the metrics, regardless of the dataset and the backbone model, showing its generalization and effectiveness.

The simple language modeling helps align the model to several KBs and gets marginal promotion, while not always effective, further proving the necessity of self annotation. The advanced baseline RAFT relies on the quality of CoT reasoning, which requires quite large amount of training. When aligned with our high-efficient training setting, its effect is not always obvious.

Task Differences. Nevertheless, our strategies still produce differentiated effects in the four scenarios. For LooGLE which evaluates the master

[3] https://huggingface.co/meta-llama/Llama-3.1-8B

[4] https://platform.openai.com/docs/models

Methods	LooGLE			ASQA			JEC-QA			BioASQ		
	F1	BERT	LLM	Match	BERT	LLM	Single	Multi	Total	F1	BERT	LLM
GPT series												
GPT-3.5	35.42	80.99	78.08	26.79	86.61	51.66	14.49	17.92	16.32	17.80	80.55	93.85
w/o QE	35.27	80.94	77.91	27.40	86.71	51.52	13.84	19.15	16.68	18.55	80.57	93.23
GPT-4o	40.20	81.70	82.93	**32.18**	**87.14**	67.76	21.95	26.42	24.33	31.73	81.31	94.15
w/o QE	40.21	81.71	**83.29**	32.15	87.10	**67.88**	20.11	**27.36**	23.98	31.39	80.83	94.46
MiniCPM-2B												
Vanilla RAG	30.92	80.70	64.76	11.91	82.30	22.92	39.24	13.87	25.69	29.27	82.37	84.92
w/o QE	30.31	80.37	64.72	12.37	82.90	22.42	38.38	14.06	25.39	30.23	82.71	83.69
RAFT	44.36	84.05	70.73	12.03	85.94	16.18	17.30	14.06	15.57	6.66	81.96	89.23
LM	50.15	84.77	65.62	10.72	81.27	21.03	47.36	7.98	23.73	55.44	88.62	81.85
Ours	54.09	86.48	75.19	15.68	85.41	24.81	**49.95**	9.94	28.91	61.38	89.95	87.69
Δ	(+23.17)	(+5.78)	(+10.43)	(+3.77)	(+3.11)	(+1.89)	(+10.71)	(-3.93)	(+3.22)	(+32.11)	(+7.58)	(+2.77)
w/o QE	53.76	86.23	73.19	16.12	85.48	25.69	49.41	10.92	**29.16**	61.91	89.91	89.54
LLaMA3.1-8B-Instruct												
Vanilla RAG	40.46	81.57	77.15	20.21	84.93	37.28	22.70	24.66	23.73	27.96	81.55	92.62
w/o QE	39.94	81.50	77.08	20.03	85.14	35.64	22.92	24.07	23.53	27.74	81.66	91.08
RAFT	42.13	84.88	77.91	23.42	85.92	38.74	23.24	15.47	19.09	44.94	83.36	93.54
LM	54.06	85.53	78.58	19.07	82.52	38.04	20.40	9.57	13.90	56.28	88.02	90.15
Ours	**62.07**	**88.63**	80.16	25.23	86.29	42.44	34.59	14.13	23.83	70.97	92.06	93.54
Δ	(+21.61)	(+6.06)	(+2.85)	(+5.02)	(+1.36)	(+5.16)	(+11.89)	(-10.53)	(+0.10)	(+43.01)	(+10.51)	(+0.92)
w/o QE	61.79	88.55	79.96	25.56	86.89	41.31	34.16	14.42	23.78	**73.30**	**92.72**	**94.48**

Table 1: KBAlign adaptation experiments on LooGLE, ASQA, JEC-QA and BioASQ. We report average for 3 random seeds.

of precise local knowledge, self-annotated tuning brings a huge improvement (over 20% on F1) and the adapted 2B model can surpass LLaMA-3.1-8B & GPT-4o performance. For ASQA emphasizing long-form answer that covers global information, however, the improvement is comparably marginal (less than 5% on Match). The first possible reason is that backbone models have already mastered the WikiPedia knowledge in pre-training, and extra adaptation is redundant. The second reason is the over emphasis of local knowledge in the responses, making QE strategy provide even more limited retrieval results.

The same trend is also reflected in the single-choice and multiple-choice tests of JEC-QA. Our method easily surpasses some legal-domain models in the former (such as 40.8 single-choice score reported for 7B legal LLM (Wan et al., 2024)), while in the latter the performance even declines slightly due to reasons such as the output format. This indicate the challenge of learning knowledge with a long information span and logical chain.

