CMPSC 497 - Final Project Report
Fine Tuning an Open-Source LLM (BART) to generate an approach for a research problem

Hrishikesh Shenai — hms6207@psu.edu

1 Introduction

Formulating a robust research approach is a critical first step in any research endeavor. Researchers often spend considerable time exploring relevant literature, designing methodologies, and refining their approach. This initial stage can be time-consuming and may benefit from automated assistance. This project aims to automate the generation of research approaches given a clearly defined research problem. We leverage the power of large language models (LLMs) by fine-tuning a pre-trained BART [1] model on a custom-built dataset of research problems and corresponding approaches. We evaluate the model's performance using the ROUGE [2], BERTScore [3] and BLEU [4] metrics and provide qualitative analysis of the generated outputs. The project demonstrates the potential of LLMs for accelerating and enhancing the research process. Such a tool could significantly accelerate the research process, allowing researchers to explore multiple potential approaches more efficiently and potentially uncover novel methodologies. This is particularly valuable for interdisciplinary research where navigating diverse methodologies can be challenging. Moreover, an automated approach generation system would be particularly beneficial for early-stage researchers, aiding them in navigating complex research domains efficiently and effectively.

1.1 Problem Statement

The primary objective of this project is to develop an automated system capable of generating detailed, structured research methodologies given a research problem. Specifically, the aim is to:

- Accurately interpret and extract critical research objectives from concise problem statements.
- Generate coherent and comprehensive methodological approaches that include clearly articulated objectives, suggested methodologies, appropriate evaluation metrics, and anticipated challenges.
- Evaluate the effectiveness and practical utility of generated methodologies through both automated metrics and human judgment.
- Critically assess the overall feasibility of this project, analyzing steps that can be taken to improve it.

2 Dataset and Data Curation

This project was unique due to the lack of quality datasets conforming explicitly to the desired format—a clear research problem paired directly with an approach to solve it. Fine-tuning Large Language Models (LLMs) requires a substantial amount of data because these models learn patterns, contextual relationships, and linguistic nuances effectively only when trained on diverse and rich examples.

To overcome this challenge, a custom dataset was constructed by integrating and processing the following three primary datasets:

- **ScisummNet [5] (Yale Scientific Article Summarization Dataset):** This dataset provides over 1,000 scholarly articles from the ACL anthology network, along with manually curated summaries. These summaries effectively encapsulate authors' approaches, methodologies, and insights, directly reflecting solutions to clearly defined research problems.
- **ArXiv Paper Summaries [6]:** Collected directly from the arXiv repository, this dataset contains thousands of academic papers, each with their title, abstract/summary, and subject classifications. Abstracts and summaries explicitly highlight research problems, objectives, methods, and outcomes, providing structured examples useful for training.

- **GPT-4.5 [7] Knowledge Base:** GPT-4.5 (as of June 2024) is trained on hundreds of thousands of research papers, abstracts, and articles from diverse sources such as arXiv, Semantic Scholar, ACL Anthology, among other open-access corpora. Leveraging GPT-4.5's extensive knowledge base, additional synthetic examples were generated by prompting GPT-4.5 with clear instructions and sample queries illustrating desired "problem–approach" structured pairs. This step significantly augmented the dataset diversity and scale.

The rationale behind combining these datasets was two-fold:

- Research paper summaries inherently reflect structured methodological frameworks employed by authors, providing a concrete and practical representation of solving real-world academic problems.
- Utilizing GPT-generated synthetic data supplemented the dataset's scale and diversity, enhancing the model's generalization capability across multiple research domains and methodological styles.

2.1 Data Processing

For the dataset consisting primarily of titles and summaries, raw textual pairs were insufficient. The project's aim required structured queries explicitly stating research problems and outputs clearly delineating structured methodological approaches. Thus, the following preprocessing steps were performed:

- **Titles** were augmented by prepending standardized prompts such as: "Provide an approach for this research problem:" "Devise a strategy to solve the following problem:", etc. This prompt standardization guides the model explicitly in understanding that the subsequent input is a research query demanding a structured and practical approach as output, effectively conditioning the model's response behavior.
- **Summaries** were enhanced by prepending clear contextual indicators such as: "The following approach is based on a research paper:" "To solve this problem, you can follow an approach as adopted in the following:", etc. Adding such clear introductory contexts aids the model in distinguishing the descriptive nature of the methodology and explicitly frames the response as a practical, structured solution derived from existing research, improving coherence, clarity, and alignment to input queries.

