Speculative RAG: Reducing Hallucinations and Enhancing Efficiency in AI Systems

Harshita S
Department of Computer Science and Engineering, RV University
harshitas.btech22@rvu.edu.in

Vasanth J
Department of Computer Science and Engineering, RV University
vasanthj.btech22@rvu.edu.in

Abstract

Recent work on RAG focused more on improving the precision and relevance of generated responses by attaching external knowledge bases to large language models. However, efficiency comes at the cost of higher latency and computational burden for RAG-based systems. This paper presents Speculative RAG, a new framework for boosting the efficiency of RAG while minimizing hallucination. We thus get more diversified and correct answers by having a bigger generalist LLM, verifying multiple drafts, produced in parallel, by a smaller, distilled specialist LLM. Each draft is based on a different subset of retrieved documents, so we have a multiplicity of perspectives on the evidence and try to avoid any possible long context dependency bias. This allows acceleration of the RAG process by shifting the task of drafting to a smaller model and just doing one verification pass on the larger model, greatly reducing the latency. Besides, we introduce speculative sampling and staged speculative decoding algorithms that optimize decoding in distributed and on-device settings to achieve large speed-ups with no loss of output quality. The vast experiments we performed confirm that our Speculative RAG surpasses other findings of several benchmarks: TriviaQA, MuSiQue, PubHealth, and ARC-Challenge. Herein, accuracy improves up to 12.97% while reducing latency to 51% compared to the respective conventional RAG systems. This framework cuts down both computational overhead and hallucination in large-scale applications of a language model.

1. Introduction

1.1 Problem Statement

Large Language Models have achieved enormous success on question-answering tasks yet fail generally on knowledge-intensive questions requiring the latest information or hard-to-find facts, with spectacular consequences of factual errors and hallucinations. This joint deficit can be addressed using such external knowledge with Retrieval-Augmented Generation, but multiple retrievals increase input length-an undesirable side-effect in terms of latency and computational cost. Recent work towards RAG focuses more attention on the quality of retrieval but almost completely disregards a concern for efficiency in processing long contexts.

We introduce Speculative RAG : a more efficient and accurate framework. In Speculative RAG, we leverage a smaller special LM to generate several answers in parallel, each of which is based on one of the different subsets of the retrieved documents. This decreases input token counts as well as increases response diversity. A larger generalist LM then makes a single verification pass to verify and select the most accurate response without needlessly processing the same set of documents more times than necessary and thereby reducing hallucination.

Our strategy simplifies the RAG procedure by deferring the drafting to the smaller LM; it also enables faster inference without sacrificing the quality of the answer. In extensively diverse experiments with a set of benchmarks, we show that Speculative RAG outperforms even the state-of-the-art base RAG systems by up to 12.97% accuracy and reduces latency up to 51%, thus making it a much more efficient solution for applications that demand real-time knowledge-intensive processing.

2. Related Works

Retrieval-Augmented Generation helps augment LLMs with knowledge from external documents toward factual accuracy. The most recent strategies are SAIL by Luo et al., 2023a and Self-Reflective RAG by Asai et al., 2023, focusing on optimization of retrieval strategies either with the expensive requirement of instruction tuning or proving inefficient for long contexts. Corrective RAG by Yan et al., 2024 provides an inexpensive evaluator but cannot reason. The Speculative RAG uses a much smaller RAG drafter to create drafts from subsets of documents, which are then validated by an LM for the added efficiency with increased accuracy.

Speculative decoding (Stern et al., 2018; Xia et al., 2024) drafts in multiple tokens with a smaller model and verifies them in parallel for reduced latency. Our method extends the concept to answer-level drafting where the generalist LM verifies the entire draft for confidence, thereby improving inference speed without loss of quality.

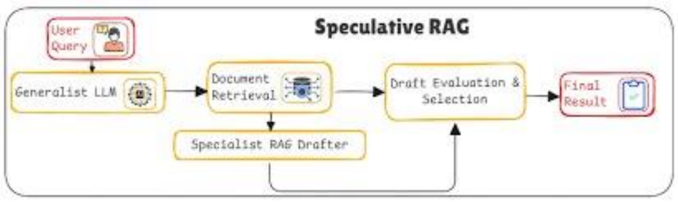
Other techniques, including quantization by Dettmers et al. (2022), distillation of Sanh et al. (2019), and multi-query attention by Shazeer (2019) have also optimized transformer sampling. These techniques further improve throughput but pay less importance to latency reduction for the timely completion of real-time tasks. This paper extends those ideas to distributed large-scale settings that provide yet more efficiency for retrieval-augmented systems.