Numerical Analysis. We search the best values for key settings including the training steps, amount of data and iteration by evaluating checkpoints in process. From Fig. 3 we can see, when learned on more QA pairs (only once), scores on LooGLE F1 (represents fact accuracy) for both backbone models improve. Interestingly, directly learning without iterative tuning (dotted curve) also displays a similar trend, while the tipping point for slowing

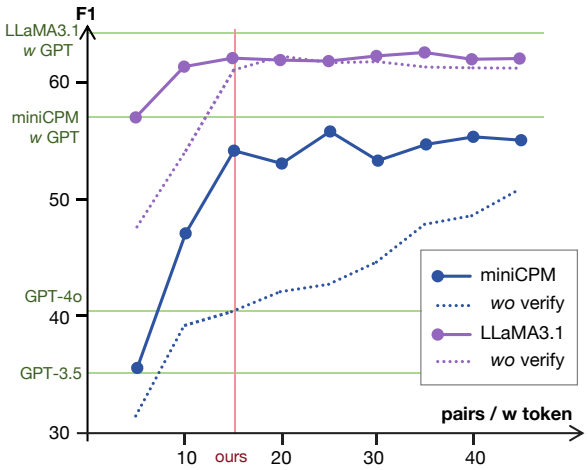

Figure 3: The impact of training amount on LooGLE performance. 'w GPT' refers to training with GPT-annotated data.

growth comes much later. This reveals the possible mechanism of self-verify task, that is, to guide the model to focus more on the problems of current stage, so as to reach convergence faster. According to the curve, we choose to provide 15 data items per 10,000 tokens for LooGLE training, and increase the data density of ASQA due to the smaller KB scale. Besides, although the tuning phase usually reuses the same data for multiple epochs of training, we observe from Fig. 4 that using half of the data to tune 2 epoch brings a quite obvious score decrease. Consider that the inference time for data annotation is acceptable compared with the training time, we

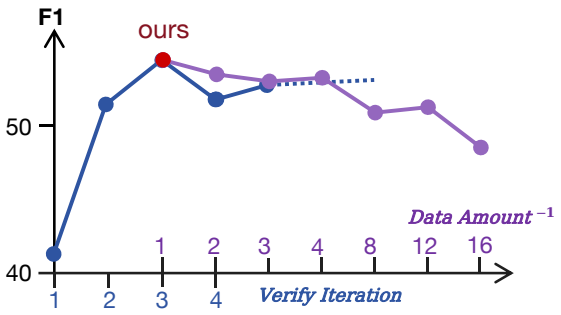

Figure 4: The impact of iteration times and data amount for fixed training steps on LooGLE performance.

recommend annotating more data and tuning with only 1 epoch.

From Fig. 4 we can also observe the performance change for different number of iterations when conducting iterative tuning. With the self-verify data, the score first increases and then keeps comparably stable as the iterations increases, showing that the verification capability helps improve model performance on downstream QA, while requiring the data quality to be high enough with a certain granularity. From the curve we recommend conducting at least 3 iterations, while it depends on actual situation in practical implementation.

Methods	F1	BLEU	ROUGE	BERT	LLM
MiniCPM-2B					
GPT Data	**56.92**	**20.72**	**52.30**	**87.16**	72.83
Ours	54.09	18.32	49.75	86.48	**75.19**
w/o know	52.16	14.71	47.85	85.75	69.35
w/o RAG	15.40	1.66	15.16	73.54	11.64
w/o QE	53.76	18.55	49.61	86.23	73.19
w/o verify	42.69	15.95	39.57	82.74	72.37
Llama3.1-8B-Instruct					
GPT Data	**64.97**	**26.41**	**59.85**	**89.56**	**80.21**
Ours	62.07	21.73	57.34	88.63	80.16
w/o know	58.32	21.22	52.32	87.12	78.31
w/o RAG	14.75	0.70	14.22	74.1	15.89
w/o QE	61.79	21.60	57.09	88.55	79.96
w/o verify	61.76	20.94	57.20	88.53	77.81

Table 2: Detailed results on LooGLE.

Ablation Study. We assess the effectiveness of our strategies by side experiments in LooGLE and ASQA, and provide the results in Table 2 and 3. To validate the quality of self annotated data, we try to replace the annotation model with GPT-4-turbo ("GPT Data") in LooGLE, and further replace the data with golden training set ("Golden") in ASQA. We find that the eventual scores are not much higher than current setting, especially when compared with the vanilla setting without adaptation, proving the usefulness of self annotation.

