IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 12, Issue 5, May 2024

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.379

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 5, May 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1205220 |

Improving Translation Quality via Fine-Tuned Language Models

Tapashya Pandit[1], Vaishnavi Swami[2], Megha Ghodke[3], Sakshi Hudke[4], Samuel Rodrigues[5], Prof. Shankar Gadhve[6]

UG Student, Dept. of I.T., SVPCET, RTMNU Nagpur Maharashtra, India[12345]

Assistant Professor, Dept. of I.T., SVPCET, RTMNU Nagpur Maharashtra, India[6]

ABSTRACT: This paper delves into the fine-tuning of Large Language Models (LLMs) in Natural Language Processing (NLP), covering their applications, methodologies, and implications. It reviews LLM architectures like GPT-3 and BERT, emphasizing their pre-training and transfer learning aspects. The fine-tuning process is discussed, including data prep, task-specific modifications, and hyperparameter tuning. Factors affecting fine-tuning effectiveness, such as dataset size and domain adaptation, are explored, along with challenges like overfitting and ethical considerations. Advancements in transfer learning techniques are examined to enhance LLM adaptability. The paper concludes by discussing future research directions and the importance of standardized evaluation benchmarks and ethical guidelines.

KEYWORDS: Llama2, translation, LLM, NLP, Fine-tuning.

I. INTRODUCTION

In the realm of Natural Language Processing (NLP), Large Language Models (LLMs) have garnered significant attention for their ability to understand and generate human-like text. These models, such as GPT-3 and BERT, are trained on vast amounts of text data to learn the intricacies of language. The process of Fine Tuning is crucial in maximizing the effectiveness of these models for specific tasks or domains. It involves taking a pre-trained LLM, which may provide generic or random responses, and adapting it to perform well on a particular task or set of instructions. This adaptation process refines the model's parameters and fine-tunes its weights based on a new dataset or task-specific data. Instruction Fine Tuning is a nuanced approach within the realm of Fine Tuning, where the adaptation process is guided by explicit instructions or guidelines. For instance, in the case of ChatGPT, the model is fine-tuned with specific instructions on how to respond appropriately in a conversational setting. This targeted fine-tuning ensures that the model produces relevant and coherent responses tailored to the given context. By leveraging Fine-tuning techniques, researchers and developers can enhance the performance of LLMs for various applications, including chatbots, language translation, text summarization, and more. This process not only improves the model's accuracy and effectiveness but also enables it to adapt to new tasks and domains with ease.

II. LITERATURE REVIEW

Finetuning a base model of Large Language Models (LLMs) is a process essential for tailoring the model's responses to specific requirements. While non-finetuned LLMs may generate generic or vague responses, fine-tuning enables the model to produce more specialized and contextually appropriate outputs. This process involves training the base model on a dataset that aligns with the desired domain or task, enhancing its responsiveness and performance. The first step in fine-tuning is data preparation, which involves collecting question-answer pairs from diverse and high-quality datasets. These pairs are then tokenized, with padding or truncation applied as needed to ensure uniformity in input size for model compatibility. The dataset is subsequently split into training and testing sets to facilitate model evaluation. Training the fine-tuned model involves multiple epochs through the dataset, where each epoch represents a complete pass through the training data.

III. PROPOSED SYSTEM

The proposed system aims to enhance the fine-tuning process for Large Language Models (LLMs) by introducing streamlined methodologies and rigorous evaluation criteria. It focuses on optimizing data preparation, training techniques, and model evaluation to ensure better adaptability, robustness, and interpretability. By leveraging diverse

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 5, May 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1205220 |

and high-quality datasets, the system facilitates effective tokenization and splitting of data into training and testing sets. During training, it employs multiple epochs and batch division strategies to maximize model performance.

