Improving Small LLMs performance with RAG

1st Thiago Souza   
Centro de Informática  
*UFPE*Recife – PE, Brazil  
tas3@cin.ufpe.br

2nd Cleber Zanchetin  
Centro de Informática  
*UFPE*Recife – PE, Brazil  
tas3@cin.ufpe.br

**Abstract —** **Large Language Models (LLMs) have revolutionized access to knowledge, enabling anyone with an internet connection to explore a vast array of topics. However, LLMs are inherently limited by their training data and struggle with subjects beyond their pre-trained knowledge. This challenge is even more pronounced in smaller LLMs, which, due to their reduced parameter count, have a lower capacity to capture complex or niche information. With the growing interest in deploying LLMs on edge devices or local systems, improving their performance is crucial. In this study, we explore how Retrieval-Augmented Generation (RAG) can enhance the inference capabilities of small LLMs by dynamically incorporating relevant context from a vector database. Our research evaluates the extent to which RAG can bridge the performance gap between small and large LLMs, making them more viable for real-world applications**.

Keywords — LLM, RAG, Retrieval-Augmented Generation, Small Language Models, Vector Databases

# Background

The introduction of Multi-Head Attention transformer-based Large Language Models (LLMs) [1] revolutionized how people learn, work, and interact with technology. OpenAI popularized these models through their online web chat tool, ChatGPT, making advanced AI-driven conversations accessible to the general public.

Before LLMs, computer-based search primarily relied on string matching or heuristic-based algorithms in platforms like Google and Bing. These approaches, while effective, often required users to formulate precise queries and manually filter through results. The advent of LLMs fundamentally changed this dynamic by enabling users to engage in natural language interactions. Instead of merely retrieving keyword-based results, LLMs interpret queries contextually and generate human-like responses, giving users the impression of conversing with an intelligent assistant.

This transformation not only enhanced search and information retrieval but also expanded how people process and synthesize knowledge. By leveraging attention mechanisms and text embeddings, LLMs facilitate semantic search, allowing users to explore topics more intuitively and extract deeper insights from their queries.

However, running inference-based pipelines on those LLMs is only possible in very powerful data centers as it’s massively dependent on a very high number of GPUs and energy availability – which are significant issues in resource constrained environments. LLMs are also very high energy consumers which makes it also very expensive to run on scale. That has created the need of having to explore and design Small Language Models (SM) that, in nature, are similar to LLMs but rely on a much smaller number of parameters hance have more lightweight inference in terms of GPU requirements and energy consumption.

| Dimension | LLMs | SMs |
| --- | --- | --- |
| Accuracy | state-of-the-art | decent |
| Generality | general purpose | task specific |
| Efficiency | resource intensive | resource efficient |
| Interpretability | low interpretable | high interpretable |

Table 1: comparisons of dimensions between LLMs and SMs

Since knowledge is encoded in parameters in language models, it’s not surprising that SMs have worse overall performance in inference tasks against their large counterparts. It’s also less capable of generalizing and handle broad spectrum of information [2].

There are many techniques such as fine-tuning, knowledge distillation or Retrieval Augmented Generation (RAG) can be used to make SMs more competitive against LLMs. This document aims to explore the use of RAG in the scope of Question / Answer task in fixed knowledge domain as a tool to make SMs a viable option in resource constrained setups.

# Experiment and Contributions

Our objective is to investigate whether and to which extent RAG can help Small Language Models improve or match performance of larger models.

## The experiment

Our exploration involved the following high-level steps: first we ask a given set of questions to various models, then we evaluate their answers by sending the answer and ground truth to a judge, then we ask the same dataset of questions along with some context data – fetched from a vector database – and process the answers to the judge. The judge itself is a Large Language Model; we selected Open AI’s GPT-4o-mini but in future work it’s worth exploring more capable models such as Open AI’s *o3* models that have higher reasoning capabilities. In this research it wasn’t possible due to its high cost per 1M tokens.

The dataset we use in this experiment is the Harry Potter book series and the Harry Potter Trivia Questions [3]

## The compared models

In our research, selected models from various vendors and of various sizes and divided them in four high level buckets: tiny (2B or less parameters), small (3B – 7B parameters), medium (8B – 14B parameters) and large (100B parameters or more).