Following these processing steps, the final dataset was constructed by systematically combining and formatting the three sources mentioned above, ensuring consistency, clarity, and suitability for effective fine-tuning of the LLM. The combined dataset consisted of ∼65000 rows.

3 LLM Selection

For this project, BART (Bidirectional and Auto-Regressive Transformers), specifically the bart-large-cnn model, was chosen. BART is a transformer encoder-encoder (seq2seq) model with a bidirectional (BERT-like) encoder and an autoregressive (GPT-like) decoder.

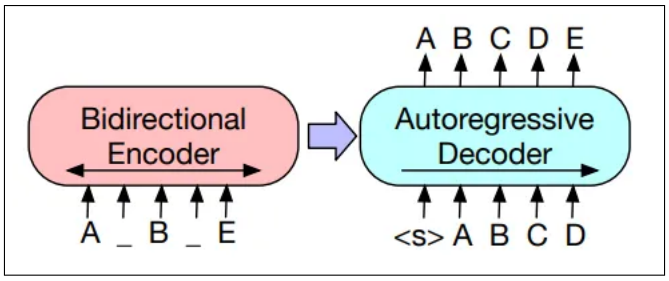


Figure 1: Bart high-level overview [8]

In the encoder, the input sequence undergoes noise transformations wherein random tokens are replaced with mask symbols, effectively corrupting the original sequence. This corrupted sequence is then fed into a bidirectional encoder, which generates contextual representations. Subsequently, an autoregressive decoder computes the likelihood of reconstructing the original, uncorrupted document based on these encoder representations. While alignment between encoder input and decoder output tokens is not strictly required in the pre-training stage, during fine-tuning, both encoder and decoder receive the clean, uncorrupted document as input. In this fine-tuning scenario, the final hidden state representations from the decoder are utilized for downstream tasks.

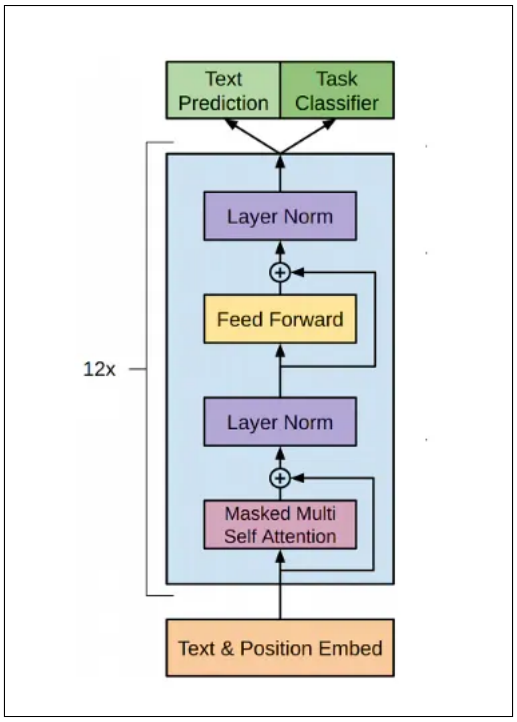


Figure 2: BART model architecture [8]

3.1 Architecture Description:

BART large uses 12 layers in the encoder and decoder as shown in the diagram 2. The bart-large-cnn architecture includes a BART model with separate encoder and decoder blocks, each composed of 12 layers. Both encoder and decoder use self-attention mechanisms (BartSdpaAttention) and GELU activation functions. The embedding layer (BartScaledWordEmbedding) maps token indices to high-dimensional vectors (size 1024). Each encoder and decoder layer contains linear projection layers for attention mechanisms, fully connected layers for feature transformations, and layer normalization for stabilizing learning. The model concludes with a linear language modeling head (lm_head) that projects decoder outputs to a vocabulary distribution for token generation. BART is particularly effective when fine-tuned for text generation (e.g. summarization, translation) but also works well for comprehension tasks (e.g. text classification, question answering).

3.2 Training details:

The Hugging Face **transformers** library for fine-tuning.