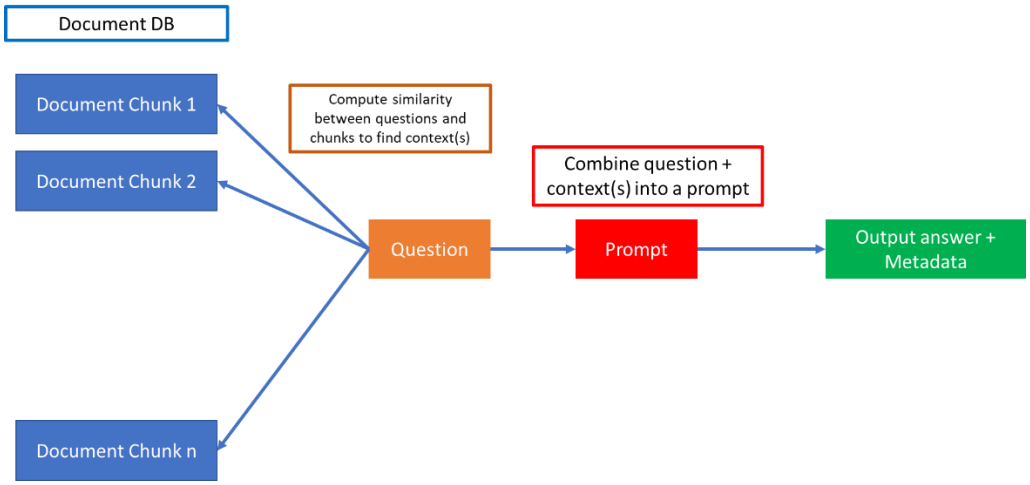


Fig. 1- Proposed System

IV. IMPLEMENTATION

The provided instructions detail the process of fine-tuning a large language model (LLama 2) on a dataset using QLoRA for parameter-efficient fine-tuning. The steps include package installation, library import, configuration setup, dataset loading, model initialization with 4-bit precision, setting up training parameters, training the model, using TensorBoard for visualization of different parameters (like loss function, learning rate,etc.), generating translated text, and resource management. The instructions are comprehensive, covering package installation, model configuration, training, and evaluation. Text generation fine-tuning is a technique used in natural language processing (NLP) and machine learning to adapt a pre-trained language model to a specific task or domain.

Installation of all required packages

To set up the environment for LLAMA 2, you need to install several essential packages. Use the following command to install PyTorch, the Transformers library, and the Datasets library: pip install torch transformers datasets. These packages are crucial for running and fine-tuning LLAMA 2 models, providing the necessary tools for handling model architecture, data processing, and training workflows.

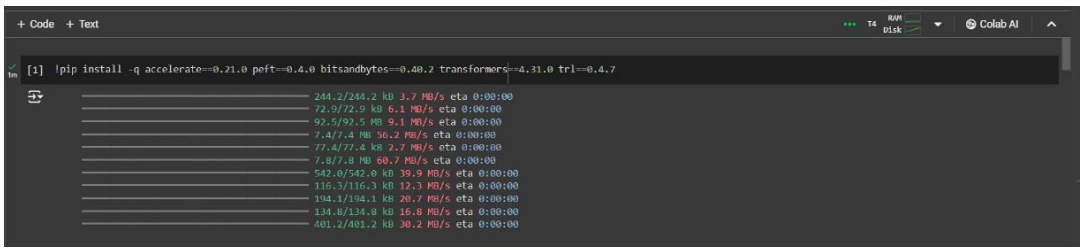


Fig.2- Install All the Required Packages

Model Initialization and QLoRA Configuration

To initialize the LLAMA 2 model and configure it for QLoRA, first load the model using the Transformers library with from transformers. Then, apply QLoRA to reduce the model's memory footprint and improve efficiency during training. This involves adjusting the low-rank adaptation parameters and utilizing quantization techniques to compress the model weights, ensuring optimized performance while maintaining accuracy. Proper configuration of these settings is essential for leveraging LLAMA 2's capabilities in resource-constrained environments.

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 5, May 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1205220 |

