machine learning & knowledge extraction

MDPI

Article

Investigating the Performance of Retrieval-Augmented Generation and Domain-Specific Fine-Tuning for the Development of AI-Driven Knowledge-Based Systems

Róbert Lakatos [1,2,3,*], Péter Pollner [4], András Hajdu [1] and Tamás Joó [3,4,*]

[1] Department of Data Science and Visualization, Faculty of Informatics, University of Debrecen, 4032 Debrecen, Hungary; hajdu.andras@inf.unideb.hu
[2] Doctoral School of Informatics, University of Debrecen, 4032 Debrecen, Hungary
[3] Neumann Technology Platform, Neumann Nonprofit Ltd., 1074 Budapest, Hungary
[4] Data-Driven Health Division of National Laboratory for Health Security, Health Services Management Training Centre, Semmelweis University, 1085 Budapest, Hungary
* Correspondence: lakatos.robert@inf.unideb.hu (R.L.); joo.tamas@emk.semmelweis.hu (T.J.)

Abstract: Generative large language models (LLMs) have revolutionized the development of knowledge-based systems, enabling new possibilities in applications like ChatGPT, Bing, and Gemini. Two key strategies for domain adaptation in these systems are Domain-Specific Fine-Tuning (DFT) and Retrieval-Augmented Generation (RAG). In this study, we evaluate the performance of RAG and DFT on several LLM architectures, including GPT-J-6B, OPT-6.7B, LLaMA, and LLaMA-2. We use the ROUGE, BLEU, and METEOR scores to evaluate the performance of the models. We also measure the performance of the models with our own designed cosine similarity-based Coverage Score (CS). Our results, based on experiments across multiple datasets, show that RAG-based systems consistently outperform those fine-tuned with DFT. Specifically, RAG models outperform DFT by an average of 17% in ROUGE, 13% in BLEU, and 36% in CS. At the same time, DFT achieves only a modest advantage in METEOR, suggesting slightly better creative capabilities. We also highlight the challenges of integrating RAG with DFT, as such integration can lead to performance degradation. Furthermore, we propose a simplified RAG-based architecture that maximizes efficiency and reduces hallucination, underscoring the advantages of RAG in building reliable, domain-adapted knowledge systems.

Keywords: generative large language model; domain-specific fine-tuning; knowledge-based system; natural language processing; machine and deep learning

check for updates

Academic Editor: Daniel E. O'Leary

Received: 25 November 2024
Revised: 22 January 2025
Accepted: 6 February 2025
Published: 10 February 2025

Citation: Lakatos, R.; Pollner, P.; Hajdu, A.; Joó, T. Investigating the Performance of Retrieval-Augmented Generation and Domain-Specific Fine-Tuning for the Development of AI-Driven Knowledge-Based Systems. *Mach. Learn. Knowl. Extr.* **2025**, *7*, 15. https://doi.org/10.3390/make7010015

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Transformer-based large language models (LLMs) have significantly advanced the field of natural language processing (NLP) by achieving unprecedented performance across various linguistic tasks. Notable models such as BERT [1], GPT [2], and XLNet [3], along with their respective derivatives, have set new benchmarks in areas like text classification, machine translation, and question-answering. Among these, generative models, particularly GPT, have demonstrated outstanding capabilities in comprehending and generating human-like text [4]. OpenAI's GPT architecture, as seen in widely adopted services like ChatGPT [5] and Microsoft's Copilot [6], has become a benchmark for generative performance, while competing models such as Google's PaLM [7], Gemini [8], and Meta's open-source LLaMA model [9] are emerging as potent alternatives.

The most advanced iterations of these LLM families have showcased remarkable comprehension and text-generation capabilities. As a result, LLMs have redefined NLP by excelling in various tasks, from generating coherent text to translating languages and answering complex questions. For instance, these models dominate in benchmarks such as the Massive Multitask Language Understanding (MMLU) dataset [10], demonstrating their superior ability to leverage pre-training knowledge. Trained on vast corpora of text and code, LLMs capture intricate linguistic patterns, enabling them to produce text that closely mirrors human writing. Beyond general tasks, their full potential becomes evident when applied to specific domains, as illustrated by the FunSearch algorithm [11], where LLMs are even employed to solve mathematical problems.

This surge in LLM capabilities has profoundly influenced the evolution of knowledge-based systems. Historically, such systems depended on structured databases and traditional querying methods to extract information [12]. However, the contextually aware text generation capabilities of LLMs present a new paradigm, extending the potential of these systems beyond mere data retrieval to performing reasoning and knowledge synthesis. This paradigm shift enables more advanced applications, particularly in question-answering and information-retrieval tasks [13].

Two primary approaches have emerged to tailor LLMs for domain-specific applications: domain-specific fine-tuning (DFT) and retrieval-augmented generation (RAG). DFT allows models to specialize in particular areas by training in domain-specific datasets, improving performance in tasks such as machine translation [14] and domain-relevant applications [15]. RAG complements this by integrating external information from relevant datasets into the prompt, enhancing the accuracy of the model and reducing the need for frequent retraining [16]. The primacy of these two approaches is further strengthened by the fact that in fierce development competition, the importance of DFT was demonstrated by the ChatGPT service. In contrast, the Copilot system showed RAG's significance. ChatGPT was the first LLM-based service capable of performing several language tasks with significant domain knowledge. Although Copilot was among the first to ensemble Microsoft's web search engine with an LLM using RAG.

However, integrating these techniques into knowledge-based systems remains a complex task. Standardized methodologies are needed for designing systems that leverage LLMs, particularly in domain-specific contexts. This gap underscores the need for further research and clear guidelines. In response to these challenges, this study aims to propose a structured methodology for constructing knowledge-based systems utilizing LLMs.

Section 2 details the databases used for parameter optimization, configuration, and method selection. Section 3 outlines our experimental setup, followed by an in-depth discussion of our methodology in Section 4. Empirical results demonstrating the performance of various approaches are presented in Section 5, while Sections 6 and 7 synthesize the essential findings and provide our conclusions and discussions.

2. Data

We applied two approaches to preparing the data. First, we examined how to create datasets for PDF and Microsoft Word-based scientific publications. The primary motivation for creating our datasets was to build our system, the effectiveness of which we can support with measurement results. Second, we composed another dataset for the measurements besides the own-created data. This second dataset was collected from publicly available data.

For the dataset we created, we curated a collection of specific publications on urban monitoring and corn cultivation with the help of the National Library of the University of Debrecen and the Faculty of Agriculture, Food Science, and Environmental Management.

This corpus, comprising 69 literature works on corn cultivation (CORN) and 83 on urban monitoring (UB), provided a rich source of domain-specific terminology and concepts. Each article or book was available to us in PDF or Word format.

As an independent and open-access dataset, we utilized the CORD-19 [17], and MedQuAD [18,19] datasets. CORD-19 is a freely available repository of thousands of scientific articles on COVID-19, SARS-CoV-2, and related coronaviruses. This dataset encompasses thousands of scholarly articles. It is in JSON format and represents about 80 GB of text data. MedQuAD includes 47457 medical question-answer pairs created from 12 NIH websites (e.g., cancer.gov, niddk.nih.gov, GARD, MedlinePlus Health Topics). The collection covers 37 question types (e.g., Treatment, Diagnosis, Side Effects) associated with diseases, drugs, and other medical entities such as tests.

3. Experimental Setup

For the design and testing of our solution, we used an NVIDIA DGX A100 workstation provided by the Faculty of Informatics of the University of Debrecen. The workstation has a Dual AMD Rome 7742 CPU with 128 cores total (Advanced Micro Devices, Inc., Santa Clara, CA, USA), 2 TB system memory, 30 TB of storage, and 8 × 80 GB NVIDIA A100 GPUs (NVIDIA Corporation, Santa Clara, CA, USA). We always worked with two GPU units while the CPU and memory were used dynamically.

4. Methodology

To decide whether RAG or DFT is the better approach to create an LLM-based system, we used the models presented in Section 4.1. The performance of these models was measured using the metrics given in Section 4.2 for both RAG and DFT applications. For these measurements, we describe in Section 4.3 how we prepared the data, and in Sections 4.4 and 4.5, we show which settings were applied for DFT and RAG.

4.1. Models

We have selected the LLM models according to the following requirements:

- The models must be LLM, well-documented, and have freely downloadable pre-trained versions.
- The models have been well implemented. That means they should be part of the model repository of the HuggingFace [20] and PyTorch [21] development libraries.
- A resource provided by the NVIDIA DGX A100 System (see Section 3) should be sufficient to train and test the models.

