Deep Neural Model for Detecting Gender Stereotypes in Text

Bachelor Term Project

by Adarsh Sharma(20ME3AI10)

Content

- About Dataset
- Different ways of fine tuning LLMs
- QLoRA
- Gemma
- Configuration
- Results
- Future work

About Dataset

We originally had the dataset with 120 gender stereotype sentences and about 10,000 non stereotype words.

As dataset is highly biased towards stereotype words, So, We have taken about only 250 sentences out of all non-stereotype sentences in the dataset

We have mixed the dataset and divided it into 5 parts with each part containing 24 gender stereotypical sentences and 50 non gender stereotypical sentences.

Some examples of gender stereotypical words in our dataset

"It is the duty of a housewife to express kindness towards the servants by measuring their physical ability and psychological attitude."

"A housewife has to observe whether her children are growing up mentally as well so that they can cope with the changing world."

Some examples of non-gender stereotypical words in our dataset

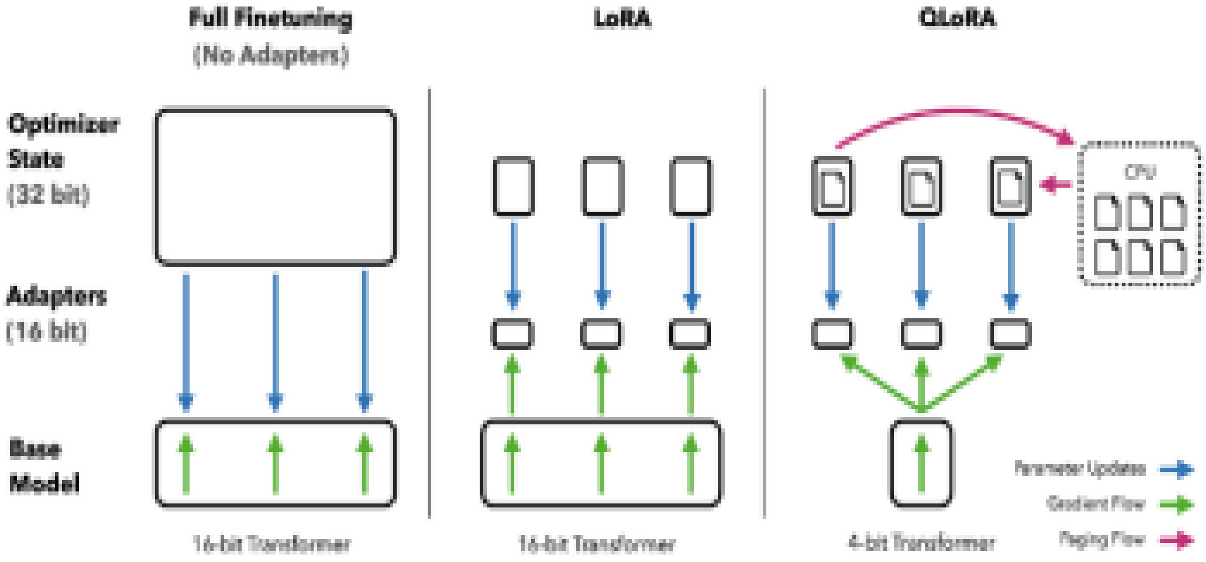
“Answer the following questions briefly.”

“What is the criteria to decide a best fuel?”

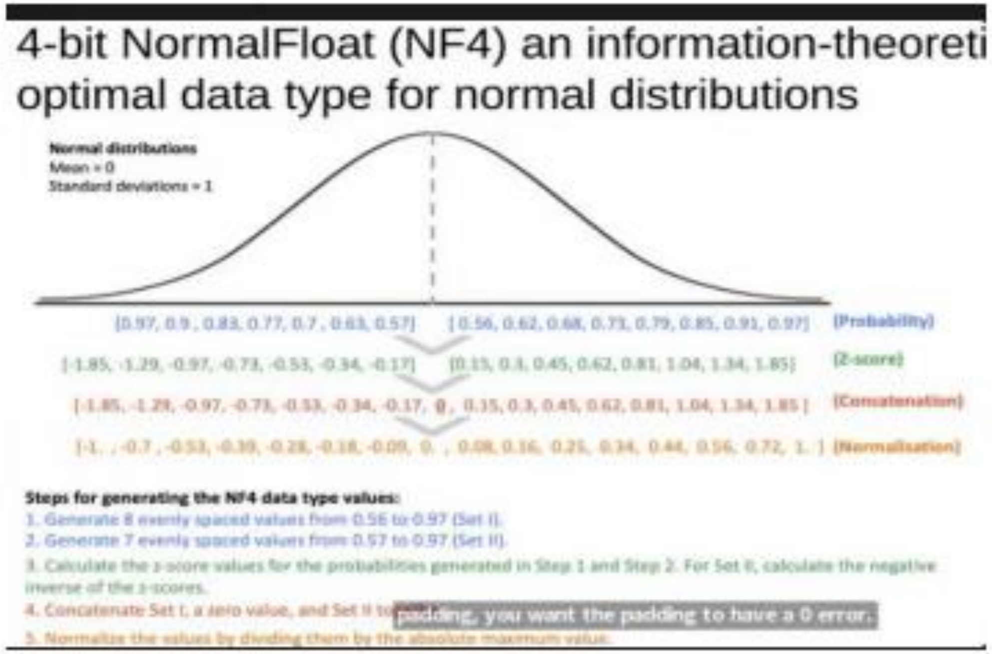
Different ways to supervised fine-tune a LLM