- **Fine-tuning Setup**: The Trainer class from the transformers library was employed to manage the training (fine-tuning) process. The dataset constructed in the dataset construction phase was loaded.
- **Tokenizer**: The BARTTokeninzer [1], which is similar to the ROBERTa [9] tokenizer, using byte-level Byte-Pair-Encoding, was used. This tokenizer has been trained to treat spaces like parts of the tokens (a bit like sentencepiece) so a word will be encoded differently whether it is at the beginning of the sentence (without space) or not. Additionally, the tokenizer appends a Beginning of String (BOS) and an End Of String (EOS) token to the data automatically, which are ¡s¿ and ¡/s¿ respectively.
- **Hyperparameters**: The following key hyperparameters:
  - per_device_train_batch_size: 8
  - per_device_eval_batch_size: 8
  - num_train_epochs: 5
  - metric_for_best_model=”rouge1”, ”bertscore”, ”bleu”

- **Hardware**: The model was trained on Google Colab's [10] L4 GPU with 22.5 GB RAM and 53 GB system RAM. The fine-tuned model was saved explicitly after training, including tokenizer (Bart tokenizer) and configuration details to facilitate correct reloading and inference.

3.3 Training steps:

- **Load Dataset**: The dataset from the CSV file was loaded and split into train and eval sets in the ratio 80:20.
- **Load Pre-trained Model**: The pretrained "facebook/bart-large-cnn" model was loaded from the Huggingface's transformers library. The model was then moved to GPU using torch.
- **Data Preprocessing**: The dataset was then tokenized using the BartTokenizer. To maintain uniformity, the input size was fixed to 512 tokens and the output size was fixed to 1024 tokens (due to hardware constraints). Inputs/Outputs less than their respective sizes were padded, and those with lengths greater than the respective sizes were truncated.
- **Loading Evaluation Metrics**: The above mentioned evalutation metrics were loaded using the evaluate library. Simple evalution functions were written that decode the predicted tokens and the ground truth tokens, and compute the similarity based on the respective metrics.
- **Training and Evaluation**: The training was initiated using the above-mentioned hyperparameters. Wandb [11] (AI developer platform) was used to track and monitor the training process. Post every epoch, the model was evaluated using the evaluation metrics.

4 Evaluation and Metrics

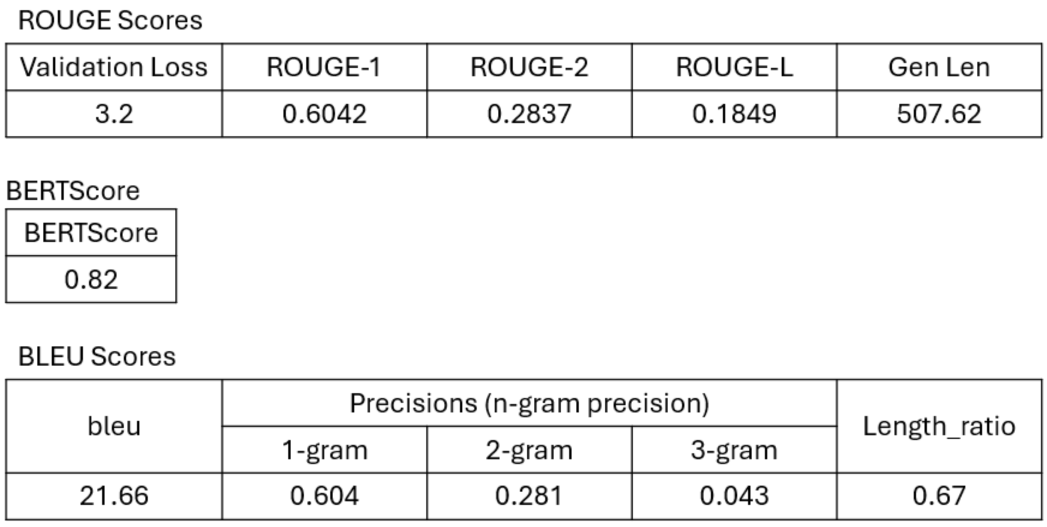
4.1 Quantitative Analysis

For quantitative analysis, three evaluation metrics were considered:

- **ROUGE (Recall-Oriented Understudy for Gisting Evaluation)**: Measures overlap of n-grams between generated and reference texts. ROUGE-L focuses specifically on longest common subsequences, assessing fluency and relevance.
- **BERTScore**: A semantic similarity metric leveraging BERT embeddings to measure the contextual similarity between generated texts and references, thus capturing deeper semantic relevance.
- **BLEU (Bilingual Evaluation Understudy)**: Evaluates the precision of n-grams overlap between generated texts and reference texts, providing an indication of linguistic accuracy.

