LoRAG : CS 7643

Daniel Nicolas Gisolfi
Georgia Institute of Technology
dgisolfi3@gatech.edu

Kaushal Gandikota
Georgia Institute of Technology
kgandikota6@gatech.edu

Abstract

As reported by a study assessing medical satisfaction in the US by the AAPA (American Academy of Physician Associates), 66% of adults have reported their experience with healthcare as "more rushed than in the past". Several companies have been started to optimize parts of the American Medical Machine and we observed an opportunity to make training medical language models more feasible on the lower end hardware used in medical practices. LoRAG connects a pretrained language model with a LoRA head for parameter efficient tuning and a RAG model to inject context specific to a desired medical field into user prompting.

1. Introduction/Background/Motivation

Advances in medical technology enabled by machine learning in the past decade have focused primarily on drug discovery, drug testing, and imaging. These innovations have largely fallen in the research side of medicine but there has been a push to integrate these innovations into patient care. A popular approach that has been adopted by companies like Ambient and Epic is to leverage the wealth of information that is collected during patient visits to augment the speed of diagnosis and treatment. We recognized the magnitude of this opportunity and sought to improve the accuracy and generative ability of existing LLMs in the context of dermatology diagnoses.

The standard of care in medicine is still largely a hybrid approach consisting of medical practitioners combing through patient history and their experience in the field to diagnose and treat. Given advances in NLP as highlighted by the popularity of LLMs such as ChatGPT and OpenEvidence, this approach is modified in select practices based on the technical literacy of the practitioners. The existing LLMs can perform well on general diagnoses but require the input of the practitioner to be descriptive with information they feel is relevant to the case. This is where we hope to leverage the context that is available to us via patient information. By integrating the patient base into the context of each query, we can remove dependence on the variable window of memory that a Physician and their team has. Connections between similar patient cases can be made with reliable cadence.

In the "Emperor of All Maladies", Dr. Siddhartha Mukherjee delves into how discoveries of cancer biology failed to permeate throughout the community. A researcher or doctor may stumble on an important detail but in the current system this information is not universally stored immediately. An LLM molded to medical data can monitor patient reports and generate connections at an unprecedented speed. The role of the physician can then shift more towards that of care provider. Due to guidelines and testing, computers in healthcare infrastructure tend to be older in architecture. The ideal solution for this problem would leverage LLMs that have been pretrained on robust architecture while being conscious of memory footprint. We propose LoRAG, a training pipeline that consists of a pretrained text-to-text model that is finetuned using Low-Rank Adaptation (LoRA) and further bolstered with Retrieval-Augmented Generation (RAG) as the candidate for this unique use case.

The data that we used for our experiments was the "Mreeb/Dermatology-Question-Answer-Dataset-For-Fine-Tuning" dataset from huggingface. In addition to covering a wide array of skin conditions (cancers, lesions, etc.) in 1500 rows of data, it also provided information on the various treatment modalities used by clinicians to treat these conditions. This was important for us since we were aiming to create a pipeline that could demonstrate clinical benefits. For tuning the RAG model we used a more targeted dermatology dataset, "kingabzpro/dermatology-qa-firecrawl-dataset", oriented around specific questions that can identify a condition.

2. Approach

Our goal for this analysis is to improve language model performance on our selected specialized domain of dermatology by combing a pretrained text-to-text model with Low-Rank Adaptation (LoRA) based fine-tuning and Retrieval-Augmented Generation (RAG) to provide context

aware answers. Using both LoRA and RAG we aim to improve the deployment and storage efficiency of a pre-trained T5 model while preserving the base models domain adjacent knowledge.

2.1. Pipeline

2.1.1 Seq2Seq Language Model

Using a T5 based model, the Sequence to Sequence architecture handles our question and answer generative task as a text-to-text problem. The question from the dataset is passed into the encoder of the model, to do so we structure the text using a prompt format.

```
"### Question:{question}\n### Answer:"
```

This format provides the context of the Q/A query to the transformers attention mechanism through the context vector. T5's encoder/decoder structure with multi-head attention, can naturally link important parts of the question to the representations it builds during encoding. The decoder then produces the answer one token at a time using cross-attention to check encoded question features. This interaction allows the model to selectively attend to key semantic features, and integrate information from both the prompt and the learned task distribution. [2]

Our final decision to utilize T5 rather than alternative general language models like Llama or GPT was driven by our resource constrained environments. We began by thoroughly evaluating numerous biomedical pre-trained models available on HuggingFace to find the most effective starting point for fine-tuning. We evaluated many models like medicalT5 and others pre-trained on varying sub domains of the medical field. We found BioT5 to be the best option as it had been trained on chemical interactions through many modalities including question and answer format, which matched our task well [6].

