CAI 4104/6108: Machine Learning Engineering

Project Report: Contrasting RAG with Fine-tuning Methods for Medical Text Summarization

Hruday Tej Akkaladevi
(Point of Contact)
hrudayte.akkalad@ufl.edu

Jordan Sabo
jordan.sabo@ufl.edu

Mai Tran
tran.mai@ufl.edu

Samuel Battis
s.battis@ufl.edu

April 26, 2024

1 Introduction

Electronic health records (EHRs) rely heavily on comprehensive doctor notes from patient consultations to ensure accurate and reliable data. Summarizing these notes automatically presents a compelling proposition due to the potential for significant time savings for clinicians. Furthermore, recent studies suggest that Large Language Models (LLMs) might outperform human-generated summaries in retaining critical information within the condensed text [15]. This improved information retention could lead to more informed clinical decision-making and potentially better patient care.

This paper presents a comparative analysis of two prominent approaches for training a LLM - Llama-2 for medical text summarization: Retrieval-Augmented Generation (RAG) and Fine-tuning. We aim to evaluate and contrast the effectiveness, robustness, and suitability of these methods for this specific domain. By analyzing the strengths and weaknesses of each approach, we hope to gain valuable insights that can inform the development of future medical text summarization systems.

By conducting a comprehensive experimental study and analysis, we aim to contribute valuable insights to the development and deployment of text summarization systems within healthcare settings. Ultimately, our goal is to advance the state-of-the-art in medical text summarization and facilitate the adoption of effective and efficient summarization techniques. This, in turn, could free up valuable clinician time, support more informed clinical decision-making, and ultimately improve patient care delivery.

2 Approach: Dataset(s) & Pipeline(s)

In this section we would go through complete methodology followed in executing the project. We describe the dataset used, the models and their concepts including how they have been trained with respective architecture diagrams. Essentially we detail on how we are solving the problem of Doctor-Patient interaction summaries for a fast and quick review of the patient diagnosis history utilizing models like lstm, bert and llama.

2.1 Dataset

We used a dataset comprised of information from three publicly available datasets. One concerns radiology reports, one concerns patient/consumer health questions, and one concerns patient-doctor dialogue. In each of these datasets, there is an input text and a target text that represents the conclusions to be drawn from the input text. This data is available at https://github.com/StanfordMIMI/clin-summ/tree/main/data.

2.2 LSTM

For the LSTM baseline, the data was first preprocessed to add start and end tokens to the summaries, then padded to a fixed length, tokenized and vectorized. The actual model used an encoder-decoder architecture, which is depicted in Figure 1. The encoder used three LSTM layers to encode the input data into two state vectors. The

decoder then took those state vectors as input, as well as the target sequence moved one timestep forward to train it using teacher forcing. The output layer was a TimeDistributed(Dense) layer with softmax as its activation function, and the model was trained with sparse categorical cross-entropy as the loss function and RMSprop as the optimizer.

At inference time, the input sequence is tokenized and vectorized in the same manner as the training data, then encoded into state vectors with the encoder. The state vectors and start token are then passed into the decoder to produce a prediction, from which the next token is sampled and appended to the output sequence, and the process repeats until the end token or maximum length is reached.

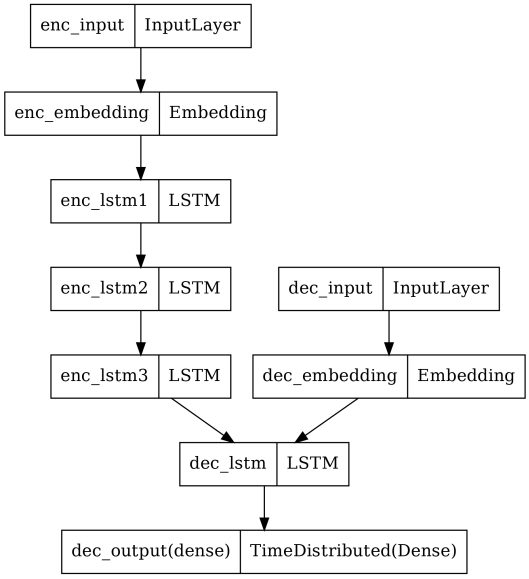

Figure 1: LSTM Architecture

2.3 BERT Fine-tuning

This pipeline takes inspiration from a Medium.com tutorial [2]. It begins by loading data from the three JSONL files containing training data from the dataset. It preprocesses the data and creates a bag of sentences from all inputs. Next, it pairs sentences for training, either picking adjacent sentences from paragraphs or pairing sentences with a random sentence from the bag. These pairs are labeled based on whether they are adjacent or not.