4.1.1 ROUGE Scores Analysis:

- **Validation Loss**: Measures how well the model fits the validation data. A value like 3.20 is quite decent considering the scale of the output.
- **ROUGE-1**: Measures unigram (single word) overlap between generated text and reference. A score of 60.42 means about 60% of words in the reference are found in the generated text.
- **ROUGE-2**: Measures bigram (two consecutive words) overlap, indicating fluency and some coherence. A score of 28.37 is modest — common for long outputs.
- **ROUGE-L**: Measures longest common subsequence (LCS) between generated and reference text. Reflects ability to generate sequences that match the structure/order of the target. 18.49 is acceptable, though there's room to improve alignment.
- **Gen Len (Generation Length)**: The average number of tokens generated per sample. 507.62 tokens is a good length for complex output like research approaches, especially considering the output tokenizer was limited to a size of 1024 due to hardware constraints. The model can be forced to generate responses of max (1024 tokens) length, however in such situation the model struggles to output acceptable responses.



ROUGE Scores

Validation Loss	ROUGE-1	ROUGE-2	ROUGE-L	Gen Len
3.2	0.6042	0.2837	0.1849	507.62

BERTScore

BERTScore
0.82

BLEU Scores

bleu	Precisions (n-gram precision)			Length_ratio
	1-gram	2-gram	3-gram	
21.66	0.604	0.281	0.043	0.67

Figure 3: Evaluation scores summary

4.1.2 Bertscore analysis:

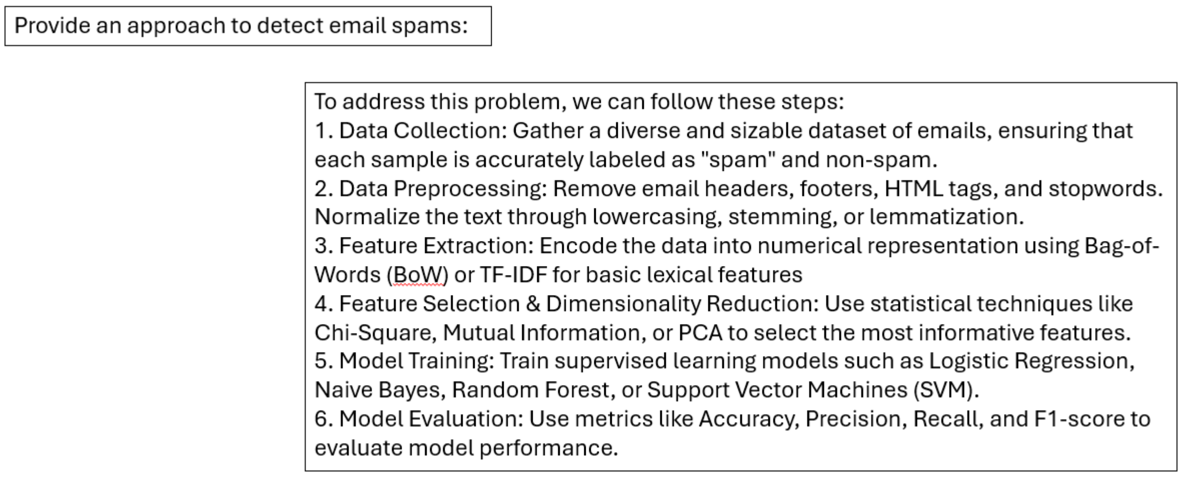
BERTScore evaluates text similarity using contextual embeddings from models like BERT or RoBERTa. Instead of comparing exact words (like BLEU or ROUGE), it compares the meaning of words and phrases. It calculates precision, recall, and F1 based on the cosine similarity of embeddings between generated text and reference. A BERTScore of 0.82 is actually quite good — especially for open-ended text generation tasks, where outputs can be semantically correct even if they don't match the reference exactly.