Based on these criteria, we selected the GPT-J-6B [22], OPT-6.7B [23], LLaMA-7B [9], and LLaMA2-7B [9] models.

4.2. Selected Metrics

The following metrics were used to determine the performance of the different language models: Bilingual Evaluation Understudy (BLEU) [24], Recall Oriented Understudy for Gisting Evaluation (ROUGE) [25], Metric for Evaluation for Translation with Explicit Ordering scores (METEOR) given by [26], and cosine similarity calculated as

$$Cosine(x,y) = \frac{xy}{\|x\|\|y\|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} (x_i)^2}\sqrt{\sum_{i=1}^{n} (y_i)^2}}, \quad (1)$$

for $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$.

BLEU is used to measure machine translation performance. BLEU measures n-gram accuracy, which means it counts how many n-grams of the generated text are found in the

reference translation. ROUGE is used to measure the performance of machine translation and text summarization tasks and measures recall, which means that it counts how many n-grams of the reference translation are found in the generated text. ROUGE is designed to work around some of BLEU's limitations. Namely, ROUGE places more emphasis on recall than BLEU and better takes into account the meaning of the text. The METEOR score is a metric used to evaluate the quality of machine translation, text summaries, and creative text formats. It considers the Recall, Precision, and word order of factors to determine how efficient a model is.

Cosine similarity is also used to measure the similarity of texts. To use it, the text must be converted into sentence or word vectors, then the cosine similarity between the vectors must be calculated. A higher cosine similarity means that the texts are more similar semantically. We applied this approach by dividing the generated and reference text into sentences with our system and then converting the individual sentences into embedded vectors using the MiniLM L6 v2 [27] sentence transformer.

We introduced this special application of cosine similarity because models for uniformly embedding full texts (document embedding) are not as efficient and sophisticated as word and sentence embedding models. In general, document-level embedding models tend to lose finer-grained semantic information in individual sentences, leading to lower accuracy when comparing texts with significant internal structural variation [28]. Therefore, we have chosen the following approach. ROUGE, BLEU, and METEOR scores were used for word-level measurements, while cosine similarity was used for sentence-level measurements. However, we had to embed the texts sentence by sentence and develop a measurement formula that compares the generated text and the reference texts sentence by sentence. This task is solved with our CS formula.

For a formal description of our method to apply cosine similarity, let $R = \{r_1, \ldots, r_M\}$ and $G = \{g_1, \ldots, g_M\}$ denote the reference and generated text, respectively, broken down to the same number of M corresponding paragraphs r_i and g_i with $i = 1, \ldots, M$. All the reference and generated paragraphs consist of sentences represented by their embedded vectors $r_i = \{rs_{i,1}, rs_{i,2}, \ldots\}$ and $g_i = \{gs_{i,1}, gs_{i,2}, \ldots\}$ for $i = 1, \ldots, M$. For each reference sentence $rs_{i,j}$ we find the best matching generated sentence in the corresponding paragraph g_i and calculate their similarity. Finally, as an overall similarity score or Coverage Score (CS) between the reference text R and the generated one G, we calculate the average similarity regarding all the composing paragraphs as

$$CS = \sum_{i=1}^{M} \frac{\max_k \left(Cosine(rs_{i,j}, gs_{i,k}) \right)}{M}. \quad (2)$$

ROUGE, BLEU, METEOR, and cosine similarity are key metrics for evaluating language models in real-world applications. ROUGE ensures content coverage, making it useful for summarization and QA systems. BLEU focuses on n-gram precision, which is critical for translation and grammatically coherent text generation. METEOR balances precision, recall, and word order, supporting creative text and personalized recommendations. Cosine similarity captures semantic meaning by comparing text embeddings, enabling applications like semantic search, clustering, and recommendation engines. These metrics provide a robust framework for assessing and improving language models in practical scenarios.

4.3. Data Preparation for RAG and DFT

We used different data preparation approaches for the RAG and DFT. For DFT, we considered the Q&A dataset-based training method (as it was used for the Stanford Al-

paca [29]) as a guiding principle. In addition, we created easily searchable datasets for RAG to identify contexts that support effective responses.

4.3.1. Q&A Datasets for DFT

To prepare Q&A datasets, we split the collected CORN and UB datasets into paragraphs. In the next step, we converted them to raw text and cleaned them with the help of human experts.

Regarding CORD-19 data, we extracted a subset from it. We selected the journal papers based on the following filter criteria:

- The papers must have abstracts.
- They must be included in the PubMed Central repository. The papers must have open access to medical biology and life science types.
- They must not contain latex elements, so they can also be readable easily and validated by human experts.

In the MedQuAD dataset, there are incomplete question-answer pairs. Therefore, we kept only those records where the questions and their corresponding answers, types, and focus values were fully available.

We divided our datasets CORN, UB, and CORD-19 into paragraphs, taking into account each model's tokenizer. When dividing the paragraphs, we worked so that the individual text parts could not be longer than 256 tokens.

The use of texts no longer than 256 tokens was important to us because the text generation time of language models is primarily determined by the number of tokens to be generated. We used the hypothesis that texts consisting of 256 tokens are long enough for sophisticated measurement. Furthermore, the generation time required for the measurement was still manageable within human time with the limits of our available resources.

To create the questions of the Q&A dataset, we used the BERT-based generator [30]. We generated five questions for each paragraph. Duplicates were filtered and removed to ensure the two questions differed for the same paragraph. Thus, we created a maximum of five but at least one question in the database for each paragraph. With this, we applied an oversampling.

The MedQuAD data had expert-validated question-answer pairs. Therefore, to ensure that they could not be semantically damaged, but in terms of their length, they are the same as the length of the answers included in the other experimental data sets, we discarded those pairs where the answers were longer than 256 tokens.

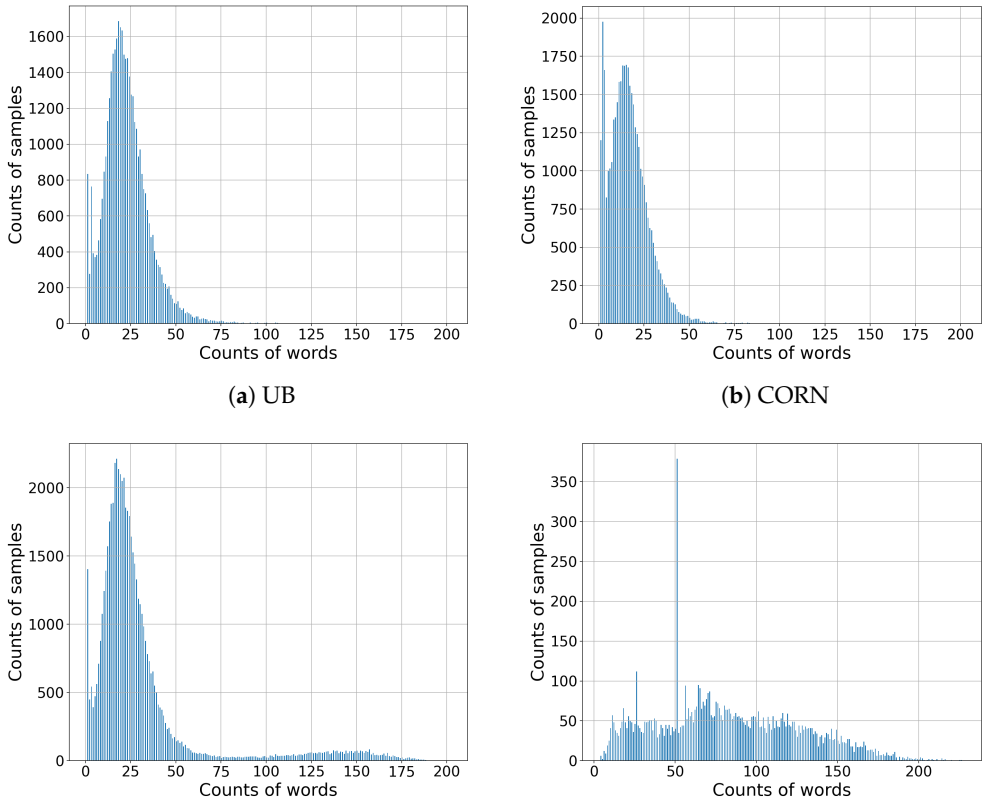
We also created a context-extended version for each dataset using a simple method. For each question, we created a context-enriched version and a version without context. Thus, each question and answer pair was represented in the dataset in two copies: one with context and one without. The goal was to allow the model to learn to answer based on context while still being able to answer correctly without context.