* Tiny: qwen2.5-0.5b, deepseek-r1-1.5b, qwen2.5-1.5b, gemma3-1b, gemma2-2b.
* Small: llama3.2-3b, llama2-7b.
* Medium: gemma3-12b, deepseek-r1-14b
* Large: GPT-4o-mini

## The Objective and Contributions

With this experiment we aim to investigate whether RAG can help small LLMs match the performance of larger models in closed domain Question-Answer tasks.

Our contributions are:

* Compare small and large LLMs in a closed-domain QA task (Harry Potter books).
* Evaluate the impact of RAG-enhanced retrieval on smaller models.
* Provide quantitative analysis using multiple evaluation metrics

# Related Work

The rise of Large Language Models has prompted an active research community to examine not only their capabilities but also the practical trade-offs between model size, domain specialization, and augmentation strategies such as retrieval augmented generation (RAG). This section highlights relevant research exploring the performance of domain-specific LLMs, the role of small models in the LLM ecosystem, and recent findings on the limitations of smaller LLMs in multi-step tasks.

There has been investigations where scientists question the necessity of training specialized domain-specific models in light of the capabilities of general-purpose LLMs like GPT-3.5 and GPT-4 [4]. Some of the findings suggest that general-purpose LLMs, when given sufficient context through prompt engineering or retrieval, can perform competitively with domain-specific models. This supports the idea that retrieval-augmented generation (RAG) may compensate for gaps in internal model knowledge, especially for more generalizable models—an idea central to our study, where we test whether smaller LLMs can similarly benefit from retrieval in a closed-domain QA task. Another study in this space highlights the growing relevance of small LLMs for edge and privacy-sensitive applications [5]. The survey emphasizes techniques such as distillation and RAG to improve their utility without increasing model size — aligning with our focus on enhancing small models through retrieval. There’s also studies on how selecting right chunk size improves results of RAG based applications by ensuring optimization in the use of the LLM context window [6].

In contrast, there’s also research demonstrating that small models struggle with complex tasks requiring reasoning and tool use. While our setting is simpler — closed-domain QA — this reinforces the motivation to explore whether small LLMs, aided by retrieval, can still perform competitively in targeted applications [7].

# Methodology

## Dataset and knowledge source

In our research to serve as the knowledge base data we used the Harry Potter Books dataset [8] that contains all seven books of the famous series. The dataset have the books in plain textual form which was a convenient aspect that facilitated the processing step in the knowledge base chunking and ingesting step in the pipeline.

## Question/Answer dataset

For the question and answer step we relied on the Harry Potter Trivia Questions dataset [9]. The dataset has two files train and test both using the *parquet* file format where each line in the files present a question and an answer. We converted the *parquet* files to a textual form so we could process them as a comma separated value file.

## Baseline evaluation

In order to serve as a baseline we ran an inference job of a total of 50 questions through the models referred and out of the answers we calculated an average score visible in the Table 2. It’s noticeable that most SMs perform quite poorly with the exception of the DeepSeekR1-14B that performs even better than the LLM GPT-4o-mini, which indicates that SMs can indeed be competitive in some closed domain tasks even without the support of techniques like RAG.

|  |  |
| --- | --- |
| Model | Score |
| qwen2.5-0.5b | 0.1184 |
| deepseek-r1-1.5b | 0.2163 |
| llama2-7b | 0.2408 |
| qwen2.5-1.5b | 0.2612 |
| gemma3-1b | 0.2735 |
| llama3.2-3b | 0.3082 |
| gemma2-2b | 0.3531 |
| gemma3-12b | 0.4939 |
| deepseek-r1-14b | 0.6306 |
| Small LLMs Average | 0.3218 |
| gpt-4o-mini (LLM) | 0.5857 |

Table : results of SMs and LLMs

## The Judge

The score evaluation was given by an LLM (GPT 4o mini) that acted as the judge of how close to the ground truth each answer was in a range that went from 1 to 10 which then got converted to a range between 0 and 1 where 0 means poor answer and 1 means perfect answer. The prompt used in the evaluation process, passed to the judge LLM, provided instructions to the LLM to be impartial and consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and specially how direct the response was [Judge Prompt]. We then calculated the average score for each model assessed.