4.1.3 Bleu Scores Analysis:

- **bleu**: This is the final BLEU score (between 0 and 1). 21.66% BLEU is low, which suggests the generated text is not matching well with the reference — either lexically or structurally. However, BLEU is exponential in behavior; small increases represent big quality improvements.
- **precisions (n-gram precision)**: Measures unigram (single word) overlap between generated text and reference. A score of 60.42 means about 60% of words in the reference are found in the generated text. These represent how many overlapping n-grams exist between the hypothesis and reference: 1-gram (unigram): 60.4% → About 60% the single words overlap 2-gram: 28.1% → Lower overlap of consecutive word pairs 3-gram: 4.3% This steep drop-off is typical, probably as the model is generating overly generic text or the output has different phrasing than the reference (BLEU is brittle to rewording)
- **length_ratio**: Measures ratio of the length of the output produced by the model to the ground truth references. A length ratio of 0.67 is quite decent, again considering the output was hardlimitted to 1024 tokens.

4.2 Qualitative Analysis

Upon evaluating the model's outputs qualitatively as observed in figures 4 and 5, it was observed that while the generated approaches generally followed a logical structure and addressed the core problem, they tended to be brief in detail. The model is able to produce step-wise procedural responses, indicating that it had learned the expected format of a solution-oriented task. However, the explanations lacked depth, domain-specific insights, or justifications behind the chosen steps. Attempts to increase the length of the outputs by adjusting decoding parameters (e.g., max token length) led to issues such as incoherence or repetitive phrasing, suggesting that the model struggles to maintain semantic consistency when forced to generate longer responses. This behavior may be attributed to limitations in training data richness, hard-limitting output token size or overfitting on shorter examples. To improve output quality, future iterations may benefit from augmenting the dataset with more diverse, detailed, and well-structured


Provide an approach to detect email spams:

To address this problem, we can follow these steps:
1. Data Collection: Gather a diverse and sizable dataset of emails, ensuring that each sample is accurately labeled as "spam" and non-spam.
2. Data Preprocessing: Remove email headers, footers, HTML tags, and stopwords. Normalize the text through lowercasing, stemming, or lemmatization.
3. Feature Extraction: Encode the data into numerical representation using Bag-of-Words (BoW) or TF-IDF for basic lexical features
4. Feature Selection & Dimensionality Reduction: Use statistical techniques like Chi-Square, Mutual Information, or PCA to select the most informative features.
5. Model Training: Train supervised learning models such as Logistic Regression, Naive Bayes, Random Forest, or Support Vector Machines (SVM).
6. Model Evaluation: Use metrics like Accuracy, Precision, Recall, and F1-score to evaluate model performance.

Figure 4: Output generated by LLM post fine-tuning (1).

examples, as well as incorporating decoding strategies like nucleus sampling or beam search to enhance fluency without compromising relevance.

5 Challenges and Lessons learned

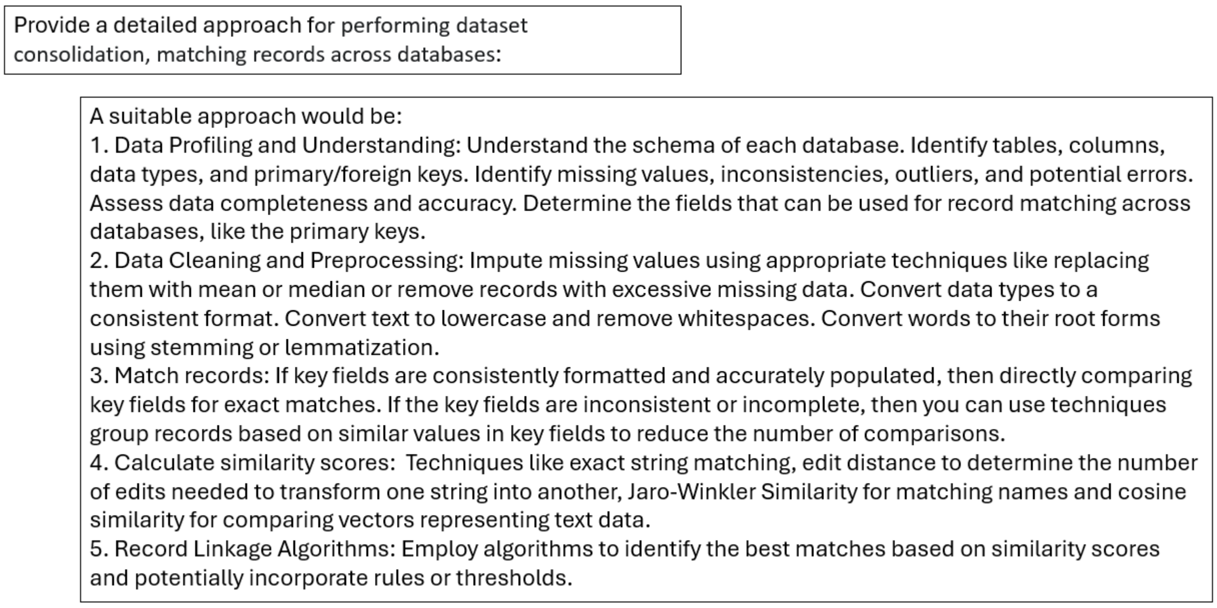
Throughout this project, several challenges were encountered, each leading to valuable insights that helped refine the approach.