To build the contexts, we randomly selected four definitely incorrect answers from the data set and then added the correct answer to this list. Thus, the list of five items contained one correct and four incorrect answers, and the correct answer was always at the beginning of the list, as expected from a semantic search when we applied RAG. Finally, we inserted the context thus created beside the question according to the appropriate prompt template to simulate the RAG behavior.

Table 1 lists the number of paragraphs and questions in the created Q&A ($Q\&A_{CORN}$, $Q\&A_{UB}$, $Q\&A_{CORD\text{-}19}$, $Q\&A_{MedQuAD}$) datasets, and Figure 1 shows the distribution of the number of words in the text for each dataset.

Table 1. Number of paragraphs and questions of Q&A datasets.

Dataset	Paragraphs	Questions
$Q\&A_{CORN}$	7058	28,790
$Q\&A_{UB}$	8553	27,974
$Q\&A_{CORD\text{-}19}$	18,004	58,290
$Q\&A_{MedQuAD}$	7871	7871



(**a**) UB (**b**) CORN

(**c**) CORD-19 (**d**) MedQuAD

Figure 1. Distributions of the number of words in the texts by dataset.

4.3.2. RAG Datasets

The performance of RAG is highly contingent on the precision of the contextual information used to generate answers. To rigorously assess RAG's capabilities, we employed two distinct methodologies.

To generate the datasets used for the RAG, we used the $Q\&A$ datasets created from the CORN, UB, CORD-19, and MedQuAD datasets. We ensured that each paragraph (answers) was accompanied by at least one corresponding question in these datasets. Each generated question was transformed into a vector representation using the MiniLM L6 v2 sentence transformer [31]. This allowed us to measure semantic similarity using our CS metric. In this setup, answers were drawn from paragraphs containing questions most similar to the reference, forming our first indexed dataset type (ID_q).

In the second approach, we segmented each answer into individual sentences and embedded these sentences using the same MiniLM L6 v2 sentence transformer. We excluded sentences shorter than 10 words or longer than 30 to optimize embedding efficiency. This enabled us to manage each sentence as a vectorized index, forming our second type of indexed dataset (ID_s).

For all the datasets CORN, UB, CORD-19, and MedQuAD, both types of indexed datasets (ID_q and ID_s) were created. Table 2 summarizes their properties.

Table 2. Number of vectorized sentences and questions of indexed datasets.

Dataset	Sentences (ID_s)	Questions (ID_q)
$Q\&A_{CORN}$	37,874	28,790
$Q\&A_{UB}$	40,002	27,974
$Q\&A_{CORD\text{-}19}$	56,861	58,290
$Q\&A_{MedQuAD}$	32,310	7871

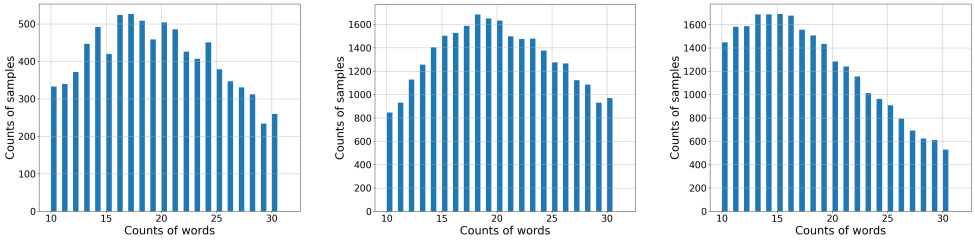
4.3.3. Training, Validation, and Test Datasets

We divided the $Q\&A_{CORN}$, $Q\&A_{UB}$, $Q\&A_{CORD\text{-}19}$, and $Q\&A_{MedQuAD}$ datasets into training and validation sets in an 80/20 ratio. Due to our question-and-answer generation methodology, a single answer may correspond to multiple similar questions. Therefore, we ensured that each question and its corresponding answer appeared only once in the validation dataset.

To evaluate the models' inference abilities, we created distinct test datasets. For this, we used questions and corresponding answers unseen by the models during fine-tuning. We employed topic modeling based on nested vectors to achieve this, generating three test datasets from $Q\&A_{CORN}$, $Q\&A_{UB}$, and $Q\&A_{CORD\text{-}19}$.

In the first step, we embed all sentences in the ID_s datasets with the Sentence Transformer. We then reduced the dimensionality of these embeddings from 386 to 2 using UMAP [32] and subsequently clustered them using HDBSCAN [33]. The clustering parameters were set to a maximum of 15 clusters and a minimum of 6 because the optimal results were obtained when the clusters contained approximately 256 tokens from sentences of 10 to 30 words. Texts outside this word range were excluded. Figure 2 demonstrates the distribution changes resulting from this filtering process.

After the mapping and clustering, the outlier clusters were removed. We then measured the exact number of tokens contained in each cluster with the tokenizer of each model, and we removed clusters with more than 256 tokens.



(**a**) CORD-19 (**b**) UB (**c**) CORN

Figure 2. Distributions of texts between 10 and 30 word counts by dataset.

The final evolution of the number of words and the counts of tokens per model can be observed for each dataset in Table 3.

Table 3. Maximum number of tokens and words per model for responses in each dataset.

Dataset	Words	Tokens of OPT	Tokens of GPTJ	Tokens of LLaMA	Tokens of LLaMA2
$Q\&A_{CORN}$	181	238	237	256	256
$Q\&A_{UB}$	183	238	237	256	256
$Q\&A_{CORD\text{-}19}$	186	242	241	256	256

Finally, we generated questions for the created clusters. The test data created contained 279 (CORN), 311 (CORD-19), and 249 (UB) question-answer pairs. The result of dimensionality reduction and clustering can be seen in Figures 3 and A6.

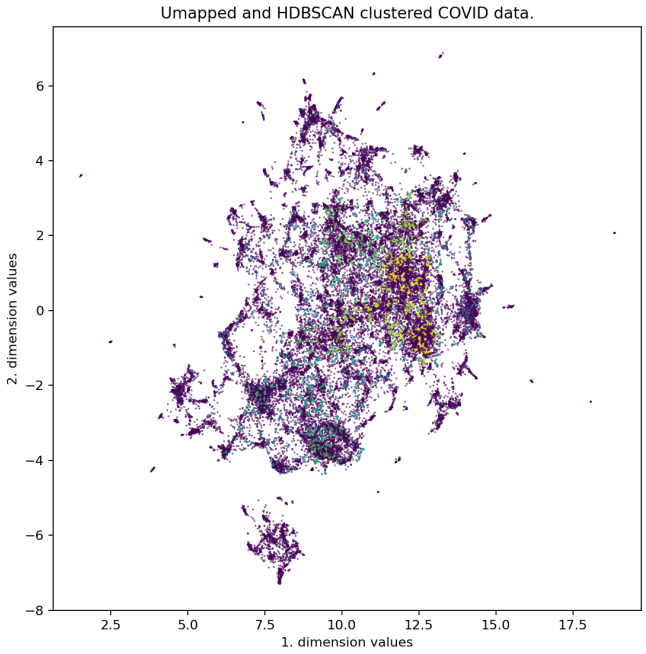


Figure 3. Two-dimensional UMAP representation of the CORD-19 dataset.

We utilized the predefined question types for the $Q\&A_{MedQuAD}$ dataset, which consists of expert-curated question-answer pairs, to generate the training, validation, and test datasets. Specifically, 80% of each question type was allocated to the training set, while 20% was reserved for validation and testing. From this 20%, we selected 300 entries for the test dataset, ensuring its size was comparable to the other test datasets.

4.4. Fine Tuning Settings

We fine-tuned our models using standard Hugging Face training code with the following belief hyperparameters in the case of all models:

- loss function: categorical cross-entropy,
- batch size: 4,
- learning rate: 2×10^{-4}
- epochs: 5,
- max new token length: 256.