## The Question Answerer AI assistant (QAAI)

The AI assistant is the component in the Question and Answer pipeline that receives the question, combines it with an instruction prompt and sends it out for LLM inference. Then the response is received and processed to extract the LLM answer from it. This is the component in the pipeline that works based off of abstractions that allow us to pass a model name and: loads model with such name into the local environment, or, in the case of a Large LLM (such as GPT 4o mini), sends out the query to a cloud based API.

## Question and Answer (Q&A) pipeline

The Q&A pipeline consists of the steps of: 1/ loading the questions and ground truths dataset and, for each question and ground truth, 2/ sending the question to the QAAI and storing the answer, then 3/ sending the tuple (question, ground truth, answer) to the evaluators, 4/ storing the result aggregated by model name locally for future analysis.

## Evaluators

For the evaluation step, we have experimented with multiple evaluators such as string distance based of the Levenshtein or Jaro-Winkler and LLM based evaluators.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Jaro Winkler | Levensthein | LLM |
| deepseek-r1-1.5b | 0.4905 | 0.9906 | 0.2163 |
| deepseek-r1-14b | 0.4818 | 0.9899 | 0.6184 |
| gemma2-2b | 0.4455 | 0.7312 | 0.3531 |
| gemma3-12b | 0.3228 | 0.5828 | 0.4878 |
| gemma3-1b | 0.5547 | 0.874 | 0.1021 |
| gpt-4o-mini | 0.3304 | 0.5792 | 0.5959 |
| llama2-7b | 0.5164 | 0.9299 | 0.2347 |
| llama3.2-3b | 0.5098 | 0.8344 | 0.3122 |
| qwen2.5-0.5b | 0.586 | 0.9 | 0.1184 |
| qwen2.5-1.5b | 0.4875 | 0.8199 | 0.2633 |

Table Evaluators results: String distance based versus LLM based score

In the Table 3 it is visible how the LLM based score is misaligned with the String based stores. Doing an in-depth human analysis on how those score evaluators aligned with the human assessment of whether the answer deserved a bad score (close to zero) versus a good score (close to one) the outcome was that the LLM based scoring was much more aligned. Our interpretation is that the nature of the evaluation doesn’t allow for string distance based evaluation as with that one loses the semantic aspect of the comparison; as such we decided to rely solely on the LLM based Evaluator described in the section The Judge.

## RAG pipeline: chunking the data

Once collected the baseline data we then designed the RAG pipeline where the first step we took was to build the process that would chunk the knowledge base data in chunks that could then be ingested by the Vector database of our choice. We decided to have three chunking strategies:

* Chunk size of 600 with overlap of 100 chunks
* Chunk size of 1000 with overlap of 100 chunks
* Chunk size of 1500 with overlap of 100 chunks

The reason was that we wanted to have the ability to experiment with various chunking configurations and measure which one performed better.

## RAG pipeline: ingesting the knowledge base data

The database of choice was Pinecone [10] which is a very popular and high performing vector database RAG pipeline based applications use but we have also experimented using ChromaDB which is an open source alternative that provided reasonable results [11]. We created three indexes in Pinecone where each one hosted one of the three chunking configurations listed above and shifted between them in our experiments.

The ingestion process involved loading the books corpus in memory and chunking them according to the configuration passed and creating documents in the vector database with a source reference pointing to which book had the context.

## RAG pipeline: context retrieval

Context retrieval is a key step in a RAG pipeline and the results of our RAG pipeline are highly influenced by the various choices one can make in this step. In our research we made several experiments with different search algorithms, various similarity thresholds and relevant hyper parameters like *k* that represents the max number of documents the pipeline will retrieve to combine with the prompt [6]. The search algorithms we experimented with were: *similarity\_score\_threshold* that computes the similarity between the context and the query and assigns a score between 0 and 1 and returns documents that have score above a certain threshold, and *mmr* (Max Marginal Relevance) that combines similarity and diversity.

# Experiments and Results

## Baseline performance

As seen in Table 2 most small LLMs performed poorly with the majority of them scoring below 0.3, of a range between 0 and 1, with the exception of the DeepSeek R1 14B model, which was actually the baseline winner, and performed even better than the only Large model we used in this experiment, GPT 4o mini. The Small LLMs average score was 0.3218 showing that, without providing more context for the LLM to perform its inference, the accuracy of the answers was in general not very useful.