Methods	Match	BLEU	ROUGE	BERT	LLM
MiniCPM-2B					
Golden	**18.90**	**4.39**	**26.89**	**88.16**	23.43
Ours	15.68	2.67	24.59	85.41	**24.81**
w/o long	13.34	1.18	21.26	81.91	24.32
w/o verify	14.28	2.22	24.32	84.02	23.20
Llama3.1-8B-Instruct					
Golden	**28.41**	**3.95**	**26.75**	**88.30**	**44.08**
Ours	25.23	3.43	23.59	86.29	42.44
w/o long	20.45	0.64	17.29	79.85	43.95
w/o verify	24.88	2.18	22.01	83.32	35.14

Table 3: Detailed results on ASQA.

Methods	F1	BLEU	ROUGE	BERT
MiniCPM-2B				
Vanilla RAG	32.03	1.64	26.88	78.95
w/o RAG	21.34	0.11	16.67	75.27
Ours	**37.70**	**1.86**	**34.08**	81.24
w/o RAG	26.62	0.77	23.24	78.17
w/o know	36.22	1.85	32.96	**81.91**

Table 4: Detailed results on WebASQ.

By conducting ablation study on ASQA, we prove that the long-dependency annotation ("Ours" vs. "*w/o* long") plays a vital role, in which the comprehensive responses are expected. Considering the higher time cost (about 4 times of short-dependency), we discard this strategy to the local QA task in LooGLE. Meanwhile, self-verify tuning ("Ours" vs. "*w/o* verify") also helps improve the performance for both dataset by correcting errors of the current stage in a targeted manner.

Improvement mechanism exploration. There may exist some test questions that have been automatically synthesized during data annotation. Though this kind of "data leakage" does not hold for KB transfer learning (because we are glad to teach the model some specific domain information), it is still necessary to discuss its influence. We conduct a cross-validation on LooGLE ("*w/o* know"), in which the amount of training data keeps the same while the exact information corresponding to the test questions are removed during self-annotation. Under this setting, the question embedding similarity between training and test set is decreased from 0.454 to 0.416 (details are shown in section A).

To further verify the generality and effectiveness of our method, we additionally evaluate on the common KBQA benchmark WebASQ. We construct a "w/o know" setting on this dataset: during self-annotation, we remove any training QA whose gold evidence or answer directly overlaps with the test

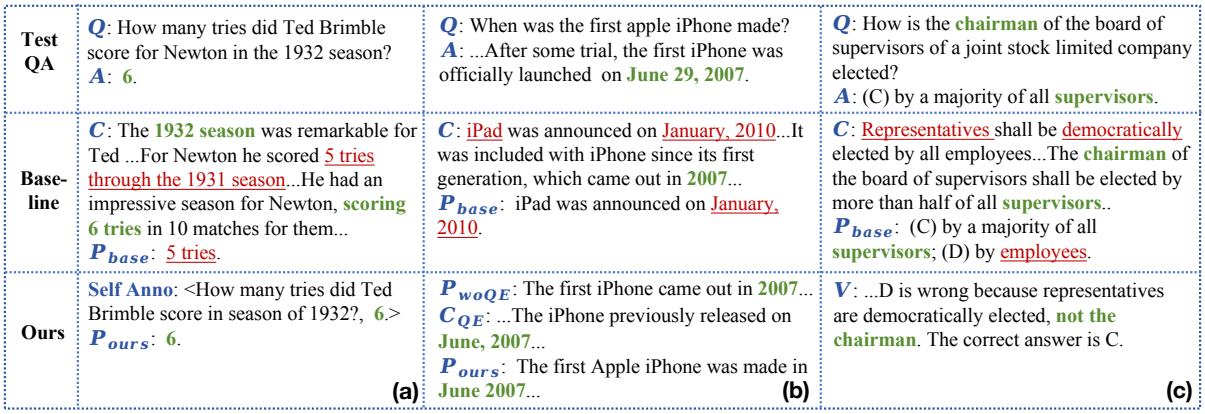

Figure 5: Cases for KBAlign and baseline comparison. We display the translation for the Chinese JEC-QA task. The **bold** text and underlined text providing correct and wrong information for the QA process.

targets, ensuring that self-annotated questions do not duplicate the test set. The number of training steps is kept identical to that in the complete setting. As shown in Table 4, KBAlign achieves significantly better performance than Vanilla RAG; moreover, the "w/o" know results on WebASQ mirror those observed on LooGLE, showing only a modest drop relative to the complete setting while still clearly outperforming the baselines. This pattern supports our claim that the improvements primarily stem from learning domain knowledge, query reformulation, and task format within RAG, rather than memorizing test-specific facts. Detailed similarity statistics and qualitative case studies are provided in section A.