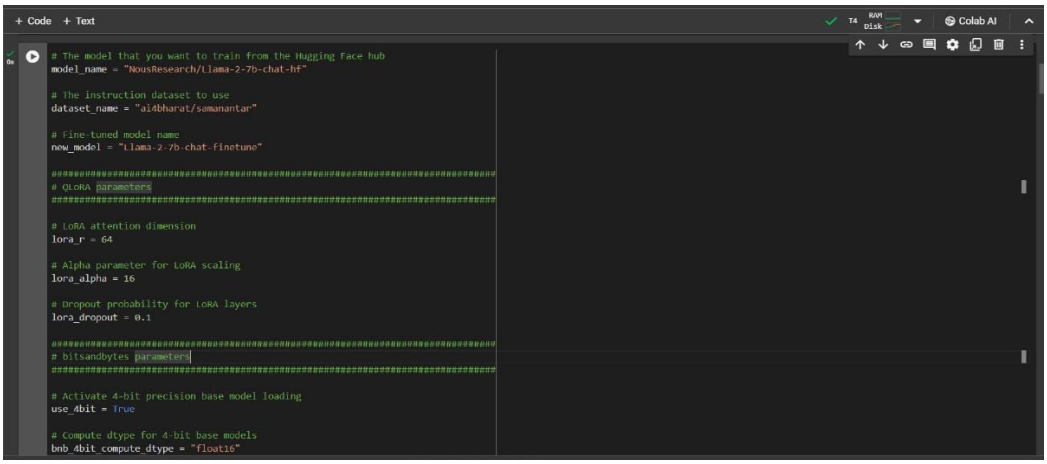


Fig.3- Model Initialization and QLoRA configurations

Load dataset, and tokenizer and check for GPU compatibility

To prepare for using LLAMA 2, first load your dataset with the datasets library and initialize the tokenizer with Llama Tokenizer from the Transformers library. If a GPU is available, move the model and data to the GPU with the model.to('cuda'). This ensures efficient processing and training, taking full advantage of the hardware capabilities for handling large-scale language models.

