Project Report:

PROBLEM STATEMENT:

Introduction to GenAI and Simple LLM Inference on CPU and Finetuning of LLM Model to Create a Custom Chatbot

1. Introduction

Generative AI (GenAI) refers to a class of artificial intelligence models capable of generating new data instances that resemble a given dataset. It encompasses a variety of techniques, including text generation, image synthesis, and more, enabling machines to create content. Large Language Models (LLMs) are a significant subset of GenAI, focusing on generating and understanding human language.

This project report details the development and implementation of a custom chatbot using the Mistral 7B LLM model. The project involves the introduction to GenAI, simple LLM inference on CPU, and the finetuning of the LLM model using the Hugging Face dataset.

2. Objectives

- To understand the fundamentals of GenAI and LLMs.
- To perform simple LLM inference on CPU.
- To finetune the LLM model to create a custom chatbot.
- To demonstrate the practical application of LLMs in chatbot development.

3. Methodology

Tools and Technologies

- Generative AI and LLM:

    Mistral 7B, Hugging Face

- Programming Languages:

    Python

- Libraries and Frameworks:

    Transformers (Hugging Face), PyTorch, Accelerate, Peft, Bitsandbytes, TRL, Auto-GPTQ, Optimum

-Development Environment:

    Jupyter Notebook, Google Colab

4. Steps

4.1. Loading the Dataset:

The Hugging Face dataset is loaded into the environment for analysis and model training.

```
from datasets import load_dataset, Dataset

data = load_dataset("tatsu-lab/alpaca", split="train")

data_df = data.to_pandas()

data_df = data_df[:5000]

data_df["text"] = data_df[["input", "instruction", "output"]].apply(lambda x:
"###Human: " + x["instruction"] + " " + x["input"] + " ###Assistant: "+ x["output"],
axis=1)

data = Dataset.from_pandas(data_df)
```

4.2. Model Initialization:

The Mistral 7B model is initialized using the Transformers library.

```
from transformers import AutoModelForCausalLM, AutoTokenizer, GPTQConfig

tokenizer = AutoTokenizer.from_pretrained("TheBloke/Mistral-7B-Instruct-v0.1-GPTQ")

tokenizer.pad_token = tokenizer.eos_token

quantization_config_loading = GPTQConfig(bits=4, disable_exllama=True, tokenizer=tokenizer)

model = AutoModelForCausalLM.from_pretrained(

  "TheBloke/Mistral-7B-Instruct-v0.1-GPTQ",

  quantization_config=quantization_config_loading,
```

4.3. Simple LLM Inference on CPU:

Performing basic text generation tasks using the Mistral 7B model on a CPU.

```
inputs = tokenizer("Hello, how are you?", return_tensors="pt")

outputs = model.generate(inputs['input_ids'], max_length=50)

print(tokenizer.decode(outputs[0], skip_special_tokens=True))
```

4.4. Model Finetuning:

The model is finetuned on the specific dataset to enhance its performance for the chatbot application.

```
from peft import LoraConfig, AutoPeftModelForCausalLM,
prepare_model_for_kbit_training, get_peft_model

from transformers import TrainingArguments

from trl import SFTTrainer

model.config.use_cache=False

model.config.pretraining_tp=1

model.gradient_checkpointing_enable()

model = prepare_model_for_kbit_training(model)

peft_config = LoraConfig(

   r=16, lora_alpha=16, lora_dropout=0.05, bias="none", task_type="CAUSAL_LM",
target_modules=["q_proj", "v_proj"]

)

model = get_peft_model(model, peft_config)

training_arguments = TrainingArguments(

      output_dir="mistral-finetuned-alpaca",

      per_device_train_batch_size=8,

      gradient_accumulation_steps=1,
```

```
        optim="paged_adamw_32bit",
        learning_rate=2e-4,
        lr_scheduler_type="cosine",
        save_strategy="epoch",
        logging_steps=100,
        num_train_epochs=1,
        max_steps=250,
        fp16=True,
        push_to_hub=True
    )
    trainer = SFTTrainer(
        model=model,
        train_dataset=data,
        peft_config=peft_config,
        dataset_text_field="text",
        args=training_arguments,
        tokenizer=tokenizer,
        packing=False,
        max_seq_length=512
    )
```

4.5. Chatbot Development:

Integrating the finetuned model into a simple chatbot application using a web framework such as Flask or Streamlit . I had used flask

```
from peft import AutoPeftModelForCausalLM
from transformers import GenerationConfig
from transformers import AutoTokenizer
import torch
tokenizer = AutoTokenizer.from_pretrained("/content/mistral-finetuned-alpaca")
inputs = tokenizer("""###Human: Why mobile is bad for human? ###Assistant: """,
return_tensors="pt").to("cuda")
model = AutoPeftModelForCausalLM.from_pretrained(
   "/content/mistral-finetuned-alpaca",
   low_cpu_mem_usage=True,
   return_dict=True,
   torch_dtype=torch.float16,
   device_map="cuda")
generation_config = GenerationConfig(
   do_sample=True,
   top_k=1,
   temperature=0.1,
   max_new_tokens=100,
   pad_token_id=tokenizer.eos_token_id
)
import time
st_time = time.time()
outputs = model.generate(**inputs, generation_config=generation_config)
```

4.6. Testing and Evaluation:

Various test cases are executed to evaluate the chatbot's performance, including common queries, edge cases, and scenarios requiring contextual understanding.

5. Results

The chatbot developed using the Mistral 7B model demonstrates effective handling of user queries with contextually appropriate responses. The finetuning process significantly enhances the model's performance, making it suitable for real-world applications.

6. Conclusion

This project highlights the potential of GenAI and LLMs in creating sophisticated chatbots. By leveraging the Mistral 7B model and the Hugging Face dataset, we successfully developed a custom chatbot that can interact with users effectively. The project underscores the importance of finetuning in enhancing LLM performance for specific applications.

TEAM MEMBERS

1. Amritpal Singh (team lead)
2. Bhavsagar
3. Abhishek menka
4. Yashika gupta