From the results, we can see that the self-adaptation still helps refine the performance, but worse than the complete setting. This indicates that domain knowledge from KB and task format is the main reason for the score rising, while the precise information related to the test data also helps.

4.5 Case Studies

We display typical cases in Fig. 5 to explain the specific usefulness of our strategies. "Baseline" refers to MiniCPM-2B and "ours" refers to the adapted version of it. Overall, KBAlign achieves a general grasp of current KB, a better knowledge answering, and a reasonable confidence verification.

Cases (a) proves the effectiveness of learning knowledge from the self-annotated data. The base model fails to extract useful knowledge (scores in 1932 season) from the indirect context, while the model learns the precise knowledge during self adaptation. Case (b) shows that our model generates a decent prediction (in 2007) though the retriever fails to locate precise information from KB, and this prediction can then help find out useful knowledge with QE, therefore the model eventually provides an even better response (in June 2007). Further, from case (c) we can see that due to the self-verify task mixed into adaptation tuning, the model can check its own prediction and provide a hint of error or incompleteness. Though the verify reason is not always accurate or helpful for modification, it is still meaningful to provide a warning when the confidence is low. Meanwhile, we can also use the verify function as a self-selector for multiple sampling results.

We also see some limitations when observing more cases. To be specific, the self-annotated contains some bias or error, and this may damage the model performance on related questions. Due to the concise language style of annotated data, our model tends to provide short responses in which some useful information may be discarded. QE strategy, in addition, does not always necessary. These negative instances remind us that we should continue to design better annotation and tuning strategies. More cases on different dataset and with various performance are provided in section A.

5 Discussion

In this paper, we introduced KBAlign, a highly efficient self-adaptive method tailored for specific KBs. During the tuning stage, inspired by human learning strategies such as summarizatio and self-reflection, we propose a combined long- and short-dependency annotation method, as well as an iterative tuning approach. These techniques enable low-cost targeted training data augmentation and efficient adaptation without requiring external

supervision. In the inference stage, we enhance the model's performance on KBQA tasks using query expansion and sampling-based self-verify strategies. Our approach demonstrates significant improvements across various datasets spanning different domains and formats. Additionally, detailed analysis provides empirical guidance regarding the best data amount required.

In future work, we aim to focus on adaptive performance enhancement in more complex scenarios, such as utilizing new tools. Additionally, we will explore the integration and collaboration of multiple models adapted to different subdomains.

Limitations

Our approach still has some limitations: (1) **Global information**: While the current method excels in KBQA tasks, especially those focused on local information within the KB, it offers less support for tasks requiring comprehensive global information analysis. This suggests a need for more refined data annotation strategies.

(2) **General Capability**: Training on small-scale targeted data can lead to a reduction in the model's general domain abilities, such as instruction-following. Mixing specific KB data with general domain data, in fact, has proved to be helpful in our side experiment, which is displayed in section A. However, this conflicts with our goal of minimizing adaptation time and cost. We may need to explore techniques like model plugins and routing selection to strike a better balance.

(3) **Retriever Adaptation**: Given the strong influence of retrieval quality on QA performance found in our practice, it may be necessary to consider adapting the retriever during specific KB adaptations. Applying self-supervised strategies to retriever training could be a promising direction.

Acknowledgement

This work is supported by the National Key Research and Development Program of China (2024YFB4505603). This work is supported by the AI9Stars community.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*.

Daixuan Cheng, Shaohan Huang, and Furu Wei. 2023. Adapting large language models via reading comprehension. *arXiv preprint arXiv:2309.09530*.

Jiaxi Cui, Zongjian Li, Yang Yan, Bohua Chen, and Li Yuan. 2023. Chatlaw: Open-source legal large language model with integrated external knowledge bases. *arXiv preprint arXiv:2306.16092*.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. Enabling large language models to generate text with citations. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Soudanim Heydar, Kanoulas Evangelos, and Hasibi Faegheh. 2024. Fine tuning vs. retrieval augmented generation for less popular knowledge. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. 2024. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023a. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023b. Large language models cannot self-correct reasoning yet. In *The*

Twelfth International Conference on Learning Representations.