4.5. RAG Settings

We use the LLAMA-2-7b model for RAG, which has been fine-tuned by instruction and human reinforcement. Therefore, it is well suited for RAG. This is because the models created in this way train on data that use prompt templates. Prompt templates distinguish different parts of the text well, making the model more sensitive to which part of the context it uses to respond. We did not find any variants from the GPT-J-6b and OPT-6.7b models that underwent reliable instructional fine-tuning, so we excluded them from the measurement of the RAG performance. LlaMA-2-7b has a maximum input size of 4096 tokens. The size of the input determines the size of the attachable context. For this reason, we introduced a limit filter to the size of the context.

The filter's task is to truncate search results that result in an input to the model larger than 4096 tokens. The input length depends on the question's length and context. Therefore,

the filter dynamically manages the length of the allowed context as a function of the size of the question.

Furthermore, a threshold parameter is introduced to control the number of context items found. The possible values of the threshold parameter range between 0 and 1, and context items with a cosine similarity below the threshold are automatically dropped. The threshold was applied with different values during the evaluation, as discussed below.

5. Results

Our evaluation strategy was to measure the models' ROUGE, BLEU, and METEOR scores. Moreover, we calculated the CS of the generated responses compared to the reference responses according to (1), and (2). In addition, during the evaluation, we followed the following strategies for measuring fine-tuned models and RAG.

We fine-tuned the GPT-J-6b, OPT-6.7b, LlaMA-7b, and LlaMA-2-7b models with the datasets $Q\&A_{CORN}$, $Q\&A_{UB}$, $Q\&A_{CORD\text{-}19}$, $Q\&A_{MedQuAD}$ to make them domain-specific. For the DFT of the models, we measured the validation accuracy at the end of each epoch and saved the models. We only evaluated the best-performing models on the test datasets. To do this, we passed all the questions from the test datasets to the most accurate models. Finally, we calculated BLEU, ROUGE, and METEOR scores and CS values between the responses generated by the models and the reference responses.

The DFT process was conducted as outlined in Section 4.4. The models consistently achieved their lowest validation loss between epochs 2 and 4 during the fine-tuning. The LlaMA-7b model was the best on the $Q\&A_{CORN}$ and $Q\&A_{CORD\text{-}19}$ datasets. On the $Q\&A_{UD}$ and $Q\&A_{MedQuAD}$ datasets, LlaMA-2-7b performed the best. The GPT-J-6b and OPT-6.7b models learned with a higher validation loss than the LlaMA models. The DFT results' measurements are given in more detail in Table A1. Further, the fine-tuning training and evaluation curves are presented in the Appendix B Figures A2–A5.

In the evaluation of RAG, the context injected and the content of the context are critical. We defined the filtering by a threshold value based on cosine similarity. The threshold value specified what was considered relevant information during the search in the dataset. As described in Section 4.3 dealing with data preparation, we worked with two types of datasets (ID_q, ID_s). The measurements were made for all datasets. The threshold values were defined on a scale from 0 to 1 with a step of 0.1. This meant that in the case of any question, we discarded matches worse than the threshold value. For example, in the case of a threshold value of 0.5 and a given question taken from the test dataset, only those paragraphs (ID_q) or sentences (ID_s) passed the filter whose indices showed a cosine similarity greater than 0.5 compared to the reference question. This also means that in the case of a threshold of 0, everything, and in the case of a threshold of 1, only the 100% semantic match is accepted. The sentences (ID_s) or paragraphs (ID_q) that passed the filter were packaged in a uniform context in descending order of similarity and passed to the model to try to answer the given question based on it. If the size of the packed context was more significant than the input of the given model allowed, the context was cut off at the maximum input size.

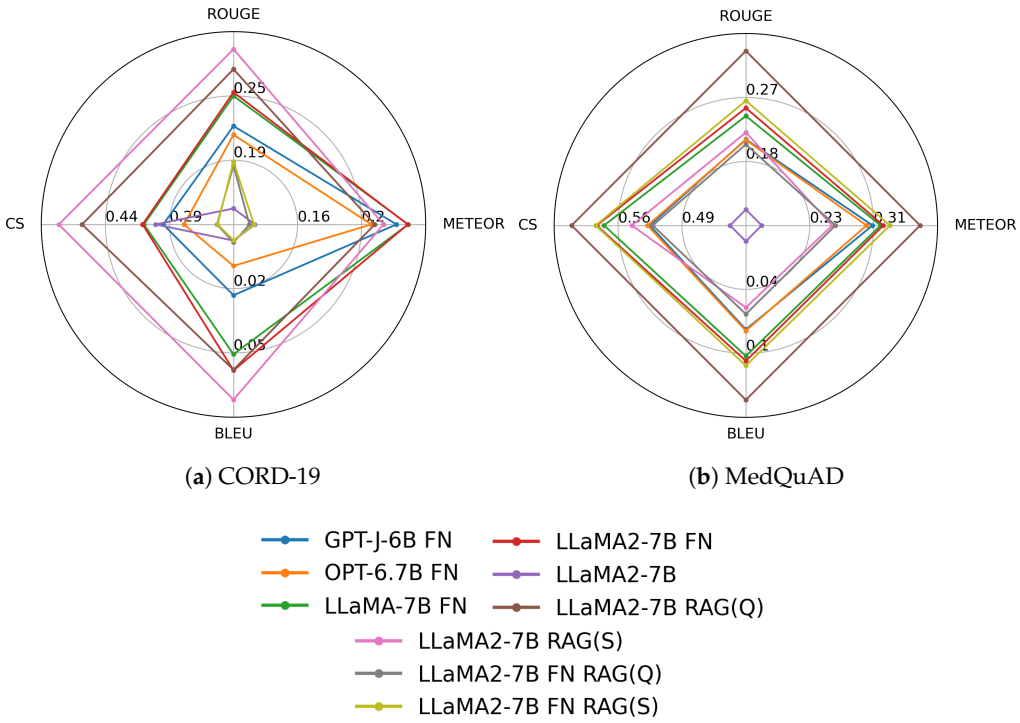
Through these rules, we controlled the size and quality of the context. We generated answers to all questions in the test dataset using all the indexed databases $ID_q^{CORN,UB,CORD-19,MedQuAD}$ and $ID_s^{CORN,UB,CORD-19,MedQuAD}$. Finally, we calculated BLEU, ROUGE, and METEOR scores and CS values between the responses generated by the models and the reference responses.

We summarize our measurement results as radar plots in Figures 4 and A1, illustrating the models' relative performance. Furthermore, Table 4 presents the average performance of each model approach. The results in Table 4, and Figures 4 and A1 show that both

DFT and RAG outperformed the baseline, i.e., the version of LlaMA-2-7b run without DFT and RAG. Furthermore, it can be said that overall, the RAG approach without DFT performed the best because models subjected to DFT could no longer produce better results using RAG.

Table 4. Average scores of each approach.

Models	ROUGE	METEOR	BLEU	CS
Baseline	0.135164	0.129689	0.002965	0.362269
DFT	0.247349	0.260628	0.060082	0.404662
RAG	0.289601	0.245677	0.068024	0.551685
RAG with DFT	0.229914	0.220869	0.043652	0.366762



(**a**) CORD-19 (**b**) MedQuAD

Figure 4. Radar plot of the evaluation results of the CORD-19 and MedQuAD models.

This is further evidenced by the optimal threshold parameters observed during the application of RAG. The best thresholds for the base LlaMA-2-7b model were an average of 0.5, whereas, for the DFT LlaMA-2-7b model, the optimal threshold was 1.0, effectively representing a 100% rejection rate. That is, the context injection could no longer help with domain-specific models. Therefore, though the domain-specific fine-tuning provides additional knowledge to the model, in return, it may impair its ability to interpret context, and it is critical for RAG. As a result, compiling a data set that conveys domain-specific knowledge that does not impair the ability of the original model to extract context is a much more difficult task than finding and passing on the appropriate context data to a properly pre-trained model. Moreover, this requires significantly less computing resources. As a result, its ecological footprint is also smaller.

It is important to note that the METEOR and BLEU scores of the fine-tuned models were better than those of the RAG models, but in terms of the ROUGE score, they were already inferior compared to the RAG. Furthermore, the RAG produced a significantly better CS than the fine-tuned models. This shows that RAG significantly improves hallucination, and although the association skills of fine-tuned models may be better, the degree of hallucination of fine-tuned models is more significant.

The best result for the test dataset was obtained using the RAG Llama-2-7b base model with the ID_s dataset. The best approaches' results are ROUGE 0.3, METEOR 0.25, BLEU 0.07, and CS 0.55. The best setup is presented in detail in Figure A7.