## Impact of RAG

To test our hypothesis, that RAG could make small models be competitive against their large counterparts in domains like the closed domains in question and answer tasks, we executed the RAG pipeline and collected the results which can be seen in the Table 4. With the introduction of RAG in the inference process the average score across small LLMs increased to 0.4272 which represented an increase of 42% compared to average of the small LLMs without the support of RAG as visible in the Table 4. We also see that not all models benefited from having RAG support such as Gemma3-12B that consistently performed worse when RAG support was provided in our experiments. For our surprise, GPT 4o mini also scored worse when RAG support was introduced with a score 21% below the execution with no RAG support.

|  |  |  |
| --- | --- | --- |
| Model | No RAG | RAG |
| gemma3-1b | 0.1021 | 0.3000 |
| qwen2.5-0.5b | 0.1184 | 0.3551 |
| deepseek-r1-1.5b | 0.2163 | 0.3571 |
| qwen2.5-1.5b | 0.2633 | 0.3939 |
| gemma3-12b | 0.4878 | 0.4204 |
| llama2-7b | 0.2347 | 0.4245 |
| gemma2-2b | 0.3531 | 0.4327 |
| llama3.2-3b | 0.3122 | 0.4898 |
| deepseek-r1-14b | 0.6184 | 0.6714 |
| Small LLMs Average | 0.3007 | 0.4272 |
| gpt-4o-mini (LLM) | 0.5959 | 0.4694 |

Table Small LLMs vs Large LLM with/out RAG

## Analysis on SMs that performed worse with RAG

## Units

* Use either SI (MKS) or CGS as primary units. (SI units are encouraged.) English units may be used as secondary units (in parentheses). An exception would be the use of English units as identifiers in trade, such as “3.5-inch disk drive”.
* Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity that you use in an equation.
* Do not mix complete spellings and abbreviations of units: “Wb/m2” or “webers per square meter”, not “webers/m2”. Spell out units when they appear in text: “. . . a few henries”, not “. . . a few H”.
* Use a zero before decimal points: “0.25”, not “.25”. Use “cm3”, not “cc”. (*bullet list*)

## Equations

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the Times New Roman or the Symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled.

Number equations consecutively. Equation numbers, within parentheses, are to position flush right, as in (1), using a right tab stop. To make your equations more compact, you may use the solidus ( / ), the exp function, or appropriate exponents. Italicize Roman symbols for quantities and variables, but not Greek symbols. Use a long dash rather than a hyphen for a minus sign. Punctuate equations with commas or periods when they are part of a sentence, as in:

*a**b* 

Note that the equation is centered using a center tab stop. Be sure that the symbols in your equation have been defined before or immediately following the equation. Use “(1)”, not “Eq. (1)” or “equation (1)”, except at the beginning of a sentence: “Equation (1) is . . .”

## Some Common Mistakes

* The word “data” is plural, not singular.
* The subscript for the permeability of vacuum **0, and other common scientific constants, is zero with subscript formatting, not a lowercase letter “o”.
* In American English, commas, semicolons, periods, question and exclamation marks are located within quotation marks only when a complete thought or name is cited, such as a title or full quotation. When quotation marks are used, instead of a bold or italic typeface, to highlight a word or phrase, punctuation should appear outside of the quotation marks. A parenthetical phrase or statement at the end of a sentence is punctuated outside of the closing parenthesis (like this). (A parenthetical sentence is punctuated within the parentheses.)
* A graph within a graph is an “inset”, not an “insert”. The word alternatively is preferred to the word “alternately” (unless you really mean something that alternates).
* Do not use the word “essentially” to mean “approximately” or “effectively”.
* In your paper title, if the words “that uses” can accurately replace the word “using”, capitalize the “u”; if not, keep using lower-cased.
* Be aware of the different meanings of the homophones “affect” and “effect”, “complement” and “compliment”, “discreet” and “discrete”, “principal” and “principle”.
* Do not confuse “imply” and “infer”.
* The prefix “non” is not a word; it should be joined to the word it modifies, usually without a hyphen.
* There is no period after the “et” in the Latin abbreviation “et al.”.
* The abbreviation “i.e.” means “that is”, and the abbreviation “e.g.” means “for example”.

An excellent style manual for science writers is [7].