Kevin Maik Jablonka, Qianxiang Ai, Alexander Al-Feghali, Shruti Badhwar, Joshua D Bocarsly, Andres M Bran, Stefan Bringuier, L Catherine Brinson, Kamal Choudhary, Defne Circi, et al. 2023. 14 examples of how llms can transform materials science and chemistry: a reflection on a large language model hackathon. *Digital Discovery*, 2(5):1233–1250.

Xue Jiang, Yihong Dong, Lecheng Wang, Fang Zheng, Qiwei Shang, Ge Li, Zhi Jin, and Wenpin Jiao. 2023. Self-planning code generation with large language models. *ACM Transactions on Software Engineering and Methodology*.

Qiao Jin, Yifan Yang, Qingyu Chen, and Zhiyong Lu. 2024. Genegpt: Augmenting large language models with domain tools for improved access to biomedical information. *Bioinformatics*, 40(2):btae075.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2023. Loogle: Can long-context language models understand long contexts? *arXiv preprint arXiv:2311.04939*.

Xun Liang, Shichao Song, Zifan Zheng, Hanyu Wang, Qingchen Yu, Xunkai Li, Rong-Hua Li, Feiyu Xiong, and Zhiyu Li. 2024. Internal consistency and self-feedback in large language models: A survey. *arXiv preprint arXiv:2407.14507*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Chen Ling, Xujiang Zhao, Jiaying Lu, Chengyuan Deng, Can Zheng, Junxiang Wang, Tanmoy Chowdhury, Yun Li, Hejie Cui, Xuchao Zhang, et al. 2023. Domain specialization as the key to make large language models disruptive: A comprehensive survey. *arXiv preprint arXiv:2305.18703*.

Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. 2023. Large language models generate functional protein sequences across diverse families. *Nature Biotechnology*, 41(8):1099–1106.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Hongjin Qian, Peitian Zhang, Zheng Liu, Kelong Mao, and Zhicheng Dou. 2024. Memorag: Moving towards next-gen rag via memory-inspired knowledge discovery. *arXiv preprint arXiv:2409.05591*.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen Chen, Ning Ding, Ganqu Cui, Zheni Zeng, and Yufei Huang. 2023. Tool learning with foundation models. *arXiv preprint arXiv:2304.08354*.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. Asqa: Factoid questions meet long-form answers. *arXiv preprint arXiv:2204.06092*.

Ruoxi Sun, Sercan Ö Arik, Alex Muzio, Lesly Miculicich, Satya Gundabathula, Pengcheng Yin, Hanjun Dai, Hootan Nakhost, Rajarishi Sinha, Zifeng Wang, et al. 2023. Sql-palm: Improved large language model adaptation for text-to-sql (extended). *arXiv preprint arXiv:2306.00739*.

Zhen Tan, Alimohammad Beigi, Song Wang, Ruocheng Guo, Amrita Bhattacharjee, Bohan Jiang, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation: A survey. *arXiv preprint arXiv:2402.13446*.

Asahi Ushio, Alva-Manchego Fernando, and Jose. Camacho-Collados. 2023. An empirical comparison of lm-based question and answer generation methods. In *Findings of the Association for Computational Linguistics: ACL*.

Zhen Wan, Yating Zhang, Yexiang Wang, Fei Cheng, and Sadao Kurohashi. 2024. Reformulating domain adaptation of large language models as adapt-retrieve-revise: A case study on chinese legal domain. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 5030–5041.

Benyou Wang, Qianqian Xie, Jiahuan Pei, Zhihong Chen, Prayag Tiwari, Zhao Li, and Jie Fu. 2023a. Pre-trained language models in biomedical domain: A systematic survey. *ACM Computing Surveys*, 56(3):1–52.

Liang Wang, Nan Yang, and Furu Wei. 2023b. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. 2023. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*.

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*, SIGIR '24, page 641–649, New York, NY, USA. Association for Computing Machinery.

Ming Xu. 2023. text2vec: A tool for text to vector. *Software*.

Di Zhang, Wei Liu, Qian Tan, Jingdan Chen, Hang Yan, Yuliang Yan, Jiatong Li, Weiran Huang, Xiangyu Yue, Dongzhan Zhou, et al. 2024a. Chemllm: A chemical large language model. *arXiv preprint arXiv:2402.06852*.

Tianjun Zhang, Shishir G Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E Gonzalez. 2024b. Raft: Adapting language model to domain specific rag. *arXiv preprint arXiv:2403.10131*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Zihan Zhao, Da Ma, Lu Chen, Liangtai Sun, Zihao Li, Hongshen Xu, Zichen Zhu, Su Zhu, Shuai Fan, Guodong Shen, et al. 2024. Chemdfm: Dialogue foundation model for chemistry. *arXiv preprint arXiv:2401.14818*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, YeYanhan YeYanhan, and Zheyan Luo. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410.

Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2020. Jec-qa: a legal-domain question answering dataset. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 9701–9708.

A Appendix

A.1 Dataset Details

LooGLE (Li et al., 2023). We use the short-dependency data in LooGLE for retrofitting, combining altogether 2.2M tokens of text as K, and the corresponding 1,951 Q&A pairs for test.

ASQA (Stelmakh et al., 2022). For each question, there exists several related segments of text from WikiPedia, and a comprehensive long answer that covers much information from them. We collect 794 Q&A pairs for test, their targeted segments and other related passages from WikiPedia, and get 1.8M tokens of text as K.

JEC-QA (Zhong et al., 2020). Related laws and reference books are seen as K, including 21M tokens of text. The train set has also not been used for adaptation, and the test set contains 1,985 of multiple choices.

BioASQ [5]. This contains 324 English questions for testing, each annotated with relevant documents, snippets, and both exact and ideal answers. We utilize 0.41M tokens of text as K.

A.2 Detailed Settings

For hyper-parameter settings, we conduct a grid search in the vicinity based on the empirical values provided in the sample code of the MiniCPM-2B model, and finally determine batch size as 8 and learning rate as $1e-5$. Other settings include warm-up steps as 50 and weight decay as 0.1. For the LLaMA-3.1-8B-Instruct model, we adopt a parameter-efficient tuning approach using the LoRA strategy, with alpha as 16 and rank as 8. Other settings include a cosine learning rate scheduler with a warm-up ratio of 0.1 and a weight decay of 0.1.

For the training process, we adopt the mixed-precision training with the BMTrain [6] and LLaMA Factory (Zheng et al., 2024) framework to speed up.

For the inference process in both annotation and test, we adopt the bge-large-en-v1.5 model for English materials and bge-base-zh-v1.5 for Chinese (Xiao et al., 2024) as the basic retriever of RAG. To ensure continuity of information, we apply an overlap rate of 15% between consecutive chunks. We adopt vLLM (Kwon et al., 2023) to speed up inference.

A.3 Prompts

Below are the prompt templates used for self annotation. For short-dependency annotation, we directly generate by:

```
You are a master of extracting questions and
answers from text.
Based on the provided content, construct five
questions and answers
that should be directly based on the text
content, separated by line breaks.
Please ensure that the expression of the
question clearly points to
the specific information in the text, and avoid
using vague or overly
```

[5] https://huggingface.co/datasets/kroshan/BioASQ

[6] https://github.com/OpenBMB/ModelCenter?tab=readme-ov-file

```
broad references. At the same time, emphasize
direct references or
specific details in the text to increase the
accuracy and depth of the problem.
The questions should be answerable in a few
words.
Output question and answer alternately on each
line.
Content: {content}
Response:
```

For long-dependency annotation, we first generate questions:

```
You will receive a document. Please generate 3
generalizable,
ambiguous questions based on the document
content. The questions
should align with the themes of the document.
Separate the questions
by line breaks.
document: {document}
output:
```

Based on the questions, we then annotate the related information:

```
You will receive a document and a question.
Please provide an answer
to the question based on the document
information. If unable to answer,
return 'none'; otherwise, output the answer
directly.
document: {document}
question: {question}
output:
```

Last, we refine the information to get the answer:

```
You will receive a concatenated answer from
multiple sources.
Please refine and optimize the expression to
make it smoother.
Output the final answer directly without
unnecessary explanation.
question: {question}
answer: {answer}
output:
```

In the iterative tuning phase, we self-verified by:

```
You are a teacher evaluating student responses.
Remember:
1. If the student's response fully aligns with
the golden answer, start your response with 'The
 student's response is correct because'.
2. Otherwise, start your response with 'The
student response is wrong because', and provide
the ERROR TYPE!!! (e.g., does not answer the
question directly, provides totally wrong
information, provides only part of the
information, provides unrelated information)
3. Notice! You are NOT ALLOWED to directly point
 out the correct answer in your verification.
You are NOT ALLOWED to directly point out the
correct answer in your verification. You are NOT
 ALLOWED to directly point out the correct
answer in your verification. You should only
tell me the correctness and the error type.
Now here are the materials:
Reference: {reference}
Question: {question}
Golden Answer: {golden_answer}
Student Response: {student_response}
Please generate your verification. You should
start with the judgement, and then EXPLAIN the
reason / the error type.
```