2.1.2 LoRA

LoRA, or Low-Rank Adaptation is a method of fine-tuning large language models by tuning the changes in the most essential parameters. This is done primarily by harnessing rank to reduce the dimensionality of the matrices that we calculate loss updates on. LoRA offers a high level of customization since it is applied to the multiple weight matrices present in LLM architecture.[3]

Using the Parameter-Efficient Fine-Tuning (PEFT) libray we utilized the LoRA adapter to target specific modules within the T5 transformer architecture. We targeted the Query (Q), Key(K), Value(V), and the Attention Output (O) for the self attention modules of the encoder, decoder, and cross-attention layers. By focusing on unfreezing the parameters of the attention mechanism, we aim to allow the model to learn a new, medical text-to-text specific attention strategy for the dermatology sub-domain. This module targeting strategy allows us to target less than ten percent of T5's 250+ million parameters ensuring only the essential weights for our generation task are tuned.

We hypothesize that the small base T5 model pre-trained on clinical or biomedical data can perform well on our dermatology Q/A dataset as the base model is structured for this task. However, unless our choice of pre-tuned T5 variant includes dermatology specific domain knowledge the model may struggle to learn and capture the specific medical terms within the data. By adding LoRA adapters to the linear attention layers of the model, we aim to adjust the attention specific parameters while leaving the remaining weights frozen. We expect the fine-tuning of the self-attention weights to help the encoder focus more effectively on the specific unseen medical terminology, relating to the datasets which has not yet been seen by the model. Additionally we expect the same fine-tuned self attention layers to help the decoder generate answers that better reflect medical context.

2.1.3 RAG

Retrieval augmented generation, or RAG, is a way to add a context window to each query that an LLM responds to. To inform the high level architecture of the RAG pipeline we referred to a tutorial from the HuggingFace OpenSource Cookbook. [7] They are used in cases when there is an additional level of information unique to a task that the generalized model is not trained on. In the case of LoRAG, this is an additional dermatology dataset that contains questions specific to the diagnosis of pathological skin conditions. The knowledge database can be thought of as a filing cabinet and each folder inside contains information from the additional dermatology dataset. We utilize a splitter mechanism to chunk sections of the knowledge store into tokens. In the context of the filing cabinet analogy, each chunk of tokens is placed in a separate file in the cabinet. There are two main structures inside the RAG architecture that connect this knowledge database to the final answer: the retriever and the reader. The retriever is responsible for selecting the relevant documents to the user's query in the knowledge database. The retriever searches through the vector database and determines the documents that are most relevant to the user query. It accomplishes this with an embedding model that embeds both the user query and the information in the documents of the vector database. Our architecture uses the "thenlper/gte-small" model for this purpose. From here, we use the cosine distance metric to determine the k-nearest neighbors to the user query within the document store. We tuned the model to use the top six documents, or six nearest neighbors, relevant to the user query. The reader is

where we connect the RAG model to the rest of the LoRAG pipeline. Once LoRA has been used to tune the underlying bioT5 model, we can use it read the output from the retriever. The reader is also responsible for matching the nearest neighbor documents to the user query with their respective text contents and then calling the tuned model on a prompt which places the retrieved document, context, and query into a text block. The manipulation of ordering question and answer that we used to solve the RAG output issues we outline in our experiments section was handled in this exact part of the reader block.

3. Experiments and Results

3.1. Metrics

To measure the validity of generated responses from the model we explored a range of Natural Language Processing (NLP) metrics. During initial training and evaluation, common N-gram metrics were used for evaluating responses. However after initially fine-tuning the model we found that despite decent ROUGE and BLEU scores, the generated text upon manual inspection failed to create a coherent sentence as a response. As a result we explored metrics which provide measures of sematic similarity, and chose the following three. Finally, we also explored how our architecture affects the memory footprint of the model with the introduction of LoRA.