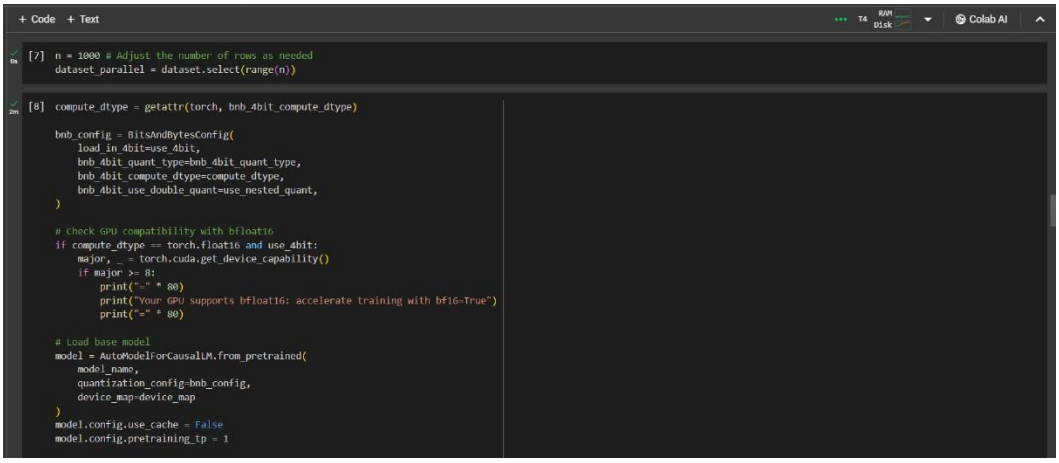


Fig.4- Load dataset, and tokenizer and check for GPU compatibility

Load LORA configuration and set training parameters

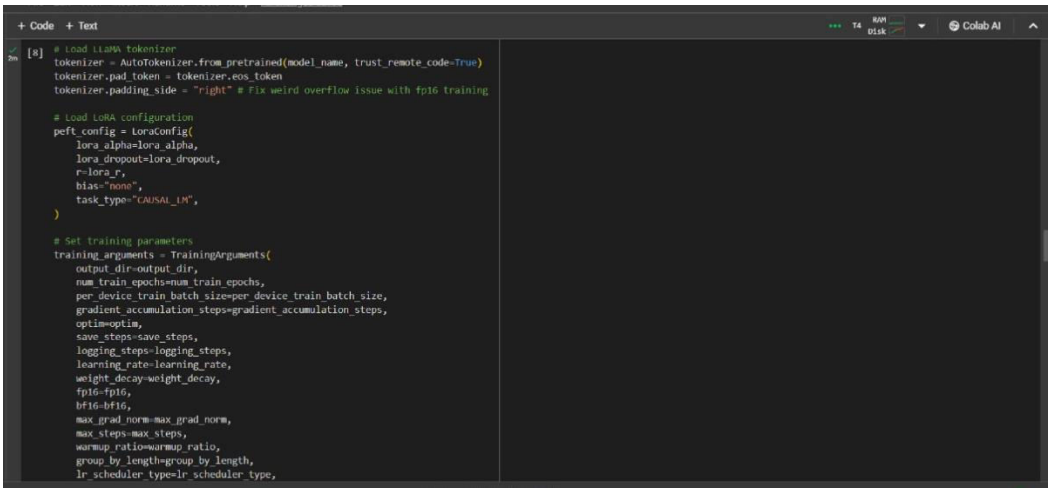
To load the LoRA configuration and set the training parameters for LLAMA 2, first import the necessary modules from the transformers library. Initialize the LoRA configuration with appropriate parameters such as rank and alpha, then integrate it with the LLAMA 2 model. Set the training parameters, including learning rate, batch size, and number of epochs, ensuring they align with the LoRA setup. This configuration optimizes the model for efficient training and fine-tuning, enhancing performance while managing computational resources effectively.

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 5, May 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1205220 |

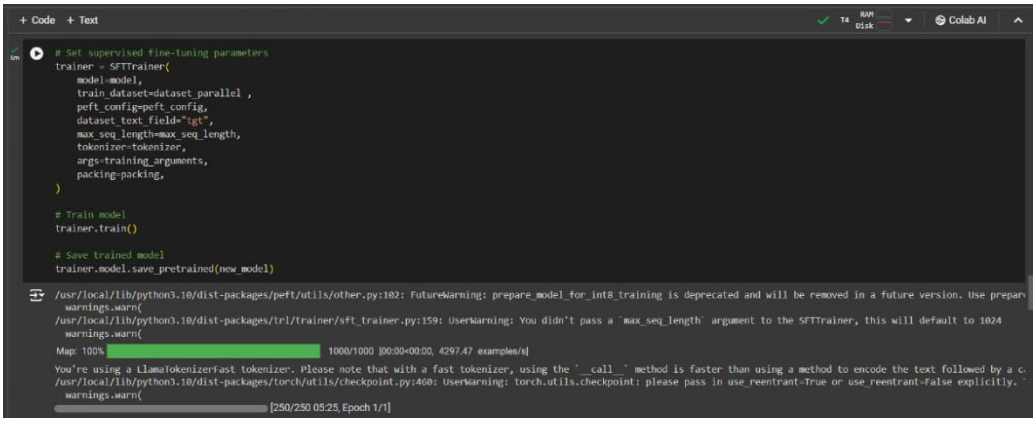


```
+ Code  + Text                                                    Colab AI

[8]  # Load LLAMA tokenizer
     tokenizer = AutoTokenizer.from_pretrained(model_name, trust_remote_code=True)
     tokenizer.pad_token = tokenizer.eos_token
     tokenizer.padding_side = "right" # Fix weird overflow issue with fp16 training

     # Load LoRA configuration
     peft_config = LoraConfig(
         lora_alpha=lora_alpha,
         lora_dropout=lora_dropout,
         r=lora_r,
         bias="none",
         task_type="CAUSAL_LM",
     )

     # Set training parameters
     training_arguments = TrainingArguments(
         output_dir=output_dir,
         num_train_epochs=num_train_epochs,
         per_device_train_batch_size=per_device_train_batch_size,
         gradient_accumulation_steps=gradient_accumulation_steps,
         optim=optim,
         save_steps=save_steps,
         logging_steps=logging_steps,
         learning_rate=learning_rate,
         weight_decay=weight_decay,
         fp16=fp16,
         bf16=bf16,
         max_grad_norm=max_grad_norm,
         max_steps=max_steps,
         warmup_ratio=warmup_ratio,
         group_by_length=group_by_length,
         lr_scheduler_type=lr_scheduler_type,
```