Below is the prompt template used for downstream QA tasks:

```
You are an expert who has read a lot of
knowledge base.
Please answer the question according to the
content of the KB.
<KB_{kb_id}> You can refer to some segments from
 the KB to help
you answer the question.
References:{references}
Now the question is: {question}
{dataset_prompt}
```

For different datasets, we change the `dataset_prompt` to adjust the output style. Specifically, we refer to ALCE (Gao et al., 2023) when designing the ASQA prompt.

```
LooGLE: Please answer this question.

ASQA: Write an accurate, engaging, and concise
answer for the given question. Use an unbiased
and journalistic tone.

JEC-QA: The answer may be multiple or single, so
 be sure to choose all the correct options.
```

Below is the prompt template used for LLM evaluation (Li et al., 2023):

```
Given one question, there is a groundtruth and a
 predict_answer.
Please decide whether they are the same or not
in semantic.
Please only output 'True' or 'False'.
Question: {question}
groundtruth = {ground_truth}
predict_answer = {predict}
```

A.4 Supplementary Case

We provide several more cases in Fig. 6. Case (d) shows that for different forms of tasks such as multiple choices, the self-annotated data can also provide key knowledge for the model. Case (e) shows a verification example in which the error can only be described in explicit natural language instead of a wrong label. Case (f) shows that our method does not always help improve the performance. In this case, the model discards some useful information due to the concise language style bias.

A.5 General Domain Performance

We conduct experiments on the widely adopted MMLU benchmark (Hendrycks et al., 2021) to

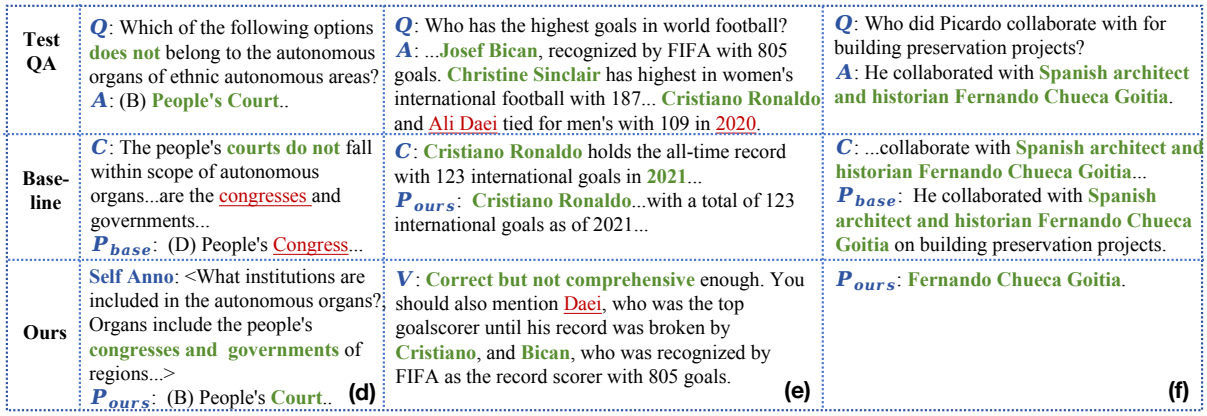

Figure 6: More cases for KBAlign and baseline comparison.

evaluate the general ability reduction of KBAlign brought to the backbone model. *w* mix refers to mixing general instruction tuning data (Ding et al., 2023) with specific KB data. As shown in Table 5, tuning on mixed data could also achieve most of the downstream improvement with a fairly small general performance degradation.

Methods	MMLU	LooGLE F1
MiniCPM-2B		
Vanilla RAG	44.07	30.92
Ours	38.54	54.09
w mix	43.45	52.84
Llama3.1-8B-Instruct		
Vanilla RAG	57.38	40.46
Ours	49.41	62.07
w mix	54.61	61.33

Table 5: Balance between domain knowledge and general capacity.

A.6 Knowledge-Separation Audit and Case Analysis

To complement the results in Table 4, we provide detailed evidence for the knowledge-separation setting as well as representative case analyses.

Similarity Analysis. In our "w/o know" setting, knowledge separation is enforced at the KB level. During construction of the training and test sets, we first identify the KB on which the test questions depend. Only questions relying on these KBs are kept for evaluation, while self-annotation for training is performed exclusively on the remaining KBs that are not used in the test questions. This ensures that the model is trained solely on annotated data from KBs unseen at test time, and therefore never observes annotated versions of the test questions or their supporting knowledge.