3.1.1 BERT F1 Score

Similar to our initial lexical based metrics, BERTScore also operates at the token level. By comparing the unigrams of the predicted text to those of the reference text we can measure the similarity between them. However, instead of comparing tokens directly, BERTScore uses a pretrained BERT encoder to obtain contextualized embeddings for each token. These embeddings allow the metric to capture semantic similarity in latent space rather than exact token matching. Using the embeddings, BERTScore computes the pairwise cosine similarity between each token in the prediction and reference. Averaging the maximum similarity between the predicted token and any of the reference tokens, the precision is calculated for the entire prediction sequence. Inversely recall is computed by the average maximum similarity for each reference token relative to the predicted sequence. The final BERTScore F1 value is then produced by combining these precision and recall scores through their harmonic mean [8].

The resulting BERT F1 Score provides a measure of token-level semantic similarity between predicted and reference medical responses. We use this metric to determine basic alignment of the tokens within the given context of the question.

3.1.2 Semantic Score

Upon further inspection of our Question/Answer dataset, we found that both the questions and responses frequently consisted of multiple sentences, resembling short paragraphs rather than single-sentence prompts. Because token-level metrics may struggle to capture the coherence and meaning across longer spans of text, we used sentence transformers to provide a vector embedding to perform a similar semantic comparison to Bert F1 Score. This vector is used for sentence-level representations to measure cohesive responses through longer reference text sequences. [1]

Following huggingface tutorials and documentation we used their guidance for constructing a SemScore. We employed a pretrained sentence-transformer model, mxbai-embed-large-v1, to generate sentence-level embeddings instead of token-level representations. These embeddings provide a higher-level semantic summary of each sentence, making them better suited for evaluating the multi-sentence medical responses. By representing each sentence as a vector, we can directly apply cosine similarity to measure the semantic relatedness between the generated answer and its reference. This method bypasses specific token-level matches and instead assesses whether the overall meaning, intent, and informational content of the response align with the ground truth.

3.1.3 NLI Entailment

While the previous two metrics guide our fine-tuning towards well structured sentences which can pass as a comprehensible response, examining the generated samples from our best tuned models showed a clear inability to reproduce logically consistent answers compared to the ground truth.

To measure the logical consistency and factuality of the generated text we explored Multi-Genere Natural Language Inference (MNLI) models. We found using a pre-trained model, in our case DeBERTa-MNLI, to perform a classification task to predict the following labels ”contradiction, neutral and entailment”. Using the logits of the classification model we apply softmax to get the probabilities that the hypothesis (generated text) is entailed by the ground truth premise. [5]

3.1.4 Memory Footprint

LoRA, as explained in our pipeline section, was designed to finetune the most essential portion of the parameters while freezing the rest to reduce the model’s memory footprint. Mathematically this can be represented as the tuning of the decomposed matrices which when multiplied are the size of any given weight matrix. By keeping track of decomposed, or ’factor’ matrices, LoRA enables us to calculate

parameter weight changes without storing the product, or total parameter weight change matrix. This key part of the architecture is what enables LoRA to minimize its memory allocation. To test and observe this in the bioT5 model, we implemented a custom metric that would determine the size in megabytes of the trainable parameters, frozen parameters, and activation memory in our different configurations. We based our high-level architecture off of a blog post but customized it to fit the needs of our model's data types and the LoRA head. [4]

Observing the differences in the frozen parameter memory and activation memory between configurations will allow us to 1) verify that LoRA is working as designed and 2) pinpoint where it creates the most performance gain. This will also be used to assess the inclusion of RAG in this pipeline, marking the key difference between our paper and existing research on the effects of LoRA.

This is especially important for our objective since the hypothesis we sought to test was that combining LoRA with BioT5 would reduce the burden on the outdated hardware that healthcare practices currently use.