# Using the Template

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

## Authors and Affiliations

**The template is designed for, but not limited to, six authors.** A minimum of one author is required for all conference articles. Author names should be listed starting from left to right and then moving down to the next line. This is the author sequence that will be used in future citations and by indexing services. Names should not be listed in columns nor group by affiliation. Please keep your affiliations as succinct as possible (for example, do not differentiate among departments of the same organization).

### For papers with more than six authors: Add author names horizontally, moving to a third row if needed for more than 8 authors.

### For papers with less than six authors: To change the default, adjust the template as follows.

#### Selection: Highlight all author and affiliation lines.

#### Change number of columns: Select the Columns icon from the MS Word Standard toolbar and then select the correct number of columns from the selection palette.

#### Deletion: Delete the author and affiliation lines for the extra authors.

## Identify the Headings

Headings, or heads, are organizational devices that guide the reader through your paper. There are two types: component heads and text heads.

Component heads identify the different components of your paper and are not topically subordinate to each other. Examples include Acknowledgments and References and, for these, the correct style to use is “Heading 5”. Use “figure caption” for your Figure captions, and “table head” for your table title. Run-in heads, such as “Abstract”, will require you to apply a style (in this case, italic) in addition to the style provided by the drop down menu to differentiate the head from the text.

Text heads organize the topics on a relational, hierarchical basis. For example, the paper title is the primary text head because all subsequent material relates and elaborates on this one topic. If there are two or more sub-topics, the next level head (uppercase Roman numerals) should be used and, conversely, if there are not at least two sub-topics, then no subheads should be introduced. Styles named “Heading 1”, “Heading 2”, “Heading 3”, and “Heading 4” are prescribed.

## Figures and Tables

#### Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation “Fig. 1”, even at the beginning of a sentence.

1. Table Type Styles

| Table Head | Table Column Head | | |
| --- | --- | --- | --- |
| Table column subhead | Subhead | Subhead |
| copy | More table copya |  |  |

1. Sample of a Table footnote. (*Table footnote*)
2. Example of a figure caption. (*figure caption*)

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity “Magnetization”, or “Magnetization, M”, not just “M”. If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write “Magnetization (A/m)” or “Magnetization {A[m(1)]}”, not just “A/m”. Do not label axes with a ratio of quantities and units. For example, write “Temperature (K)”, not “Temperature/K”.

##### Acknowledgment *(Heading 5)*

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

##### References

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first ...”

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors’ names; do not use “et al.”. Papers that have not been published, even if they have been submitted for publication, should be cited as “unpublished” [4]. Papers that have been accepted for publication should be cited as “in press” [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

1. G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*
2. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
3. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
4. K. Elissa, “Title of paper if known,” unpublished.
5. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
6. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
7. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.
8. K. Eves and J. Valasek, “Adaptive control for singularly perturbed systems examples,” Code Ocean, Aug. 2023. [Online]. Available: <https://codeocean.com/capsule/4989235/tree>
9. D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” 2013, arXiv:1312.6114. [Online]. Available: <https://arxiv.org/abs/1312.6114>
10. S. Liu, “Wi-Fi Energy Detection Testbed (12MTC),” 2023, gitHub repository. [Online]. Available: https://github.com/liustone99/Wi-Fi-Energy-Detection-Testbed-12MTC
11. “Treatment episode data set: discharges (TEDS-D): concatenated, 2006 to 2009.” U.S. Department of Health and Human Services, Substance Abuse and Mental Health Services Administration, Office of Applied Studies, August, 2013, DOI:10.3886/ICPSR30122.v2

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.**

##### APPENDIX

## Judge Prompt

[Instruction]\nPlease act as an impartial judge

and evaluate the quality of the response provided by an AI

assistant to the user question displayed below in a trivia game.

The question is about something in the world the Harry Potter book series.

Your evaluation should consider factors such as the helpfulness, relevance,

accuracy, depth, creativity, and specially how direct the response was.

[Ground truth]\n{reference}\nBegin your evaluation

by providing a short explanation. Be as objective as possible.

After providing your explanation, you must rate the response on a scale of 1 to 10

by strictly following this format: [[rating]], for example: Rating: [[5]].

[Question]\n{input}\n\n[The Start of Assistant\'s Answer]\n{prediction}\n\

[The End of Assistant\'s Answer]',