QLoRA

- 4bit Normalfloat type quantization
- Double quantization
- Paged optimizer



4bit normal floating point representation



Low rank parmater 'A' and 'B' in LoRA

$$W_0 + \Delta W = W_0 + BA, \text{ where } B \in \mathbb{R}^{d\times r}, A \in \mathbb{R}^{r\times k},$$

$$W_0 \in \mathbb{R}^{d\times k},$$

- Gemma is a decoder only model
- While the Gemma 7B model leverages a multihead attention mechanism, Gemma 2B utilizes multiquery attention. This approach aids in reducing memory bandwidth requirements during the inference

1. Prompt given while training:

```
generate_prompt
          Analyze the stereotype of the sentence enclosed in square brackets,
          determine if it is positive, or negative, and return the answer as
          the corresponding stereotype label "positive" or "negative"

          [Rivers , like the Colorado River , carry enormous loads of sand and soil that is picked up from erosional processes
.] = Negative<eos>
```

2. Prompt given while testing

```
Analyze the stereotype of the sentence enclosed in square brackets,
          determine if it is positive,or negative, and return the answer as
          the corresponding stereotype label "positive" or "negative"

          [Turbine Fig . Each turbine is made of curved blades arranged like the sails of a windmill .] =
```

```
compute_dtype = getattr(torch, "float16")

bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=False,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=compute_dtype,
)
```

- Configurations

▶Configurations:

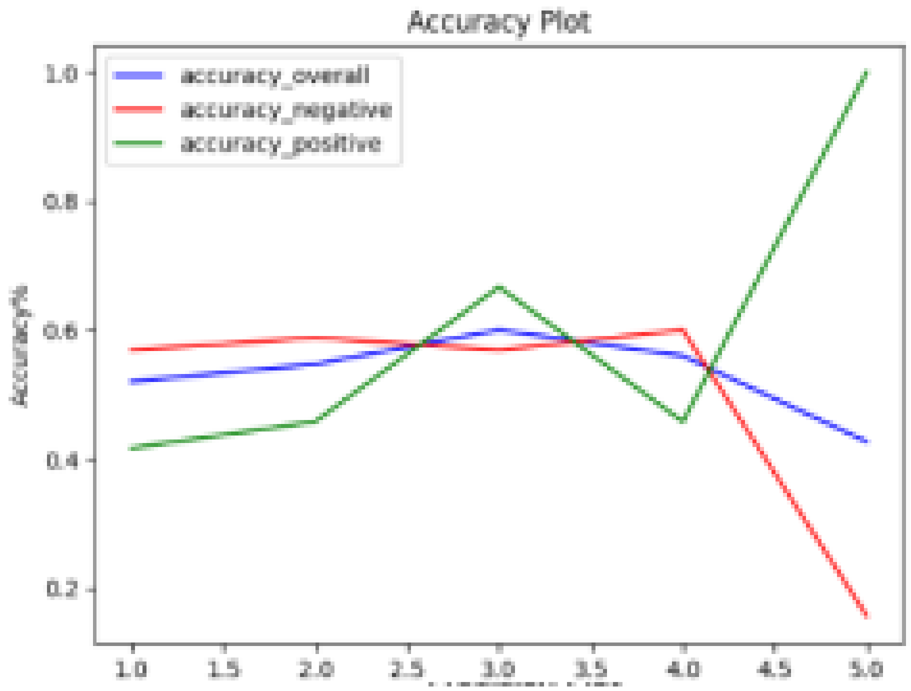
```
trainer = SFTTrainer(
    model=model,
    train_dataset=train_data,
    peft_config=peft_config,
    dataset_text_field="text",
    tokenizer=tokenizer,
    max_seq_length=max_seq_length,
    args=training_arguments,
    packing=False,
)
```