3.2. Experiments

3.2.1 Training Objective

We performed evaluations on the base pipeline across multiple medical Q/A datasets and pre-trained T5 variations, this study was conducted across narrowing searches within the base parameters of the model. Throughout these experiments we utilized our three established quantitative sematic metrics to guide our search. We noted a common trend amoungst the evaluations, BERT F1 score seemed trivial for the model to reach once passing below an eval loss of 4. Often, this score improves the earliest of the measures, indicating that the model quickly learned to use semantically appropriate tokens. Similarly, the SemScore metric would also rise an epoch or two behind BertScore F1, signifying that the generated paragraphs were stringing context together better to capture the overall sentence level meaning. However trailing behind, the NLI Entailment score consistently plateaued at a low maximum evaluation score of only 0.2 to 0.3 for even our best performing models. As an alternative approach for exploring the search space for an ideal solution we updated the objective function of our search, modifying the score of a given epochs result to the following formula.

$$Score = EvalLoss - (0.25 * Entailment)$$

Initially, we experimented with a much higher weighting factor for entailment, closer to 0.5 to balance the importance of both low loss with logical consistency. However we quickly noticed that without strong lexical fidelity the model could not make any meaningful improvement on entailment as structured sentences are a prerequisite. The near equally weighted scoring factor resulted in an unstable exploration of the parameter space. Upon further inspection we observed a worsening of all semantic metrics as the direction of improvement for the evaluation loss was curtailed by our entailment metric. Confirming the samples manually, we found malformed words and other chunks of misplaced tokens in the samples with readable portions of text between. We noted that for the model to start improving in entailment it needs very well defined responses with decent semantic relationships before logical equivalence is preserved. To counteract this negative effect we explored a range of weights for the entailment score, settling on the 0.25. This multi objective training is used throughout our experiments.

3.2.2 BioT5 Base fine-tuning

Our first experiment evaluated the ability of the base BioT5 model to adapt to and learn the new, closely related medical text-to-text task. We began by fine-tuning the model and manually examining a sample of its generated responses to assess overall coherence and alignment with the expected medical context. However, our early experiments with BioT5 highlighted a fundamental challenge in domain adaptation. For example, during initial setup when testing, feeding the dermatology Q/A data into the transformer resulted in sequences resembling Protein-text Pairs.

```
"bop><p>M<p>K<p>R...<p>N<p>E...<eop"
```

We observed that the model defaulted to generating known tokens from a specific modality, protein-text pairs, which the model was exposed to during its biomedical training. With only minimal exposure to the new target domain, the model failed to generalize and instead relied heavily on its original pre-trained representations. As a result, it struggled to recall or produce natural text responses aligned with the Q/A modality.

To address this, we performed several grid searches to fine-tune the model and encourage it to learn our medical text and question response structured patterns rather than reverting to its pre-trained modalities. This process yielded a model that began to capture some of the necessary token sequences needed for a cohesive response, although the improvements were limited. The fine-tuned model showed early signs of adapting but still displayed strong residual bias toward its original training domain as seen in the following example.

```
"protect skin from UV rays and UV rays.."
```

When manually evaluating the quality of sampled outputs we continued to observe Protein-text pairs interwoven between samples like the above partially legible responses. In cases like the sample, the model exhibited mode collapse

Table 1. Evaluation metrics for trained models.

Model	Trainable Params	Loss	BERTScore F1	Mean Sentence Similarity	Mean Entailment Prob.
BioT5 Base	255m	0.34	0.79	0.62	0.34
BioT5 LoRA (Self-Attn)	3.5m	2.9	0.86	0.85	0.15
BioT5 LoRA (All Linear)	6.7m	3.6	0.88	0.89	0.24