6. Discussion

Our research provides insights into comparing RAG and DFT methods, but we need to highlight the limitations of our research results to advance further research. Due to the limitations of our computational resources, the study worked with data sets of limited size. Also, an important limitation was the 256-token text limit, which does not adequately represent potential performance differences resulting from longer document processing.

Although the models used (GPT-J-6B, OPT-6.7B, LLaMA, LLaMA-2) were state of the art at the time of the research, rapid developments mean that models are changing rapidly. However, it is important to note that, at present, there are no significant structural changes in the evolution of the models at a deep architectural level.

The results mainly apply to the following application areas: creative tasks, assistant-like conversations, and knowledge-based applications. In these areas, the relationship between context and text generation may follow different patterns. Our highlighted limitations indicate the need for further research with larger datasets, longer context windows, or newer models.

7. Conclusions

In this study, we have shown that Retrieval-Augmented Generation consistently outperforms Domain-Specific Fine-Tuning across several evaluation metrics, including ROUGE, BLEU, and our Coverage Score (CS), making RAG a more practical approach for building LLM-based knowledge-based systems.

Since CS is based on cosine similarity, applied sentence by sentence, higher CS values indicate that RAG's context-based generation significantly improves the content match with the reference, underscoring the sensitivity of the self-attention mechanism in transformer-based architectures.

In addition, the enhanced performance of RAG stems from its ability to leverage efficient semantic search within indexed databases as well, allowing it to retrieve relevant information with higher accuracy. This reduces hallucinations and improves the overall reliability of the model's responses, particularly in knowledge retrieval tasks, offering a clear advantage over the more resource-intensive and specialized DFT approach.

However, despite its strengths, further research is required to explore optimal strategies for integrating RAG and DFT effectively and to evaluate their performance in more complex and diverse tasks. This study's findings demonstrate the potential of RAG-based architectures in advancing the development of robust, domain-adapted knowledge-based systems but also underline the challenges of striking a balance between creativity and factual accuracy in large language models.

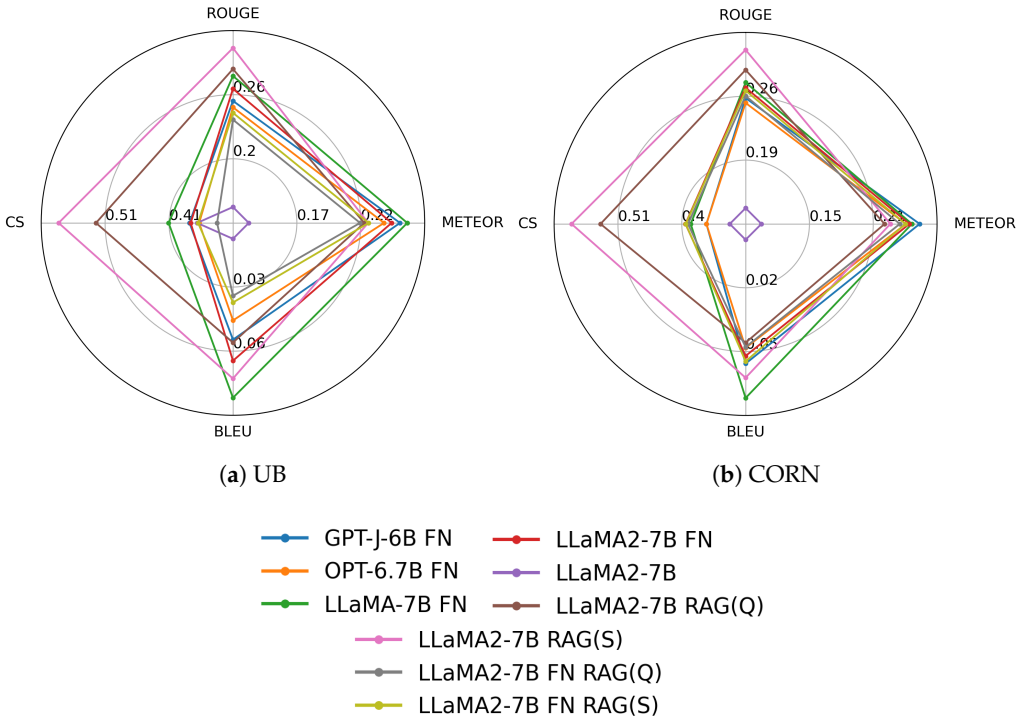
Author Contributions: R.L.: Conceptualization, Investigation, Methodology, Software, Writing—original draft. P.P.: Supervision, Validation, Writing—review & editing. A.H.: Supervision, Writing—review & editing. T.J.: Supervision. All authors have read and agreed to the published version of the manuscript.

Funding: The project no. KDP-2021 has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development, and Innovation Fund, financed under the C1774095 funding scheme. In addition, the study received further funding from the National Research, Development and Innovation Office of Hungary grant RRF-2.3.1-21-2022-00006 (Data-Driven Health Division of National Laboratory for Health Security).

Data Availability Statement: One part of the data used is confidential so the authors do not have permission to share this data. To the public part of the data and codes, the authors have shared on GitHub [34].

Conflicts of Interest: R.L. and T.J. were employed by Neumann Technology Platform, Neumann Nonprofit Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A. Additional Measurement Results



(**a**) UB (**b**) CORN

Figure A1. Radar plots of the evaluation results of the CORD-19 and MedQuAD models.

Table A1. The loss of the models measured on the base (B) and context-extended (C) validation datasets ($Q\&A_{CORN}$, $Q\&A_{UB}$, $Q\&A_{CORD\text{-}19}$, $Q\&A_{MedQuAD}$) by epochs.

Epoch	GPT-J-6b	OPT-6.7b	LLaMA-7b	LLaMA2-7b	
	B	B	B	B	C
			$Q\&A_{CORN}$		
1	0.416608	0.596310	0.384520	0.388159	0.157360
2	0.196132	0.216316	0.162778	0.181099	0.135747
3	0.163878	0.163674	0.144640	0.149348	0.138408
4	0.159600	0.153637	0.144515	0.149162	0.153498
5	0.168813	0.155910	0.154746	0.156936	0.164009
			$Q\&A_{UB}$		
1	0.511894	0.825766	0.447366	3.398389	0.343479
2	0.209409	0.258804	0.180724	0.819327	0.192512
3	0.170602	0.171143	0.150210	0.186827	0.191028
4	0.164166	0.159860	0.153346	0.145882	0.202750
5	0.172908	0.161635	0.162520	0.150440	0.213680
			$Q\&A_{CORD\text{-}19}$		
1	0.586879	1.618626	0.678659	0.488456	0.290350
2	0.238213	0.471962	0.218672	0.217865	0.219062
3	0.192331	0.227678	0.182879	0.187428	0.216804
4	0.186943	0.190710	0.185803	0.187884	0.230001
5	0.194221	0.187811	0.195959	0.198900	0.241326

Table A1. *Cont.*

Epoch	GPT-J-6b	OPT-6.7b	LLaMA-7b	LLaMA2-7b	
	B	B	B	B	C
		$Q\&A_{MedQuAD}$			
1	1.026613	1.180150	1.032789	0.820101	0.608959
2	0.948013	1.002593	0.802833	0.782055	0.563524
3	1.142371	1.024832	0.852340	0.930519	0.640563
4	1.300829	1.096023	1.060206	1.061301	0.715862
5	1.385511	1.191012	1.175565	1.182239	0.776762

Appendix B. Training and Evaluation Curves

The domain-specific fine-tuning processes training and evaluation curves can be seen in Figures A2–A5 by model and for each dataset.