3. Methodology

3.1 Speculative RAG Drafter

Speculative RAG is a new approach to improve LLM's performance on knowledge-intensive tasks, not involving significant parameter scaling or instruction-tuning of the model at large. It adopts a divide-and-conquer strategy and uses a smaller specialist LM for generating answer drafts and rationales based on retrieved, verified documents by a larger generalist LM, known as the RAG verifier. In this way, it is possible to conduct efficient reasoning over retrieved information with no loss of speed.

MDrafter is tuned for generating an answer, together with the rationale of this answer for a specific question, and relevant documents together with a rationale for explaining why this answer is plausible according to the documents. Further improvement in producing responses grounded was achieved by incorporating instruction tuning; this extension extends each training triplet (query, answer, document) with a rationale. The final rationale is synthesized using strong LMs and then fine-tuned with a standard language modeling objective in order to guide the process of generation in MDrafter.



Further, SPECULATIVE RAG makes use of a multi-perspective sampling strategy to augment the numbers of diverse retrieved documents for each query, while the diversity of documents retrieved supports that the generated drafts of MDrafter are well-rounded and contextually grounded. These drafts are then parallelized through processing; reliability of each draft is measured using a conditional generation probability such that a good response is elicited with strong rationale support.

3.2 Speculative RAG Verifier

After the RAG Drafter MDrafter produces drafts and rationales, RAG Verifier MVerifier verifies pairs of drafts and rationales and selects the most reliable answer (typically, using an off-the-shelf, pre-trained language model rank-ing draft-rationale pairs (α, β) for self-consistency and reliability while skipping otherwise redundant retrieval results).

There are two evaluation scores in determining draft quality: the *self-consistency score* (ρSelf-contain) and the *self-reflection score* (ρSelf-reflect). The former calculates the probability that a given pair consisting of a question together with a draft, will be self-coherent. The latter uses an prompting environment in which MVerifier checks whether the rationale for an answer is itself justified, answering "Yes" or "No". Both are computed in a single forward pass of the model, utilizing auto-regressive properties to accumulate probabilities over relevant tokens .

$$\underrightarrow{Q, \overbrace{\alpha, \beta}^{\rho_{\text{SC}}}, R, \overbrace{\text{"Yes"}}^{\rho_{\text{SR}}}} \Rightarrow \begin{cases} \rho_{\text{SC}} = \prod_{t_i \in \alpha} P(t_i|t_{<i}) \cdot \prod_{t_i \in \beta} P(t_i|t_{<i}) \\ \rho_{\text{SR}} = \prod_{t_i \in \text{"Yes"}} P(t_i|t_{<i}) \end{cases}$$

The final score for each draft is given by the product of its reliability, self-consistency, and self-reflection scores (ρj = ρDraft,j · ρSC,j · ρSR,j). The highest scoring draft is selected as the final answer to the question.

3.3 Speculative RAG Sampling

Speculative sampling reduces latency in the large LLM decoding, especially with regard to large transformers, and increases efficiency. The sampling focuses on sampling different tokens in parallel using a rejection sampling scheme to recover the target model's distribution.

Key Takeaways

- **Sampling Latency** For large models, e.g., transformers Megatron-style, the latency to sample K tokens in parallel is basically the same as the latency of sampling a single token because of the two following components:

- **Linear Layers** Memory-bound as the amount of data being processed through them is not too large, so the latency is almost unaffected when processing a handful of tokens in parallel.
- **Attention Mechanism** The KV-cache size is fixed and therefore there is not much latency impact of parallelizing token generation.
- **All-reduces** Communication overhead in distributed models is latency-bound for small K, contributing similar delays to sampling and scoring.

- **Modified Rejection Sampling** the sampling procedure is modified. Accept or reject draft tokens based on their probabilities from the draft model and the target model. The acceptance probability thus follows this function:

acceptance=min(q(x~n,1|x1…xn)p(x~n,1|x1…xn))
acceptance=min(p(x~n,1|x1…xn)q(x~n,1|x1…xn))

If a token is accepted, it is added to the sequence; otherwise, it is resampled from the target distribution. This guarantees that tokens are aligned with the distribution in the target model while still keeping sampling efficiency.

Performance Benefits:

In the draft-accept loop, at least one token will always be accepted. This ensures that there is always progress in generating the token.
If all the draft tokens are accepted, then a full sequence may be generated normally in return yielding up to K tokens per loop instead of just K as in naive implementations.

Cross Compatibility with Other Techniques: The technique is sufficiently flexible and shall allow integration with other optimizations, like quantization and multi-query attention, for even more acceleration of sampling or for some memory usage reductions.