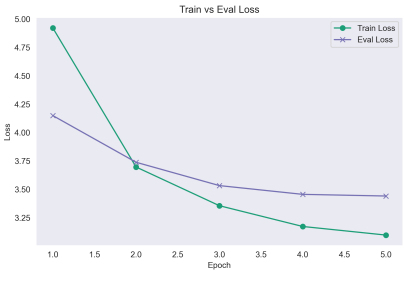

Figure 1. Base BioT5 best Model Loss Curve

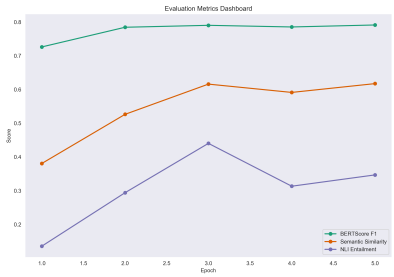

Figure 2. Base BioT5 best Model Metrics

in which it repeatedly generated the same tokens. This behavior shows that while our best result did begin to adapt the bioT5 model for the task, this limited exposure to the new specific prompt and response structure was only partially learned. Our analysis of the text samples confirmed that despite the well defined search our best parameters for fine-tuning the model to our task fell short. To confirm this, we plotted the loss curve seen in figure 1. Although the training and evaluation curve appears stable and convergent, it does not reflect the model's inability to escape its pre-trained modality. The real issue only appears in the semantic evaluation metrics shown in Figure 2. When the model fell into mode collapse and began repeating tokens at some point in many of the responses, the resulting text lost coherence and logical structure. Without learning the structure it is impossible to maintain meaningful sentence-level similarity or entailment, leading to both metrics at low values , plateauing very early. The partial sentences seen in the samples are well described by the much higher Bert Score F1 as at least on the individual token level the model could be improving, but stayed below eighty percent showing that the generated text did not align well with the input.

3.2.3 LoRA Target Modules

We repeated the same experiment using LoRA to adapt a targeted subset of the trainable parameters of the base T5 model. We again performed a comparable grid search, tuning the LoRA specific hyperparameters along with the original parameters used for T5 to find an ideal combination. For the search we again used our entailment weighted evaluation loss as the objective to minimize. Unlike the fully fine-tuned BioT5 baseline, when adapting the model with LoRA we targeted the self-attention modules (Q, K, V, O), allowing our fine-tuning to focus on learning the token and response structure patterns needed for the task. This precise adaptation avoided the mode collapse observed in the best of the baseline model, instead allowing the resulting responses to be coherent and utilize the newly learned dermatology medical terms needed to properly answer the prompts. This result is confirmed using the sematic similarity metrics, seen in table 1, where both token level and sentence level semantics have improved greatly from the baseline reaching about 0.8. By using LoRA to concentrate on the self attention weights we were able to to reduce the targeted parameters to 3.5 million from the more than a quarter billion updated during the base BioT5 training. The reduction in training scope has proven effective at focusing the model on training its self attention weights and relying on its remaining pre-trained knowledge to solve the task effectively.

However, although the self-attention-focused LoRA configuration resolved sample-quality degradation, the mean entailment score did not improve to the same extent. To further enhance LoRA's effectiveness, we examined the architecture and implementation details of the base T5 model and identified additional modules that could be adapted. This led us to extend LoRA beyond the attention mechanisms to include key components of the Feed-Forward Network (FFN), targeting the linear layers $W_{i,0}$, $W_{i,1}$ and W_o [2]. Adding these three additional modules increased the trainable parameters to nearly 7 million, providing additional capacity for tuning the model for greater entailment probability while minimizing loss.

However, as per table 1 , expanding LoRA to target all

linear layers led to worse evaluation loss and only marginal changes in semantic similarity. While our training objective was partially successful the mean entailment probability has not yet returned to that of the base BioT5 model without LoRA adaption. Incorporating these additional layers increased LoRA's expressive capacity while preserving the efficiency benefits of parameter-efficient fine-tuning.

3.2.4 LoRA Memory Footprint

As mentioned in the Memory Footprint section of our Metrics, LoRA's advantage in model size is that for each parameter weight change that is performed on weight matrices, there is no storage of the composite change matrix. The values of the change matrix are calculated in real time and losses are determined with each forward pass since the computational burden is less when performing operations in the lower rank factor matrices.

We recorded memory values for each configuration we run. Our results on the baseline and LoRA model memory testing are shown below 2 . The activation memory decreased with the inclusion of LoRA. The activation memory encompasses all intermediary memory used in the forward pass. Although the amount of activation memory decreased, it was not a significant amount. We believe this is because bioT5 is already very light. The important note is that we did observe a directional change that aligned with our hypothesis and it is likely to create a much more dramatic change when using larger models that can produce more robust and accurate outputs for medical data generation.

3.2.5 Tuning RAG

During our tuning of RAG we tested various combinations of chunk sizes for the splitter and maximum length of parameters in the model configuration and observed some interesting behavior in how the additional dermatology dataset for RAG was processed into output . When max length was set to 128 tokens for the model parameter configuration, the answer that was passed into the documents of our knowledge database was often truncated too early. The responses to each question in the additional dermatology dataset would exceed the token limit of 128, resulting in final RAG outputs that defaulted to the base bioT5 model chemical structure outputs. Once we increased the max length in the configuration to 512 tokens, the maximum for bioT5, we were able to accommodate the question, answer, and a portion of the context as well. This enabled the prompt to contain the necessary information for the LLM to respond with a natural language response about surgical treatment options for melanoma. In the absence of a clear question and answer as limited by the 128 token limit, the bioT5 model resorted to returning SMILE molecular structures. This was most likely caused by the abundance of SMILE data in the pretraining for bioT5.