- **LLM selection:** There are a vast majority of open-source LLMs available, each having its own features and contraints. Selecting an LLM to fine-tune on a particular task was crucial. This process gave valuable insights into various LLM architecture, helping to understand their capability.
- **Buliding Customized Dataset:** This project highlighted the importance of high-quality data for fine tuning effective LLMs. Building the dataset was the most time-consuming aspect of the project, but it was crucial for achieving good performance.
- **Computation contraints:** Fine-tuning a large language model like BART can be computationally intensive, necessitating careful management of resources. This also encouraged the exploration of checkpoint-based training.
- **Evaluation Strategies:** Evaluating the efficacy of the generative capabilities of LLM is not straightforward and tends to be rather subjective. This project helped to explore various evaluation strategies and how they work.

6 Critical assessment and potential issues with the project

The project was an interesting challenge, testing the capabilities of open-source LLMs to generate insightful approaches. The following highlight some potential issues with the project as identified:

- **Dependence on Existing Data:** LLMs are trained on vast amounts of existing text and code. They learn to generate text that is similar to what they have seen before. This makes it difficult for them to come up with truly original ideas that deviate significantly from established methodologies. They excel at synthesizing and recombining existing knowledge, but not at creating fundamentally new knowledge.
- **Lack of "Understanding":** Open-source LLMs do not possess genuine understanding of the research problems they are presented with. They process text statistically, identifying patterns and relationships between words. They lack the deep domain expertise, critical thinking skills, and intuitive leaps that drive innovative research.


Provide a detailed approach for performing dataset consolidation, matching records across databases:

A suitable approach would be:
1. Data Profiling and Understanding: Understand the schema of each database. Identify tables, columns, data types, and primary/foreign keys. Identify missing values, inconsistencies, outliers, and potential errors. Assess data completeness and accuracy. Determine the fields that can be used for record matching across databases, like the primary keys.
2. Data Cleaning and Preprocessing: Impute missing values using appropriate techniques like replacing them with mean or median or remove records with excessive missing data. Convert data types to a consistent format. Convert text to lowercase and remove whitespaces. Convert words to their root forms using stemming or lemmatization.
3. Match records: If key fields are consistently formatted and accurately populated, then directly comparing key fields for exact matches. If the key fields are inconsistent or incomplete, then you can use techniques group records based on similar values in key fields to reduce the number of comparisons.
4. Calculate similarity scores: Techniques like exact string matching, edit distance to determine the number of edits needed to transform one string into another, Jaro-Winkler Similarity for matching names and cosine similarity for comparing vectors representing text data.
5. Record Linkage Algorithms: Employ algorithms to identify the best matches based on similarity scores and potentially incorporate rules or thresholds.

Figure 5: Output generated by LLM post fine-tuning (2).

- **Lack of effective dataset:** As mentioned in the dataset section, the absence of relevant datasets hinders the ability to effectively fine-tune an LLM for generating approaches to research problems. While a dataset was constructed for training, its quality and diversity are limited, potentially restricting the model's ability to generalize.
- **Evaluation Challenges:** Evaluating the novelty of a generated research approach is inherently subjective and difficult to automate. Existing metrics like ROUGE and BERTScore primarily measure similarity to existing text, not originality or innovativeness.
- **Limited Creativity:** Open-source LLMs are good at pattern recognition and interpolation, but true creativity often involves breaking established patterns and making unexpected connections. LLMs struggle with this type of out-of-the-box thinking.
- **Ambiguity in research problems:** The efficacy of the approach produced by the LLM highly depends on the precision with which the research problem is described to model. Since the input text is generally short, it can lead to ambiguity, affecting the relevance of the output produced.