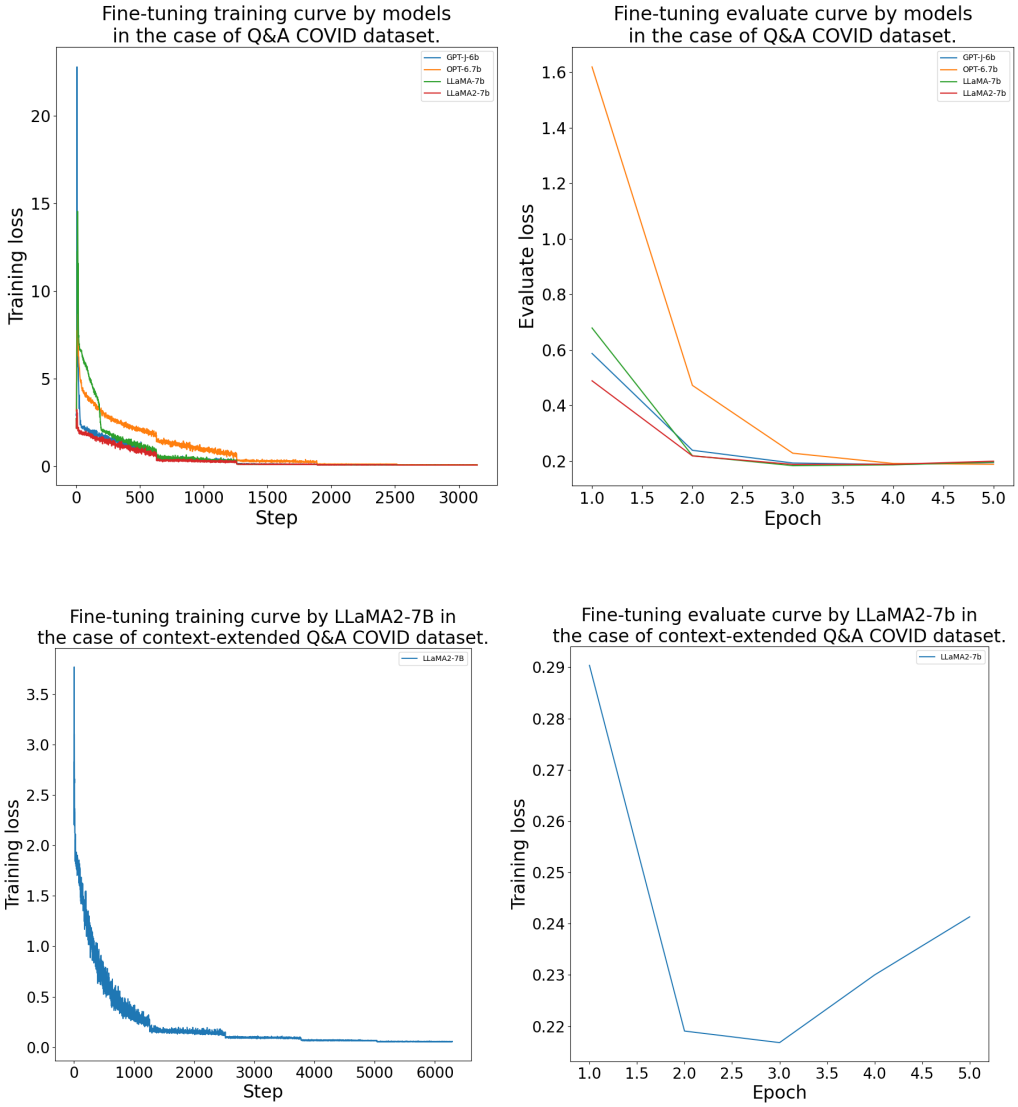


Figure A2. Domain-specific fine-tuning training and evaluation curves by model for the CORD-19 dataset.

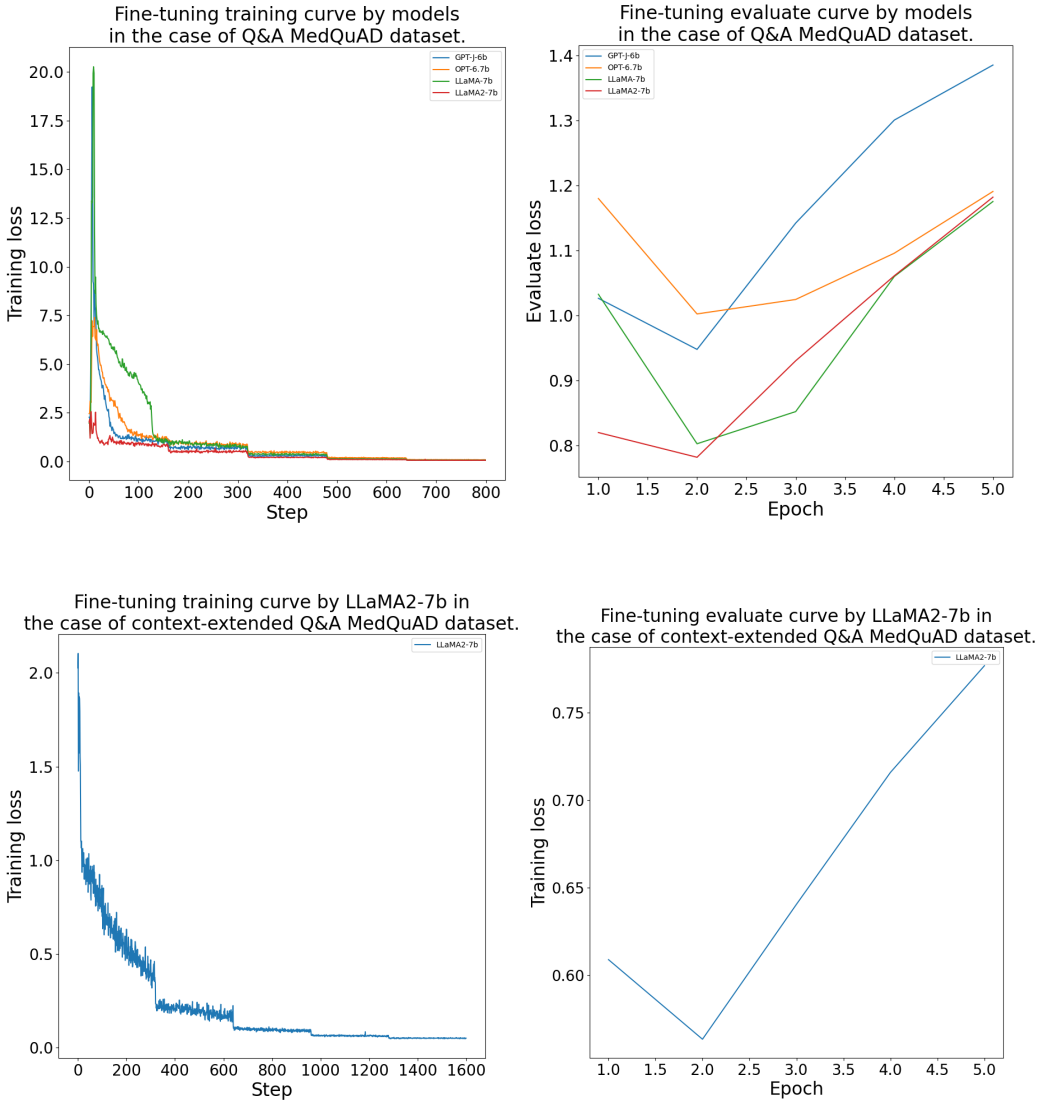


Figure A3. Domain-specific fine-tuning training and evaluation curves by model for the MedQuAD dataset.

