# Making LLMs Work for Enterprise Part 1: Overview, LLM Selection & Performance Measures

## Planning / Implementation

Building an effective retrieval-augmented generation (RAG) application requires careful planning end execution. To achieve this, you'll need to follow these essential steps:

1. Choose the right large language models (LLMs) for generating synthetic data, generating responses to end users, and evaluating performance.
2. Create a comprehensive dataset that covers the range of user inputs and desired responses.
3. Fine-tune an LLM for domain-specific RAG applications to ensure accurate results.
4. Optimize embeddings models, which convert text into semantically representative numbers, to improve context retrieval.
5. Implement context retrieval enhancements for an optimal user experience.

When choosing LLMs, consider factors such as model sizing, licensing terms, the domains of previously trained on data, and context window (maximum prompt) size. Maximizing the use of larger models while adhering to available hardware restraints can significantly impact application performance. Benchmarks should also be consulted to ensure selected LLMs exhibit robust capabilities for handling RAG question answering tasks.

Throughout the development process, it's vital to quantitatively evaluate both retrieval and generation components of the RAG application using a suitable framework such as Ragas. Ragas is an open-source toolkit that uses LLMs to scalably score the accuracy of both the retrieval and generation components of RAG systems. By adhering to these guidelines and employing best practices, an enterprise LLM RAG application can deliver superior results in handling user queries while enhancing overall system efficiency. Stay tuned for the next article where we dive into creating a dataset for validating and evaluating the RAG application.

## Series Overview

This article is the first in a five-part series that covers the process of creating an enterprise LLM RAG application. The series can be followed in chronological order as a guide. Table 1 shows the main deliverables that can result from following the instructions in each article.

Table 1. Main deliverables from each article in the series

| # | Article Title | Deliverables |
|---|---|---|
| **1** | Overview, LLM Selection & Performance Measures (this article) | • Open source LLM for synthetic dataset generation<br>• Open source LLM to fine-tune<br>• Open source LLM for application performance evaluation<br>• Application performance evaluation toolkit |
| **2** | RAG Fine-tuning Dataset Creation | • Definition of scope for user inputs to the application<br>• Document collection for context in retrieval augmented generation<br>• Dataset for fine-tuning an LLM for domain-specific retrieval augmented generation |
| **3** | Generative LLM Fine-tuning for RAG (coming soon) | • Fine-tuned LLM for domain-specific retrieval augmented generation |
| **4** | Embeddings Model Fine-tuning for RAG (coming soon) | • Fine-tuned embeddings model for improved domain-specific context retrieval |
| **5** | RAG Search Enhancements (coming soon) | • Enhanced context retrieval pipeline with advanced search functions |

## LLM Selection

This process uses LLMs in three ways:

1. *Synthetic dataset generation:* An LLM is inferenced to synthetically generate a dataset.
2. *Fine-tune*: An LLM undergoes parameter-efficient fine-tuning (PEFT) and will be inferenced in production.
3. *Evaluation*: An LLM is inferenced to evaluate outputs from the RAG application.

### Model Sizing

The LLMs can be the same base model, but it is better to use a larger LLM for inference and fine-tune a smaller LLM to make the most of development resources. For example, if 1xA100 80GB GPU is available for development, that GPU can inference Llama 2 13B, but can only fine-tune Llama 2 7B, since inference requires less VRAM than fine-tuning. In general, larger models have higher performance, so it is important to maximize the model size at each step given our resources.

Most LLMs can be loaded in 16-bit datatypes (i.e., fp16 or bf16). For inference, the amount of VRAM needed, in bytes, is equal to the number of parameters multiplied by 2 (16 bits equals 2 bytes). Therefore, on a GPU with 80GB VRAM, it is possible to inference an LLM of up to 40 billion parameters. For training, the amount of VRAM needed is quadrupled compared to inference, so an LLM of up to 10 billion parameters could be fine-tuned on the same GPU.

In production, the fine-tuned LLM can be inferenced on a smaller GPU than the one used for fine-tuning, following these same guidelines.

### Benchmark Performance

Once the maximum model sizes have been determined, models should be selected that score well on relevant benchmarks. Table 2 below shows common tasks and corresponding benchmarks.

Table 2. Consider relevant tasks and select models that perform well on relevant benchmarks (David Taubenheim, The Fast Path to Developing with LLMs, NVIDIA LLM Developer Day, November 17, 2023)

| Task Type | Benchmarks | Benchmark Summary |
|---|---|---|
| **Reasoning** | HellaSwag | Common sense reasoning that is trivial for humans |
| | WinoGrande | Common sense reasoning by resolving ambiguities in sentences |
| | PIQA | Physical commonsense in everyday scenarios |
| **Reading comprehension/question answering** | BoolQ | Yes/no question answering based on questions and corresponding passages |
| | TriviaQA | Answering trivia questions based on very long context |
| | NaturalQuestions | Answering real user Google searches based on Wikipedia articles |
| **Math word problems** | MATH | Challenging competition mathematics problems |
| | GSM8K | Grade school math word problems |
| | SVAMP | Elementary-level math word problems |
| **Coding** | HumanEval | Simple interview-type software questions |
| | MBPP | Basic Python programming problems |
| **Multi-task** | MMLU | Question answering on a wide range of academic subjects |
| | GLUE | Reading comprehension and logic |
| **Separating fact from fiction in training data** | TruthfulQA | Truthfulness and avoiding common human misconceptions on a wide range of topics |

For RAG question answering applications, reading comprehension/question answering tasks are relevant, so BoolQ, TriviaQA, and NaturalQuestions are all benchmarks to consider. A good resource for seeing leader boards of high performing LLMs for notable benchmarks is https://paperswithcode.com/.