4. Experimental Results

4.1 BaseLines

Standard RAG requires all the retrieved documents in the prompt as contextual information for generating responses. In this method, the model applies off-the-shelf large language models, including Mistral7B, Mistral-Instruct7B, Mixtral8x7B, Mixtral-Instruct8x7B, and Alpaca7B for experiments. The performance of Toolformer7B, an LLM instruction-tuned to use external tools such as search engines, and SAIL7B, an LLM instruction-tuned on the Alpaca instruction tuning set augmented with search results from sources such as DuckDuckGo and Wikipedia, is also gauged. In Standard RAG, the model produces responses based on the full set of retrieved documents without any further refinement or reflection on the quality of the retrievals.

Self-Reflective RAG and Corrective RAG are advanced RAGs used for enhancing the quality of information in the context based on which responses will be generated. The self-RAG instruction process tunes the model to generate self-reflection tags that will enable the model to fetch extra documents that it requires and position itself to criticize the retrieved documents before providing a response. This makes the model improve their responses constantly using evaluation of the contexts. CRAG includes an external evaluator, which scans the quality of the retrieved documents and filters out the poor ones to reach the response generation. The Self-CRAG method applies the same principle of the Self-RAG approach to filtered documents obtained with CRAG; the Self-CRAG method further enhances the quality of the input documents it retrieves with the Self-CRAG method. The above methods are used for ensuring the responses generated are based on high-quality relevant information.

4. 2 Experiment Settings

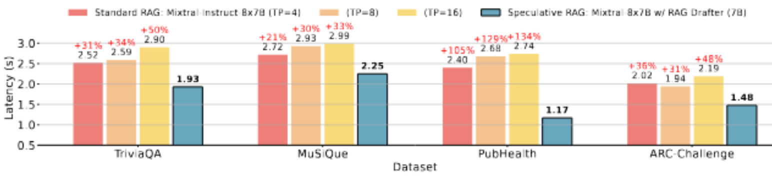
In the experiment settings, Mistral7B (v0.1) will be taken as the base language model (LM) for the RAG drafter and can either be Mistral7B (v0.1), or Mixtral8x7B (v0.1) as RAG verifier, which we will refer to as MVerifier-7B or MVerifier-8x7B, respectively, without fine-tuning. The retrieval makes use of InBedderRoberta, a lightweight instruction-aware embedding model, to pre-compute embeddings of retrieved documents. Inference will be carried out by the use of vLLM framework with greedy decoding temperature = 0. The experiments use settings from Asai et al. (2023) and include a challenging benchmark, MuSiQue (Trivedi et al., 2022), which trains on RAG reasoning instead of evidence citation. For the datasets TriviaQA, PubHealth, and ARC-Challenge, the top 10 documents are returned and 5 drafts are generated per query using a subset of 2 documents ($k = 2$). For MuSiQue, 15 documents are retrieved as answer documents and 10 drafts are produced as intermedia outputs, each based on a subset of 6 documents, to tackle more involved reasoning.

The experimental setup evaluates Speculative RAG under various settings with comparison to several RAG approaches, such as Standard RAG, Self-Reflective RAG, and Corrective RAG. Some performance metrics across several datasets of speculative RAG indicate improved outcomes compared to the others. Specifically, MDrafter-7B alone and in combination with MVerifier-7B or MVerifier-8x7B outperformed standard and reflective RAG in terms of delivering more accurate answers.

4. 3 Main Results

We test SPECULATIVE RAG with respect to the baseline RAG methods and also other sophisticated techniques such as Self-Reflective RAG and Corrective RAG using four datasets: TriviaQA, MuSiQue, PubHealth, and ARC-Challenge. We report the results for MDrafter-7B and that for the MDrafter-7B together with the RAG verifier, such as MVerifier-7B, MVerifier-8x7B. Accuracy remains the primary metric as used in the majority of the previous works (Asai et al., 2023; Yan et al., 2024).