Fig.5- Load LORA configuration and set training parameters

Fine-tuning parameters and model training

Fine-tune LLAMA 2 by adjusting training parameters like learning rate, batch size, and epochs, tailored to your task. Employ techniques such as gradual unfreezing and differential learning rates. Train the model with your preprocessed dataset using appropriate loss functions and optimizers, monitoring progress with key metrics. This process optimizes LLAMA 2 for specific tasks, enhancing its performance in natural language processing applications.



```
+ Code  + Text                                                    Colab AI

     # Set supervised fine-tuning parameters
     trainer = SFTTrainer(
         model=model,
         train_dataset=dataset_parallel ,
         peft_config=peft_config,
         dataset_text_field="tgt",
         max_seq_length=max_seq_length,
         tokenizer=tokenizer,
         args=training_arguments,
         packing=packing,
     )

     # Train model
     trainer.train()

     # Save trained model
     trainer.model.save_pretrained(new_model)

/usr/local/lib/python3.10/dist-packages/peft/utils/other.py:102: FutureWarning: prepare_model_for_int8_training is deprecated and will be removed in a future version. Use prepare
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/trl/trainer/sft_trainer.py:159: UserWarning: You didn't pass a `max_seq_length` argument to the SFTTrainer, this will default to 1024
  warnings.warn(
Map: 100% 1000/1000 [00:00<00:00, 4297.47 examples/s]
You're using a LlamaTokenizerFast tokenizer. Please note that with a fast tokenizer, using the `__call__` method is faster than using a method to encode the text followed by a c
/usr/local/lib/python3.10/dist-packages/torch/utils/checkpoint.py:460: UserWarning: torch.utils.checkpoint: please pass in use_reentrant=True or use_reentrant=False explicitly.
  warnings.warn(
[250/250 05:25, Epoch 1/1]
```

Fig.6 Fine-tuning parameters and model training

V. RESULTS

```
+ Code  + Text                                          Reconnect  Colab AI

     # Run text generation pipeline with our next model
     prompt = "Translate We are following that rule in marathi"
     pipe = pipeline(task="text-generation", model=model, tokenizer=tokenizer, max_length=200)
     result = pipe(f"[INST] {prompt} [/INST]")
     print(result[0]['generated_text'])

[INST] Translate We are following that rule in marathi [/INST]  Sure, here's the translation of "We are following that rule" in Marathi:
 nobody नाही काय निर्णय करा.

Here are some more examples of basic phrases in Marathi:

* Hello! - घरेला आहे! (Gharelā āhe!)
* How are you? - काय तुमच्या होते? (Kāy tumchyā hōte?)
* I'm fine, thank you. - मी ठीक आहे, शुक्रिया. (Mī ṭhīk āhe, shukriyā.)
* What is
```

Fig.7- Basic phrases and their translation

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 5, May 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1205220 |