### Other Considerations

- Training type: For both synthetic data generation and fine-tuning, it is best to use base models instead of already fine-tuned models. For evaluation, an instruction-fine-tuned model (also called a “chat” model) will generally perform better.
- Context size: Each model can act on a static prompt size. Models with larger context sizes give applications the benefit of putting more context into the prompts. It is important that the synthetic dataset generation LLM has enough context size to see several examples of questions, answers, and supporting context within each prompt to generate a new data point. It is also important that the LLM for fine-tuning has a large enough context window to fit a user question and enough context to support in answering the question. In evaluation, the LLM must have similar context size as the fine-tuned LLM, since it will have to fit a question, supporting context, and answers to evaluate the full performance of the application.
- License terms: It is important to review the license of any model selected. Allowed uses of the model should be reviewed, since some model licenses may forbid using the model’s outputs to train another model.
- Domain: If the application focuses on a certain domain, such as finance or medicine, consider

models that have been pretrained with a larger portion of data in that domain.

## Performance Measures

Throughout the development process, it is important to quantitatively evaluate both the retrieval and generation components of the RAG application. The second article in this series discusses the process of creating a dataset that covers the scope of user inputs the application should be able to respond to, and the desired responses to those inputs. A subset of that dataset can be used for validation of the application.

Ragas is a framework for evaluating RAG applications that uses an LLM to score the outputs of a RAG application. The Ragas framework measures performance of retrieval and generation components independently, so developers can identify which step in the application needs improvement. Using Ragas with a validation dataset can yield the following metrics, as defined by the Ragas documentation, which should be tracked with every iteration of the application's development to measure improvements:

- Retrieval
  - Context precision - the signal to noise ratio of retrieved context
  - Context recall - can it retrieve all the relevant information needed to answer the question
- Generation
  - Faithfulness - how factually accurate is the generated answer
  - Answer relevancy - how relevant is the generated answer to the question
- End-to-end evaluation
  - Answer semantic similarity - semantic resemblance between the generated answer and the ground truth
  - Answer correctness - accuracy of the answer compared to the ground truth

It is important to measure the performance of RAG applications quantitatively to support metrics-driven development. Ragas enables the use of an LLM for measuring performance, so the process is scalable and repeatable, generating a set of actionable metrics.

## Conclusion

In conclusion, the creation of an enterprise LLM RAG application involves several crucial steps, each delivering distinct deliverables that contribute to the overall performance and effectiveness of the system. These steps include selecting appropriate LLMs for synthetic dataset generation, fine-tuning for domain-specific retrieval augmented generation, fine-tuning embeddings models, implementing search enhancements, and evaluating the application using quantitative measures.

To create an effective enterprise-level RAG application using LLMs, consider factors like model size, license terms, training methods, context capacity, and domain expertise. Maximizing larger models while respecting hardware constraints enhances performance. Benchmarks are crucial for verifying LLM suitability in handling RAG question answering tasks.

Quantitative evaluation of both retrieval and generation components is essential throughout the RAG application development process, employing frameworks such as Ragas for accurate assessment. Following these guidelines and best practices ensures informed development for continuous system improvement. In the next article, we'll discuss defining the scope of user inputs and creating and augmenting a dataset that can be used for fine-tuning an LLM.

**Read the next article in this series** , Part 2: RAG Fine-Tuning Dataset Creation .

For more information on Lenovo offerings for Generative AI, see the Reference Architecture for Generative AI Based on Large Language Models (LLMs), available from https://lenovopress.lenovo.com/lp1798-reference-architecture-for-generative-ai-based-on-large-language-models.

## Author

Chris Van Buren is a Staff Data Scientist at Lenovo. He researches generative AI for enterprise use cases and has developed retrieval augmented generation (RAG) applications with open source, on-premises LLMs.

## Related product families

Product families related to this document are the following:

- Artificial Intelligence
- ThinkSystem SR675 V3 Server
- ThinkSystem SR680a V3 Server
- ThinkSystem SR685a V3 Server

## Notices

Lenovo may not offer the products, services, or features discussed in this document in all countries. Consult your local Lenovo representative for information on the products and services currently available in your area. Any reference to a Lenovo product, program, or service is not intended to state or imply that only that Lenovo product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any Lenovo intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any other product, program, or service. Lenovo may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

Lenovo (United States), Inc.
8001 Development Drive
Morrisville, NC 27560
U.S.A.
Attention: Lenovo Director of Licensing

LENOVO PROVIDES THIS PUBLICATION ”AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. Lenovo may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

The products described in this document are not intended for use in implantation or other life support applications where malfunction may result in injury or death to persons. The information contained in this document does not affect or change Lenovo product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of Lenovo or third parties. All information contained in this document was obtained in specific environments and is presented as an illustration. The result obtained in other operating environments may vary. Lenovo may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any references in this publication to non-Lenovo Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this Lenovo product, and use of those Web sites is at your own risk. Any performance data contained herein was determined in a controlled environment. Therefore, the result obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.



This document, LP1953, was created or updated on May 2, 2024.

Send us your comments in one of the following ways:

- Use the online Contact us review form found at:
  https://lenovopress.lenovo.com/LP1953
- Send your comments in an e-mail to:
  comments@lenovopress.com

This document is available online at https://lenovopress.lenovo.com/LP1953.

## Trademarks

Lenovo and the Lenovo logo are trademarks or registered trademarks of Lenovo in the United States, other countries, or both. A current list of Lenovo trademarks is available on the Web at https://www.lenovo.com/us/en/legal/copytrade/.

The following terms are trademarks of Lenovo in the United States, other countries, or both:
Lenovo®

Other company, product, or service names may be trademarks or service marks of others.