The preprocessed data is then tokenized using a tokenizer, taken from the pretrained 'bert-based-uncased' model [3], and a Masked Language Modeling (MLM) strategy is applied where random tokens in the input are masked, except for special tokens. This prepares the data for training in a BERT model. To organize this tokenized data, a custom dataset class (BERTDataset) is created, and a data loader is set up to load batches of this data during training. A BERT model is also initialized from the pretrained 'bert-base-uncased' model.

The model training phase involves setting up an AdamW optimizer (with a learning rate of 0.00001) and defining the number of training epochs. The training loop runs for the 10 epochs, iterating through batches of tokenized data. In each iteration, it computes the loss, performs back-propagation, and updates the model parameters.

2.4 Llama

Our research utilizes Llama-2, a family of large language models (LLMs) developed by Meta AI in 2023. These models are particularly adept at understanding and processing complex language, making them well-suited for the task of medical text summarization. We will be leveraging both Retrieval-Augmented Generation (RAG) and Fine-tuning techniques to train Llama-2 for this specific domain, allowing us to compare their effectiveness in capturing crucial medical information while maintaining clear and concise summaries.

2.4.1 PEFT using Lora

As the complexity of Large Language Models keeps increasing every day, with billions of parameters being added in every generation of an LLM, it becomes very challenging to fine-tune the LLMs due to computational and memory overheads [9]. To tackle this issue, a Parameter Efficient Fine-Tuning method called Low-Rank Adaptation (LoRA) [8] has been introduced to freeze certain layers and train only a few layers custom to our dataset in an efficient and cost-effective way. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times d}$, its update is constrained by representing it with a low-rank decomposition $W_0 + \Delta W = W_0 + BA$, where $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times d}$, and the rank $r \ll d$ [8]. During training, W_0 remains frozen and does not receive gradient updates, while A and B contain trainable parameters. Both W_0 and $\Delta W = BA$ are multiplied with the same input x, and their respective output vectors are summed coordinate-wise. The modified forward pass can be expressed as:

$$h = W_0 x + \Delta W x = W_0 x + BAx \tag{1}$$

A is initialized with random Gaussian values, and B is initialized with zeros, ensuring that $\Delta W = BA$ is zero at the beginning of training [8]. $\Delta W x$ is scaled by α_r, where α is a constant, and r represents the rank. The choice of "r" is crucial for the LoRA algorithm to work because tuning it very low would result in the loss of crucial features as we would implicitly remove linearly dependent features, and by choosing a large "r", we increase the dimension and the number of linearly dependent features [1].

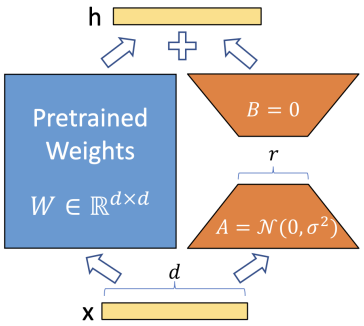

Figure 2: Illustration of the Low-Rank Adaptation (LoRA) technique [8].

2.4.2 Retrieval Augmented Generation

The Retrieval-Augmented Generation (RAG) framework, pioneered by Lewis et al. (2020) [10], aims to enhance the capabilities of conversational AI models by incorporating domain-specific knowledge during the response generation process.

In the RAG framework, there are three key components: a retriever, an adder, and a generator. The retriever's role is to retrieve relevant context-specific knowledge given a query. Its architecture involves computing a probability distribution over passages of knowledge, denoted as $p_\eta(z|x)$, where x represents the query. This distribution is proportional to the exponential of a scoring function $d(z)^{q(x)}$, where $d(z)$ represents the embedding of a passage using BERT and $q(x)$ represents the embedding of the query using BERT. These two elements are concatenated by the adder and used as the basis for BART generation.

By integrating retrieval-based methods with generation-based methods, RAG facilitates the generation of responses that are not only contextually relevant but also enriched with domain-specific knowledge, leading to more accurate and informative interactions in conversational AI systems.