Table 6 reports the similarity scores between training and test questions, where each test question is matched with its most similar training question. We can see that the overall similarity under *w/o know* is lower than under *w know*, indicating that our filtering effectively removes high-overlap samples.

Setting	Mean	Std	Median
w/o know	0.4160	0.0775	0.4133
w know	0.4536	0.1165	0.4400

Table 6: Similarity scores between training and test questions on WebASQ.

Overall, the similarity under "w/o know" is significantly lower than under "w know". To further analyze whether any residual knowledge overlap exists and to better understand the reasons for KBAlign's improvement, we examine several high-similarity question pairs. We categorize these cases by whether they represent true answer overlap or merely semantic paraphrases without overlapping answers; representative cases are listed and discussed in the following paragraph.

Case Studies. To qualitatively assess residual overlap, we examined the most similar training questions for four representative test questions under both settings. In each case, the top similarity score under "w/o know" is markedly lower than under "w know", showing that our KB-level filtering effectively removes direct answer overlaps:

w/o know:

- **Test Q:** who was judy collins married to
  Train Q: Who is the spouse of Brooke Collins?

Similarity score: 0.7321

- **Test Q:** what form of government does czech republic have
  Train Q: What is the administrative area type of the Czech region?
  Similarity score: 0.7301

Summary of "w/o know": Although the surface similarity remains moderate, the training question concern different entities or concepts, indicating that the filtering step removes direct answer overlaps and leaves only benign semantic parallels.

w know:

- **Test Q:** what language do argentina use
  Train Q: What is the official language of Argentina?
  Similarity score: 0.9266

- **Test Q:** in what country is amsterdam
  Train Q: What is the state of Amsterdam?
  Similarity score: 0.9123

Summary of "w know": In contrast, "w know" retains very high similarity scores. These cases are mostly paraphrastic matches with different answer focuses, reflecting high semantic overlap but still minimal factual leakage compared to direct duplication.

Algorithm 1 KBAlign Framework

Require: Model M, Retriever R, Golden context C_g, Question Q, Answer A, Split size k
Ensure: Fine-tuned model M_k

Annotation Process:
procedure SHORTANNOTATION(C_g)
  $Q_{\text{short}} \leftarrow M(C_g)$
  $C_R \leftarrow R(Q_{\text{short}})$
  $C \leftarrow C_g \oplus C_R$
  $A_{\text{short}} \leftarrow M(Q_{\text{short}} \oplus C)$
  return $\langle Q_{\text{short}}, A_{\text{short}} \rangle$
end procedure
procedure LONGANNOTATION($\{S_i\}_{i=1}^n$)
  $C_g \leftarrow \bigoplus_{i=1}^n S_i$
  $Q_{\text{long}} \leftarrow M(C_g)$
  $C_R \leftarrow R(Q_{\text{long}})$
  for $i = 1, \ldots, n$ **do**
    $C_i \leftarrow S_i \oplus C_R$
    $I_i \leftarrow M(Q_{\text{long}} \oplus C_i)$
  end for
  $A_{\text{long}} \leftarrow M(Q_{\text{long}} \oplus \bigoplus_{i=1}^n I_i)$
  return $\langle Q_{\text{long}}, A_{\text{long}} \rangle$
end procedure
Training Phase:
Split annotated data $\{\langle Q, A \rangle\}$ into k parts $\{\langle Q_i, A_i \rangle\}_{i=1}^k$
Initial Tuning:
$\mathcal{L}_1 = 0.5\mathbb{E}[\|M(Q_1) - A_1\|] + 0.5\mathbb{E}[\|M(Q_1 \oplus R(Q_1)) - A_1\|]$
$M_1 \leftarrow \arg\min_M \mathcal{L}_1$
Iterative Verifying:
for $i = 2$ **to** k **do**
  $C_R \leftarrow R(Q_i)$
  $P_i \leftarrow M_{i-1}(Q_i \oplus C_R)$
  $V_i \leftarrow M_{i-1}(Q_i \oplus P_i \oplus A_i)$
  $\mathcal{L}_i = 0.375\mathbb{E}[\|M(Q_i) - A_i\|] + 0.375\mathbb{E}[\|M(Q_i \oplus C_R) - A_i\|]$
  $+0.125\mathbb{E}[\|M(Q_i \oplus P_i) - V_i\|] + 0.125\mathbb{E}[\|M(Q_i \oplus C_R \oplus P_i) - V_i\|]$
  $M_i \leftarrow \arg\min_{M_{i-1}} \mathcal{L}_i$
end for