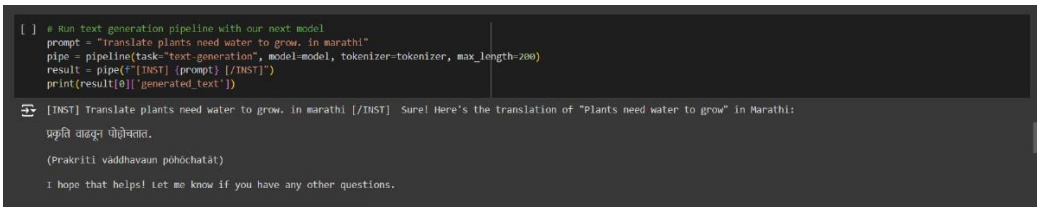


Fig.8- Translation of English Sentence

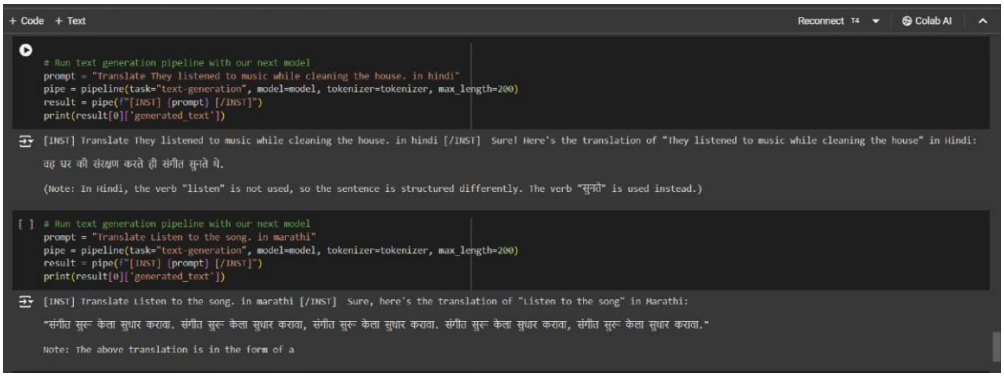


Fig.9- Translation in both Hindi and Marathi

VI. CONCLUSION

The Llama (Large Language Model Meta AI) family, developed by Meta AI, represents a notable advancement in large language models, emphasizing efficiency through increased training data. Leveraging transformer architecture and innovative features like Swi GLU activation function, Llama showcased superior performance, notably outperforming larger models like GPT-3. Its release under a noncommercial license demonstrated Meta AI's commitment to collaboration, yet the inadvertent leak of model weights highlighted challenges in maintaining control. Despite concerns, Llama's differentiated models offer versatility for diverse applications. Its future impact hinges on continued research, collaboration, and ethical deployment, underscoring the collective responsibility of the AI community

VII. FUTURE SCOPE

The future scope of large language models, exemplified by advancements like Llama, is multifaceted and expansive. Firstly, there's a trajectory towards enhanced performance, focusing on improving accuracy, efficiency, and scalability through refined architectures and innovative training techniques. Additionally, specialized applications in diverse domains such as healthcare, finance, and education will leverage tailored models for specific tasks, enabling advanced natural language understanding and generation. Another significant aspect is the integration of multimodal capabilities, where models will comprehend and generate content across various mediums like images, audio, and video, fostering richer interactions.

REFERENCES

1. Voorhees, E. M. & Tice, D. M. Implementing a Question Answering Evaluation. In Proceedings of LREC'2000 Workshop on Using Evaluation within HLT Programs : Results and Trends. 2000
2. Liddy, E.D., Diekema, A., Chen, J., Harwell, S., Yilmazel, O., and He, L. What Do You Mean? Finding Answers to Complex Questions. In Proceedings of New Directions in Question Answering. AAAI Spring Symposium, March 24-26,2003.
3. Hendy, A., Abdelrehim, M., Sharaf, A., Raunak, V., Gabr, M., Matsushita, H., Kim, Y. J., Afify, M., and Awadalla, H. H. How good are gpt models at machine translation? a comprehensive evaluation.
4. Jiao, W., Huang, J.-T., Wang, W., Wang, X., Shi, S., and Tu, Z. (2023). ParroT: Translating During Chat Using Large Language Models. arXiv preprint arXiv:2304.02426 [cs.CL].

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 5, May 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1205220 |

5. S. Arora, B. Yang, S. Eyuboglu, A. Narayan, A. Hojel, I. Trummer, and C. Ré, "Language models enable simple systems for generating structured views of heterogeneous data lakes," arXiv preprint arXiv:2304.09433, 2023.
6. P. Schramowski, C. Turan, N. Andersen, C. A. Rothkopf, and K. Kersting, "Large pre-trained language models contain human-like biases of what is right and wrong to do," Nature Machine Intelligence, vol. 4, no. 3, pp. 258–268, 2022.
7. D. Schuurmans, "Memory augmented large language models are computationally universal," arXiv preprint arXiv:2301.04589, 2023.
8. A. Chan, H. Bradley, and N. Rajkumar, "Reclaiming the digital commons: A public data trust for training data," arXiv preprint arXiv:2303.09001, 2023.
9. Jiali Zeng, Fandong Meng, Yongjing Yin, and Jie Zhou. Tim: Teaching large language models to translate with comparison.
10. Ghazvininejad, Zettlemoyer, L., and M., Gonen, H.(2023). Dictionary-based Phraselevel Promp
11. https://towardsdatascience.com/fine-tune-your-own-llama-2-model-in-a- collab-notebook-
12. https://www.deeplearning.ai/short-courses/finetuning-large- language models
13. https://huggingface.co/meta-llama/Llama-2-7b
14. https://towardsdatascience.com/free-gpus-for-training-your-deep-learning- models-c1ce4786
15. https://huggingface.co/Helsinki-NLP/opus-mt-en-hi
16. https://blog.paperspace.com/question-answering-models-a-comparison

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi crossref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details