2.4.3 Llama with PEFT

The section 2.4.1 explains how we have utilized the benefits of Low Rank Adaption and fine tune the Large Language Model custom to our dataset and problem. In this subsection we will present how we have implemented the LoRA technique in finetuning the Llama-2 model. We have performed the dataset preprocessing (which would be explained in the following sections) in a format suitable for efficient training for Llama-2. We have also utilized

the advantages the Quantization[6] provides in efficient and fast training of Large Language Models, and loaded the pretrained "Llama-2-7b-chat-hf" model. the model will load weights quantized to 4 bits and use bfloat16 for computation during inference, potentially improving efficiency and reducing memory usage while maintaining accuracy. Then for every training example, we will have a prompt format, which would instruct the LLM on what action to be taken, as the Llama-2 model is a general purpose model.

Then using the LoRa configuration with r=8, we initilize the peft client which then would be passed on to the training arguments where we define all the parameters like batch size, optimizer, lr scheduler type. And then utilizing the SFTTrainer [5] we initiate the model training and monitor the training loss.

2.4.4 Llama with RAG

In developing the Retrieval Augmented System for the Text Summarization task, we use the approach very similar to PEFT, where we train the neural network based on the prompt engineering. Using the SimpleInputPrompt library, using it we will append a prompt to every training set. This prompt will instruct and guide the Large Language Model to provide suitable response. Using Huggingface library we will import the "Llama-2-7b-chat-hf" model. As there is no finetuning involved here, we do not require LoRA or Quantization here. Then we prepare the necessary components for semantic search, including embedding models and service context settings, to facilitate efficient document indexing and retrieval using the Langchain framework and sentence-transformer model[13]. We put all the datasets (or a Knowledgebase in RAG terms) in one directory and using the SimpleDirectoryReader we load the data into service context. We then create a vectorstore index, which would be used for retrieval process of RAG, we use this index as a query engine and using the prompt we generate the response (summary) to the doctor-patient interaction.

3 Evaluation Methodology

3.1 Data Engineering

The dataset used was already split into training, validation and testing sets of appropriate proportions in each of the three categories, so we used those as provided. We aggregated all 3 corresponding subsets for the specific tasks; all three training sets were combined into one large training set, and the same applies to the validation and testing data. Although there was not much data cleaning/pre-processing was involved, the data transformation in a suitable properly formatted text file is done to build the knowledge base for the RAG model, this would help the Retriever to capture the require information, without any overlapping between 2 records in the dataset. As, it is a dataset for NLP model, there was not much enhancement needed, and this dataset being a medical records, filtering using profanity was not necessary.

3.2 Metrics

To evaluate the models, we used BERTScore, BLEU, and ROUGE as our metrics.

- **BERTScore** is an evaluation metric for text-generation that computes a similarity score between tokens in the generated text and tokens in the reference text using contextual embeddings produced by the BERT model [17].
- **BLEU** evaluates machine translation by calculating the modified n-gram precision of the generated text compared to the reference text. The formula is

$$\text{BLEU} = \text{BP} \cdot \exp(\sum_{n=1}^{N} w_n \log p_n)$$

  where N is the maximum length of n-gram used, w_n are positive weights summing to one, p_n is the modified n-gram precision, and BP is the brevity penalty, which penalizes translations that are shorter than the reference text [12].
- **ROUGE** is a set of evaluation metrics that measures the quality of a summary by counting the number of overlapping n-grams, word sub-sequences, or pairs of words between the reference and generated summaries [11]. There are multiple variations of ROUGE scores, and we decided to use ROUGE-1.

3.3 Baselines

We trained two baseline models to compare against the performance of the LLaMA models: a fine-tuned BERT model, and an LSTM model.

3.3.1 LSTM

The Long Short-Term Memory architecture was originally designed as a solution to backpropagation errors leading to exploding/vanishing gradients. It uses a memory cell along with gates to selectively retain or override information in the memory cell [7]. This lends itself to tasks in which long-term dependencies are needed to be remembered, and as such makes it a suitable architecture for a baseline model in our text summarization task. In particular, our LSTM baseline used three LSTM layers in the encoder, and one LSTM layer in the decoder.

3.3.2 BERT

BERT is a bidirectional transformer trained on next sentence prediction and masked language modeling, and can be fine-tuned with only one additional output layer[16][4]. Unlike directional models that process text input in a sequential manner, either left-to-right or right-to-left, the transformer encoder in BERT takes in the entire sequence of words simultaneously and uses self attention to capture the relationships between words.