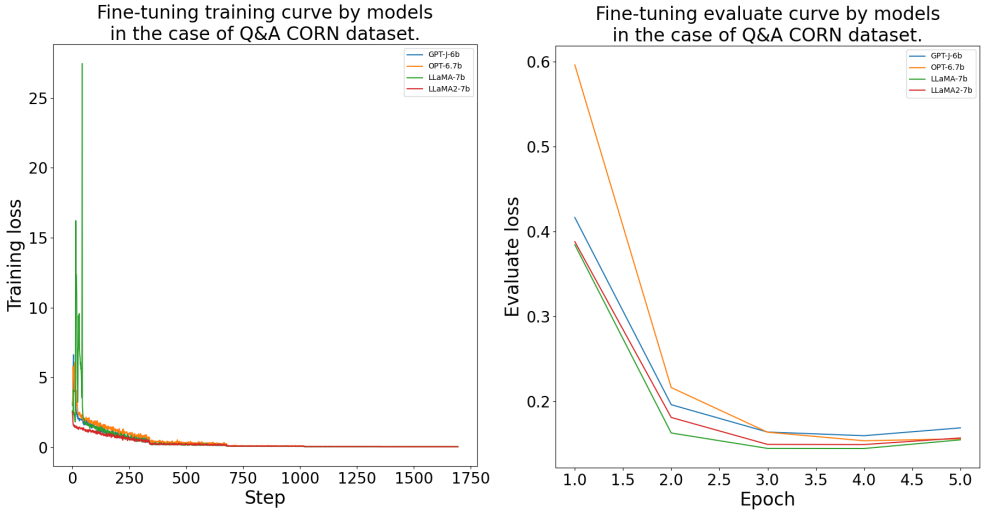


Figure A4. *Cont.*

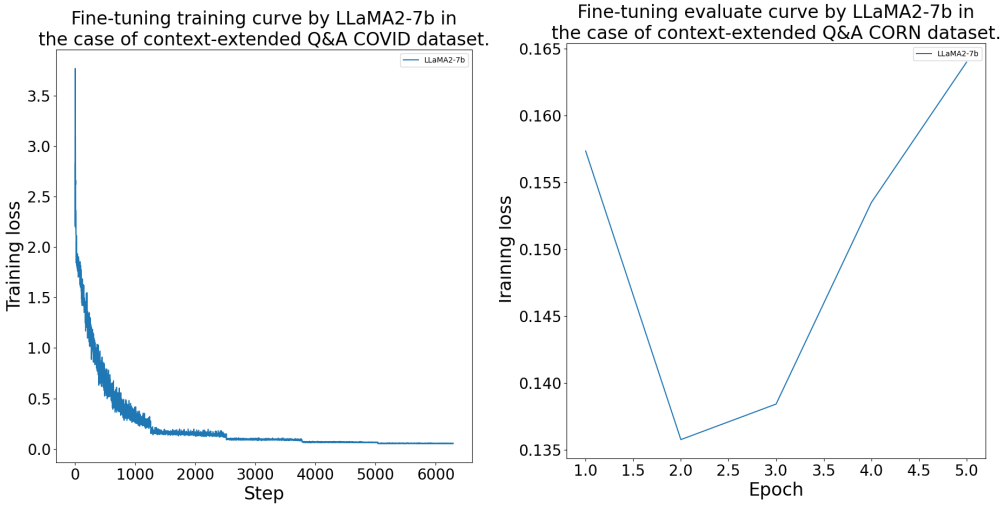


Figure A4. Domain-specific fine-tuning training and evaluation curves by model for the CORN dataset.

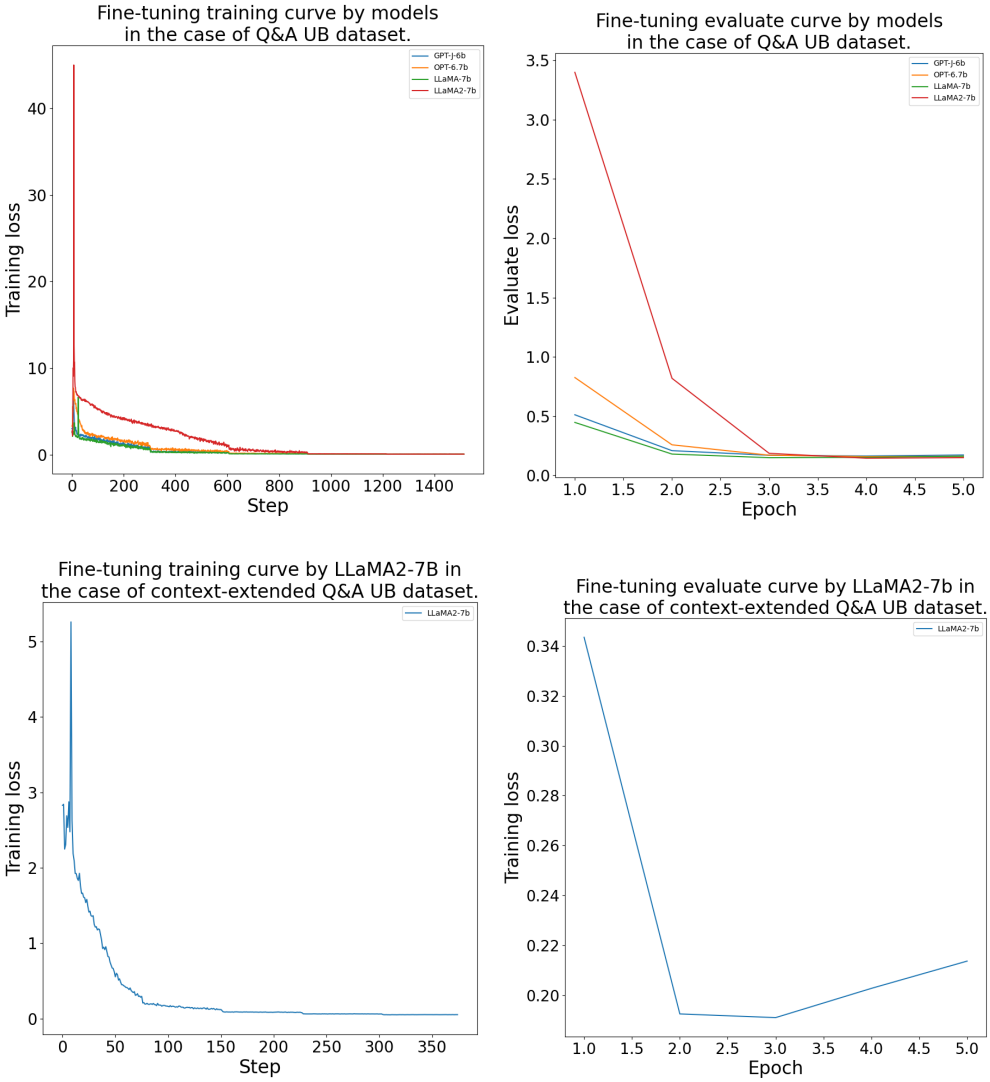
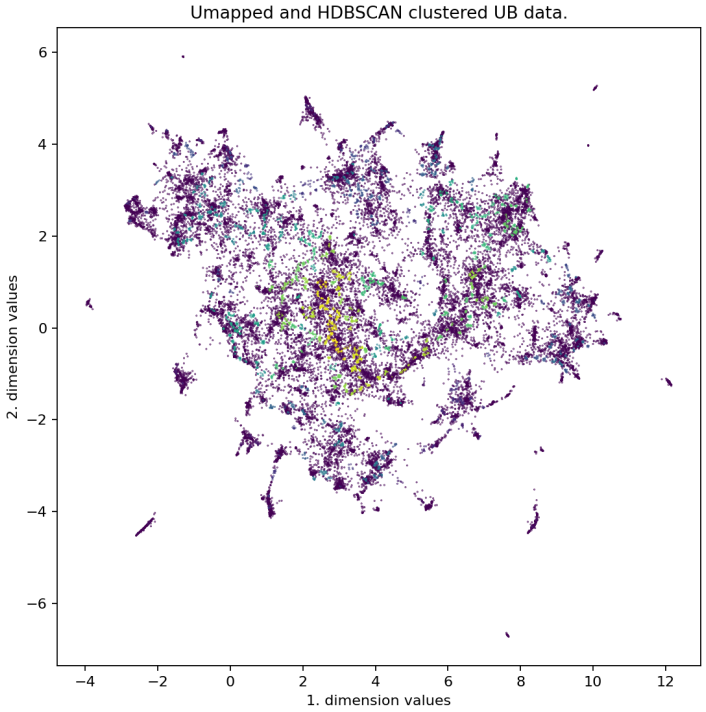
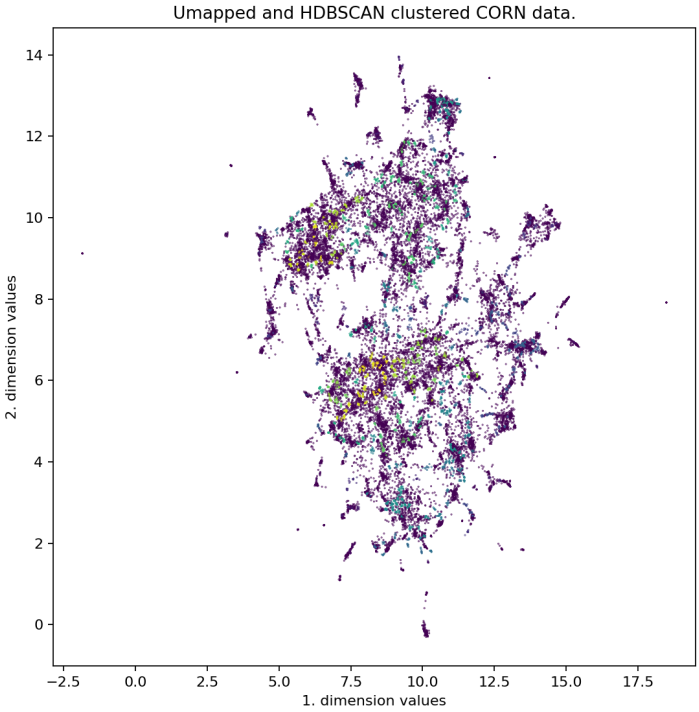


Figure A5. Domain-specific fine-tuning training and evaluation curves by model for the UB dataset.