- **Superior Performance over Baselines**: SPECULATIVE RAG is always more superior to all other baseline models in the four benchmarks from Table 1. This can be seen as below, in particular, MVerifier-8x7B + MDrafter-7B does better compared to the best of standard RAG models, which is Mixtral-Instruct8x7B, in 0.33% of TriviaQA, 2.15% of MuSiQue, and 12.97% PubHealth while 2.14 of ARC-Challenge. With MDrafter model, the MVerifier-7B is also superior to other Self-Reflective Corrective RAG methods, plus in the majority of scenarios, it is also better than the other baselines.
- **Effective Instruction Tuning for RAG Drafter**: Our instruction tuning significantly enhances the reasoning capability of the drafter model (Hsieh et al., 2023) and significantly improves performance compared to Mistral7B when compared with MDrafter-7B. Moreover, Mixtral8x7B shows considerable improvements when combined with the instruction-tuned RAG drafter, MDrafter-7B, with a gain of 14.39% on TriviaQA, 12.41% on MuSiQue, 39.52% on PubHealth, and 31.83% on ARC-Challenge. Mistral7B also enjoys the same advantages, for instance on TriviaQA with 19.76%, MuSiQue with 14.32%, PubHealth with 40.94%, and ARC-Challenge with 33.44%. These are because the superior reasoning abilities of the RAG drafter decrease the retrieval redundancy, which makes it possible to generate more accurate answer drafts by taking into account diverse views from the retrieval results.
- **Reliable Scoring by RAG Verifier** : Dependent draft verification by the generalist language model further enhances the performance. There is a significant performance leap while comparing the MDrafter-7B with MVerifier-7B + MDrafter-7B. The instruction-tuned RAG drafter excels well in generating answer drafts from the retrieved documents, but the language modeling power of generalist LMs is actually used to evaluate each draft in terms of its reasoning. This approach is not only effective but also straightforwardly implementable, which, in turn, points to the efficiency of this verification method.



4. 4 Latency Analysis

We evaluate the latency of Standard RAG and SPECULATIVE RAG over four datasets: TriviaQA, MuSiQue, PubHealth, and ARC-Challenge. For each dataset we sample 100 cases at random and report the average processing time of each case in Figure 2. In order to simulate real life, we handle each case individually and without batching. We primarily test MVerifier-8x7B + MDrafter-7B on SPECULATIVE RAG and Mixtral-Instruct8x7B on Standard RAG since they outperform the closest competitors baselines presented in Table 1. For parallel drafting, we utilize 5 endpoints of MDrafter-7B on TriviaQA, PubHealth, and ARC-Challenge, and 10 endpoints for MuSiQue due to this model's greater number of drafts. We adapt Mixtral-Instruct8x7B to the GPU memory using tensor parallelism at parallelism sizes of 4, 8 and 16. It turns out that increasing tensor parallelism is not a speedup because of overhead in aggregating and communicating tensors. On the other hand, SPECULATIVE RAG with its small RAG drafter and parallel draft generation is always worse than that latency on all the datasets. It can decrease latency up to 23.41% on the TriviaQA dataset, up to 17.28% on MuSiQue, 51.25% on PubHealth, and 26.73% on ARC-Challenge; thus, how our approach can reduce processing times without dropping their substantial performance.

5. Conclusion:

SPECULATIVE RAG splits the RAG task into two separate stages: draft and verification. Drafting is done by a very small, dedicated RAG drafter, while verification is done by a large generalist language model (LM). This enhances the quality of the answer candidates, reduces the number of tokens in input, and reduces position bias over long contexts by producing multiple drafts in parallel from different document subsets. This leads to great quality and performance improvements of the generated final output. SPECULATIVE RAG reaches an improvement up to 12.97% in accuracy and a reduction of 51% in latency compared with traditional RAG-based systems. This work further brings out the scope of collaborative architectures to enhance RAG performance through decomposition of the task.

Limitations:

Though SPECULATIVE RAG enhances both accuracy and efficiency, its use comes at the expense of having to train a whole separate drafter model, which adds overhead compared to just using the standard instruction-tuned RAG model. However, training the drafter model is computationally cheap.

6. References:

[1] Heming Xia Tao Ge. " Speculative Decoding: Exploiting Speculative Execution for Accelerating Seq2seq Generation." 2022.

[2] Izacard, G., Caron, M.Unsupervised dense information retrieval with contrastive learning.2021

[3] Zilong Wang, Zifeng Wang. " Speculative RAG: Enhancing Retrieval Augmented Generation through Drafting." 2024.

[4] Charlie Chen , Sebastain Borgeaud. "Accelerating Large Language Model Decoding with Speculative Sampling." 2023..

[5] Sehoon Kim, Karttikeya Mangalam.“Speculative Decoding with Big Little Decoder” , 2023.

[6] Gao, Y., Xiong, Y. Retrieval-augmented generation for large language models.2020

[7] Baek, J., Jeong, S.Knowledge-augmented language model verification.2022

[8] Ma, X., Yang, X. Efficient llm pretraining and inference with unlimited context length.2024

[9] Min, S., Krishna, K. Conference on Empirical Methods in Natural Language Processing. 2023

[10] Min, S., Krishna, K. Conference on Empirical Methods in Natural Language Processing. 2023