The self attention mechanism works when processing a word by assigning attention scores to all the other words in the sentence based on how relevant they are to the current word. This gives BERT the ability to weigh the importance of each word's contribution to the overall meaning of the sentence. And this context information along with the hidden state is passed on to the decoder part of the model. This ability to determine which words are most important to the overall meaning makes BERT a good baseline for meaning related tasks such as summarization.

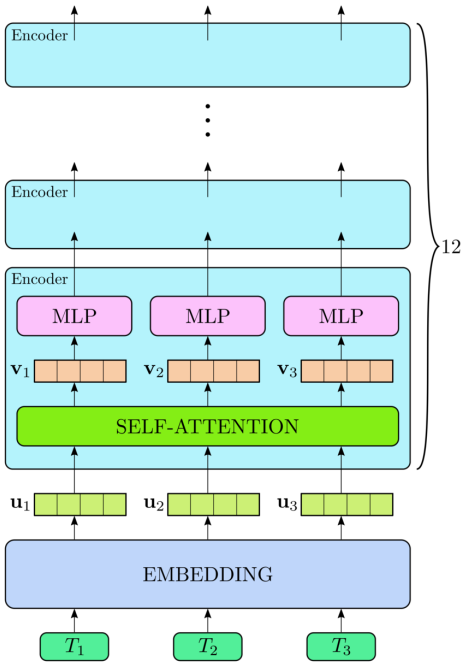

Figure 3: Overview of BERT Architecture [14].

4 Results

After performing the experimentation on the models over the dataset we have consolidated the results over the metrics BLEU, ROGUE and BERT Score are shown in the Table 1. And after examining the results the LLaMA's dominance is clearly evident over the baseline models. The main reason of the LSTM models failure can be due

to its in-competency to remember long-term dependencies as the architecture of LSTM is linear and sequential in nature.

On the other hand BERT Fine-Tuned model performs very well compared to the LSTM model. This is due to the introduction of the concept of attention, this attention is used to pass on the context information from the earlier layers along with the hidden state to the decoder model. With this self-attention mechanism the long term dependencies are well retained and the model performs well.

	BLEU	ROUGE	BERT		
			F1	Precision	Recall
LSTM	0.005	0.51	0.67	0.69	0.65
BERT Fine-Tuned (FT)	0.35	0.59	0.77	0.71	0.89
Llama Fine-Tuned	0.27	0.39	0.81	0.89	0.91
Llama RAG	0.39	0.48	0.88	0.84	0.92

Table 1: Metrics for different models

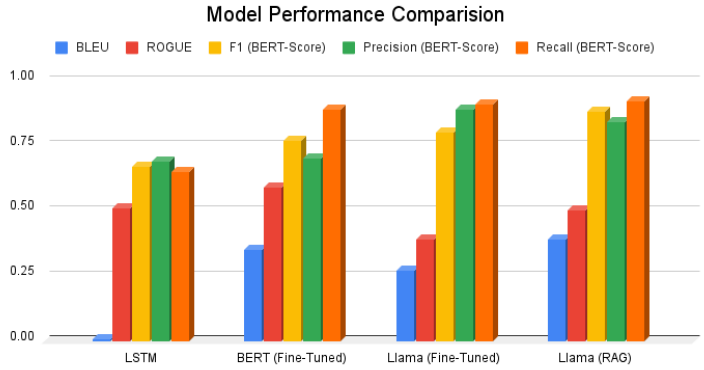

Figure 4: Model Comparison Visualization

The LLaMA model, when fine-tuned, performed very well compared to the baseline LSTM and BERT models. This can be attributed to its key characteristic, RoPE (Rotary Positional Embeddings). This feature not only helps to remember contexts in long sequences but also the relations between words that are placed far apart in sentences. The fine-tuning process ensures the model is trained and performs well in domain-specific tasks, thus producing good BERT scores. While the LLaMA-finetuned model performs poorly when considering BLEU and ROGUE scores, this could be due to the fact that our model is generative, and the sentences and tokens generated by the model will be quite different from the ground truth provided in the dataset. In addition to summarizing the medical texts, the LLaMA models recognized context related to medical history and added AI ethics statements warning users to seek actual professional medical help. These additional statements also contribute to the low BLEU and ROGUE scores, as there involves word-to-word comparison and similarity checks in the score calculation.