Table 2. Memory Metrics for models.

Metric	BioT5 Base	BioT5 LoRA
Trainable Param (mb)	961.7	27.5
Activation (mb)	1794.5	1737.7

3.3. Results

Overall our results partially confirmed our initial hypothesis, that a medical pre-trained T5 based model could perform well on the Dermatology Question/Answer task. While the base T5 architecure is well suited for the sequence-to-sequence medical text generation, its performance is heavily dependent on the multiple tasks and modalities a model was pre-trained with. In particular, while LoRA proved capable of adapting the model to the new domain, our experiments showed that the choice of modules to target from the architecture is crucial. Targeting only the self-attention layers produced more stable and coherent outputs, whereas extending LoRA to all linear layers introduced substantial variability in the semantic similarity and logical consistency of generated responses.

These findings show that even with domain adjacent pretrained biomedical models like BioT5, differences in text structure required to solve a new task like the natural language Question/Answer task can require significantly more fine-tuning. One way to address the additional training requirements we observed is to vary the modules targeted during adaptation, allowing the model to retain useful components of its base training while selectively re-learning the parts needed for the new task. Our experiments demonstrate that parameter-efficient methods like LoRA can successfully adapt a biomedical Seq2Seq model to a specialized subfield, but only when the adaptation targets are carefully selected to balance expressiveness and stability

The inclusion of RAG into our testing allowed for more precise answers in our analysis but this benefit was overshadowed by the bioT5 model resorting to default settings whenever the question was not aligned with the FAISS indexed vectorized database. We tested different chunking sizes and max lengths to create the right kind of prompt for the reader to provide to the bioT5 but we found a high level of sensitivity that led to inadequate results. Namely, moving the max length of parameters from 256 to 512 should have theoretically enabled the model to be comfortable with more difficult questions, but this also had the negative effect of pushing our document sizes past the token limits visible to the model. A "spillover" effect was noted where whichever portion of the prompt was at the bottom in reading order was likely to be truncated. The model was then fed an incomplete prompt which lacked the key words required to prompt it out of its default generations.

4. Source Code

https://github.com/dgisolfi/LoRAG

References

[1] Alan Akbik Ansar Aynetdinov. Semscore: Automated evaluation of instruction-tuned llms based on semantic textual similarity, 2024. 3

[2] Adam Roberts Katherine Lee Sharan Narang Michael Matena Yanqi Zhou Wei Li Peter J. Liu Colin Raffel, Noam Shazeer. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020. 2, 5

[3] Phillip Wallis Zeyuan Allen-Zhu Yuanzhi Li Shean Wang Lu Wang Weizhu Chen Edward J. Hu, Yelong Shen. Lora: Low-rank adaptation of large language models, 2021. 2

[4] Mark. How to get model information: Size, parameters, and architecture details, 2025. no listed author, just referred to as 'Mark'. 4

[5] Jianfeng Gao Weizhu Chen Pengcheng He, Xiaodong Liu. Deberta: Decoding-enhanced bert with disentangled attention, 2021. 3

[6] Jinhua Zhu Kehan Wu Kaiyuan Gao Lijun Wu Yingce Xia Rui Yan Qizhi Pei, Wei Zhang. Biot5: Enriching cross-modal integration in biology with chemical knowledge and natural language associations, 2023. 2

[7] Aymeric Roucher. Advanced rag on hugging face documentation using langchain, 2025. 2

[8] Felix Wu Kilian Q. Weinberger Yoav Artzi Tianyi Zhang, Varsha Kishore. Bertscore: Evaluating text generation with bert, 2019. 3

Student Name	Contributed Aspects	Details
Daniel Nicolas Gisolfi	ML Pipeline	Dataset Load/Preprocess, Seq2Seq LM, LoRA, Model Metrics, and Optuna (Grid-Search)
Kaushal Gandikota	RAG implementation and overall ML Pipeline evaluation	Dataset Load/Preprocess for VectorDB, FAISS indexing, Retrieval, LoRA Context Injection Pipeline, Memory Metrics

Table 3. Contributions of team members.