Some ideas that can be explored deeply to improve the project:

- **Incorporate Domain-Specific Knowledge:** Fine-tuning LLMs on highly specialized datasets within a specific research area could help them generate more tailored and potentially novel approaches within that domain.
- **Integrate with External Knowledge Sources:** Connect LLMs to knowledge graphs, databases, or ontologies to provide them with structured information and reasoning capabilities beyond their training data. This could help them make new connections and generate more informed approaches.
- **Prompt Engineering for Creativity:** Develop advanced prompting techniques that encourage the LLM to explore unconventional ideas. This could involve prompts that explicitly ask for unusual or innovative approaches, or prompts that incorporate constraints or challenges to stimulate creative problem-solving.
- **Incorporate Domain-Specific Knowledge:** Train LLMs using reinforcement learning with human feedback. Reward the model for generating novel and feasible approaches, as judged by human experts. This iterative feedback loop can guide the LLM towards generating more creative outputs.

7 Conclusion

Throughout this project, significant insights were gained into the workings of LLMs, including how to effectively fine-tune and use them for specialized tasks. The process underscored the crucial role that a well-structured and relevant dataset plays in achieving meaningful results, highlighting the challenges posed by dataset scarcity or inadequacy. Moreover, it emphasized the critical need for adequate computational hardware resources, particularly GPUs, which directly impact the feasibility and efficiency of training complex models.

As a future scope, we could explore leveraging more powerful and efficient open-source LLMs such as LLaMA 3 [12], Mistral [13], and Falcon [14], potentially leading to improved performance in generating structured research methodologies. Additionally, more sophisticated utilization of GPT models (such as GPT-4.5 or subsequent releases) for advanced data augmentation and preprocessing could effectively address the persistent challenges of dataset scarcity, significantly enhancing training effectiveness and the practical utility of resulting models.

References

[1] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880, Online. Association for Computational Linguistics.

[2] Lin, Chin-Yew. (2004). ROUGE: A Package for Automatic Evaluation of summaries. Proceedings of the ACL Workshop: Text Summarization Braches Out 2004. 10.

[3] Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q. & Artzi, Y. (2019). BERTScore: Evaluating Text Generation with BERT.. CoRR, abs/1904.09675.

[4] Papineni, Kishore & Roukos, Salim & Ward, Todd & Zhu, Wei Jing. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. 10.3115/1073083.1073135.

[5] Yasunaga, Michihiro & Kasai, Jungo & Zhang, Rui & Fabbri, Alexander & Li, Irene & Friedman, Dan & Radev, Dragomir. (2019). ScisummNet: A Large Annotated Dataset and Content-Impact Models for Scientific Paper Summarization with Citation Networks. 10.48550/arXiv.1909.01716.

[6] Soumik Rakshit, Sayak Paul. 2021. arXiv Paper Abstracts. https://www.kaggle.com/datasets/spsayakpaul/arxiv-paper-abstracts

[7] OpenAI. 2025. ChatGPT. [Large language model]. https://chat.openai.com/chat

[8] Nadira Povey. 2022. BART Model Architecture. https://medium.com/@nadirapovey/bart-model-architecture-8ac1cea0e877

[9] Liu, Yinhan & Ott, Myle & Goyal, Naman & Du, Jingfei & Joshi, Mandar & Chen, Danqi & Levy, Omer & Lewis, Mike & Zettlemoyer, Luke & Stoyanov, Veselin. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. 10.48550/arXiv.1907.11692.

[10] Google. (2025). Google Colaboratory. https://colab.research.google.com/

[11] Weights & Biases. (2025). https://docs.wandb.ai/company/academics#cite-weights-and-biases

[12] Stefan Emil, Repede & Brad, Remus. (2024). LLaMA 3 vs. State-of-the-Art Large Language Models: Performance in Detecting Nuanced Fake News. Computers. 13. 292. 10.3390/computers13110292.

[13] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lelio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut ´ Lavril, Thomas Wang, Timothee Lacroix, and William El Sayed. Mistral 7B. ´ arXiv preprint, 2023a. URL https://arxiv.org/abs/2310.06825.

[14] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Hamza Alobeidli, Alessandro Cappelli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for falcon LLM: outperforming curated corpora with web data only. In Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS '23). Curran Associates Inc., Red Hook, NY, USA, Article 3464, 79155–79172.