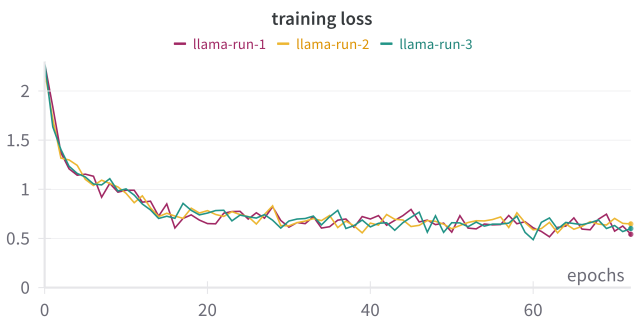

Figure 5: Training Loss for Hyperparameter Tuning

When we consider the performance of LLaMA using the RAG architecture, it performs very similar compared to the fine-tuned version, and in some scenarios will perform better compared to the fine-tuned model. This can be attributed to the additional contextual information that we provide the model while performing the task of summarization, all thanks to the retrieval process that digs the most similar documents form the knowledge base. We have performed hyper-parameter tuning for the llama-2 finetuned model by changing the LoRA constant "r" and the training batch size which can be visualized using figure 5. Where run-1 has the value of r as 8, run-2 has the value of r as 4 and run-3 has the r as 16. Although there is no much difference in the training loss reduction which might be due to the volume of the data we used for training but the value of r=8 has produced the best model.

5 Conclusions

In summary, our experimentation across various models, including LSTM, BERT Fine-Tuned, Llama Fine-Tuned, and Llama with RAG architecture, demonstrates the clear dominance of Llama models over baselines. This superiority is attributed to the RoPE feature, facilitating the retention of long-term dependencies. While BERT Fine-Tuned models benefit from attention mechanisms, LLaMA models, with their fine-tuning process and integration of AI ethics statements, excel in domain-specific tasks, despite lower BLEU and ROGUE scores. Furthermore, the utilization of the RAG architecture enhances Llama's performance by providing additional contextual information through retrieval processes from a knowledge base. These findings underscore the effectiveness of LLaMA models in medical text summarization tasks and highlight the importance of ethical considerations in AI applications.

Although our methodology is restricted by a small dataset comprising around 3000, as a result, there was not much variation in some of the examples, and in particular, the radiology reports included many summaries that were very similar. As such, the models may not be able to generalize well against new medical texts. With more computational power, we would be able to train the models on larger and more varied datasets to enhance their performance on new input. RAG is one exception to this, because it uses the retrieval system from the knowledge base, it demands an updated database to provide a good summarization on the given user query.

In the future, there is a lot of scope to explore the limits of our project, as it is a generalized model, we have not customized it to a particular user group. In other words it would provide very similar outputs to a doctor and a normal user who does not have a general medical knowledge. So, customization of the model to a particular user group can be an area where we could improve the current architecture.

Another area where our project can be scaled to is deploying the project in real time for the users to use and provide a real-time feedback with suggestions on medications, related news-articles, other people's posts on social media, so that users have a complete knowledge on the issue.

References

[1] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13766*, 2020.

[2] James Briggs. How to train bert, Sep 2021.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[5] Hugging Face. Hugging face transformers: Trl documentation. https://huggingface.co/docs/trl/en/sft_trainer, Year Accessed.

[6] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference, 2021.

[7] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997.

[8] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shun Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2022.

[9] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[10] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.

[11] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[12] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics.

[13] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[14] Phillip Schonfelder. Deep learning-based entity recognition in construction regulatory documents. *Simplified, high-level view of the Bert Architecture*, 2021.

[15] D Van Veen, C Van Uden, L Blankemeier, JB Delbrouck, A Aali, C Bluethgen, A Pareek, M Polacin, EP Reis, A Seehofnerová, N Rohatgi, P Hosamani, W Collins, N Ahuja, CP Langlotz, J Hom, S Gatidis, J Pauly, and AS Chaudhari. Clinical text summarization: Adapting large language models can outperform human experts. *NIH*, 2023.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[17] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert, 2020.

A Code Repository

https://github.com/hruday-tej/Clinical-Notes-Summarizer-RAG-FineTune

B Llama Deployed Model

https://www.kaggle.com/models/hrudaytej/llama-finetuned-for-summarization