Appendix C. Additional UMAP Representations



(**a**) UB



(**b**) CORN

Figure A6. Two-dimensional UMAP representation of the UB and CORN datasets.

Appendix D. Flow Diagram

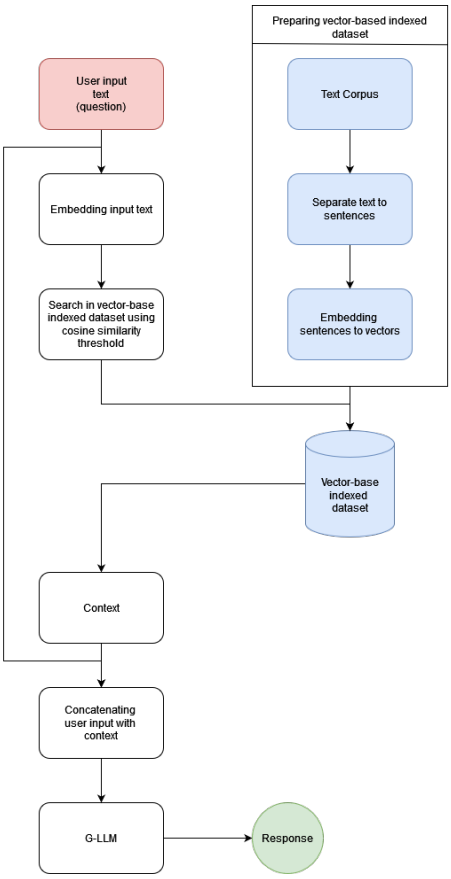


Figure A7. Domain-specific flow diagram of the RAG model (best approach) that uses a search engine based on the vectorial embedding of sentences.

References

1. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
2. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving Language Understanding by Generative Pre-Training. 2018. Available online: https://hayate-lab.com/wp-content/uploads/2023/05/43372bfa750340059ad87ac8e538c53b.pdf (accessed on 12 March 2024).
3. Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J.; Le, Q.V.; Salakhutdinov, R. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv* **2019**, arXiv:1901.02860.
4. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. *OpenAI blog* **2019**, *1*, 9.
5. OpenAI. ChatGPT. 2024. Available online: https://openai.com/index/chatgpt/ (accessed on 12 March 2024).
6. Microsoft. Microsoft Copilot. 2024. Available online: https://copilot.microsoft.com (accessed on 12 March 2024).
7. Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H.W.; Sutton, C.; Gehrmann, S.; et al. PaLM: Scaling Language Modeling with Pathways. *arXiv* **2022**, arXiv:2204.02311.
8. AI, G. Gemini. 2024. Available online: https://gemini.google.com/app (accessed on 12 March 2024).
9. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. LLaMA: Open and Efficient Foundation Language Models. *arXiv* **2023**, arXiv:2302.13971.
10. Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; Steinhardt, J. Measuring massive multitask language understanding. *arXiv* **2020**, arXiv:2009.03300.
11. Romera-Paredes, B.; Barekatain, M.; Novikov, A.; Balog, M.; Kumar, M.P.; Dupont, E.; Ruiz, F.J.; Ellenberg, J.S.; Wang, P.; Fawzi, O.; et al. Mathematical discoveries from program search with large language models. *Nature* **2023**, *625*, 1–3. [CrossRef] [PubMed]
12. Petroni, F.; Rocktäschel, T.; Lewis, P.; Bakhtin, A.; Wu, Y.; Miller, A.H.; Riedel, S. Language models as knowledge bases? *arXiv* **2019**, arXiv:1909.01066.

13. Raiaan, M.A.K.; Mukta, M.S.H.; Fatema, K.; Fahad, N.M.; Sakib, S.; Mim, M.M.J.; Ahmad, J.; Ali, M.E.; Azam, S. A review on large Language Models: Architectures, applications, taxonomies, open issues and challenges. *IEEE Access* **2024**, *12*, 26839–26874. [CrossRef]
14. Zheng, J.; Hong, H.; Wang, X.; Su, J.; Liang, Y.; Wu, S. Fine-tuning Large Language Models for Domain-specific Machine Translation. *arXiv* **2024**, arXiv:2402.15061.
15. Jeong, C. Fine-tuning and utilization methods of domain-specific llms. *arXiv* **2024**, arXiv:2401.02981.
16. Guu, K.; Lee, K.; Tung, Z.; Pasupat, P.; Chang, M. Retrieval augmented language model pre-training. In Proceedings of the International Conference on Machine Learning, Virtual Event, 13–18 July 2020; pp. 3929–3938.
17. Wang, L.L.; Lo, K.; Chandrasekhar, Y.; Reas, R.; Yang, J.; Burdick, D.; Eide, D.; Funk, K.; Katsis, Y.; Kinney, R.; et al. CORD-19: The Covid-19 open research dataset. *arXiv* **2020**, arXiv:2004.10706v4.
18. Ben Abacha, A.; Demner-Fushman, D. MedQuAD: A Medical Question Answering Dataset Containing Diverse Healthcare Topics. In *Studies in Health Technology and Informatics*; 2019; pp. 307–311. Available online: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-3119-4 (accessed on 12 March 2024).
19. Ben Abacha, A.; Demner-Fushman, D. A Question-Entailment Approach to Question Answering. *BMC Bioinform.* **2019**, *20*, 511:1–511:23. [CrossRef] [PubMed]
20. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *arXiv* **2020**, arXiv:1910.03771.
21. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*; Curran Associates, Inc.: Vancouver, BC, Canada 2019; pp. 8024–8035.
22. Wang, B.; Komatsuzaki, A. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. 2021. Available online: https://github.com/kingoflolz/mesh-transformer-jax (accessed on 12 March 2024).
23. Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X.V.; et al. Opt: Open pre-trained transformer language models. *arXiv* **2022**, arXiv:2205.01068.
24. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. Bleu: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 6–12 July 2002; Isabelle, P., Charniak, E., Lin, D., Eds.; Association for Computational Linguistics: Philadelphia, PA, USA, 2002; pp. 311–318. [CrossRef]
25. Lin, C.Y. ROUGE: A Package for Automatic Evaluation of Summaries. In Proceedings of the Text Summarization Branches Out, Barcelona, Spain, 25–26 July 2004; pp. 74–81.
26. Banerjee, S.; Lavie, A. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Ann Arbor, MI, USA, 29 June 2005; pp. 65–72.
27. Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; Zhou, M. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. *arXiv* **2020**, arXiv:2002.10957.
28. Joshi, B.; Shah, N.; Barbieri, F.; Neves, L. The Devil is in the Details: Evaluating Limitations of Transformer-based Methods for Granular Tasks. In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain, 8–13 December 2020; Scott, D., Bel, N., Zong, C., Eds.; International Committee on Computational Linguistics: Barcelona, Spain, 2020; pp. 3652–3659. [CrossRef]
29. Taori, R.; Gulrajani, I.; Zhang, T.; Dubois, Y.; Li, X.; Guestrin, C.; Liang, P.; Hashimoto, T.B. Stanford Alpaca: An Instruction-Following Llama Model. 2023. Available online: https://github.com/tatsu-lab/stanford_alpaca (accessed on 12 March 2024).
30. Voidful. Context Only Question Generator. 2024. Available online: https://huggingface.co/voidful/context-only-question-generator (accessed on 1 January 2024).
31. Reimers, N.; Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv* **2019**, arXiv:1908.10084.
32. Lawrence, N.D. A unifying probabilistic perspective for spectral dimensionality reduction: Insights and new models. *J. Mach. Learn. Res.* **2012**, *13*, 1609–1638.
33. Campello, R.J.; Moulavi, D.; Zimek, A.; Sander, J. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Trans. Knowl. Discov. Data (TKDD)* **2015**, *10*, 1–51. [CrossRef]
34. Lakatos, R. Retrieval-Augmented Generation Versus Domain-Specific Fine-Tuning. 2024. Available online: https://github.com/robertlakatos/ragvsfn/tree/main (accessed on 1 January 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.