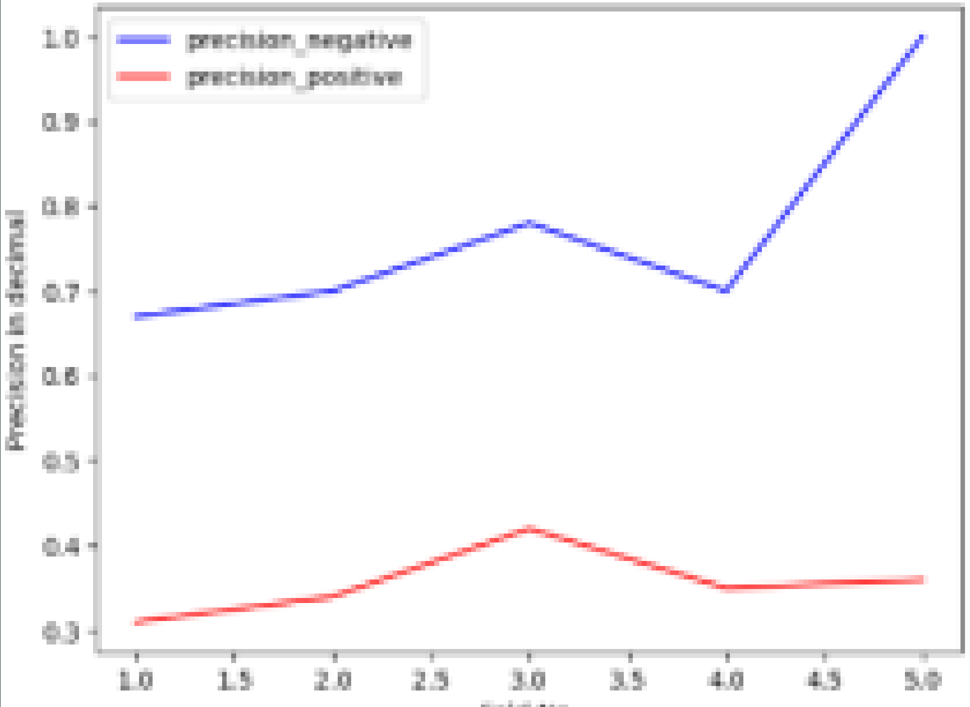
```
training_arguments = TrainingArguments(
    output_dir="logs",
    num_train_epochs=10,
    gradient_checkpointing=True,
    per_device_train_batch_size=1,
    gradient_accumulation_steps=8,
    optim="paged_adamw_32bit",
    save_steps=0,
    logging_steps=25,
    learning_rate=2e-4,
    weight_decay=0.001,
    fp16=True,
    bf16=False,
    max_grad_norm=0.3,
    max_steps=-1,
    warmup_ratio=0.03,
    group_by_length=False,
    evaluation_strategy="no",                  # no evaluation done
    lr_scheduler_type="cosine",
    report_to="tensorboard",
)
```

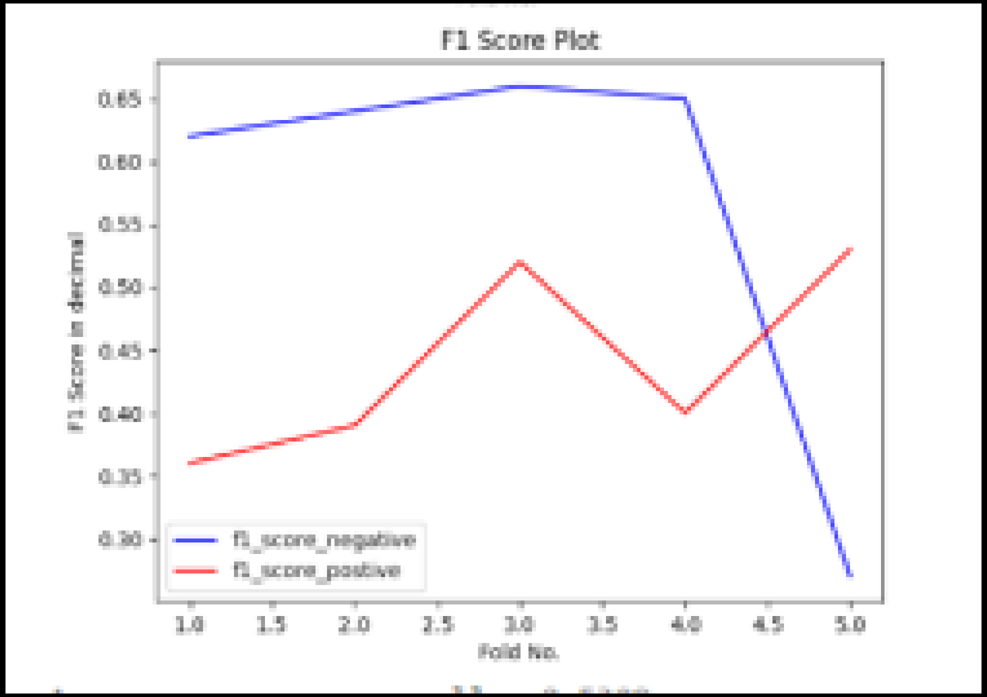
```
peft_config = LoraConfig(
    lora_alpha=16,
    lora_dropout=0,
    r=64,
    bias="none",
    task_type="CAUSAL_LM",
    target_modules=["q_proj", "k_proj", "v_proj", "o_proj",
                    "gate_proj", "up_proj", "down_proj",],
)
```

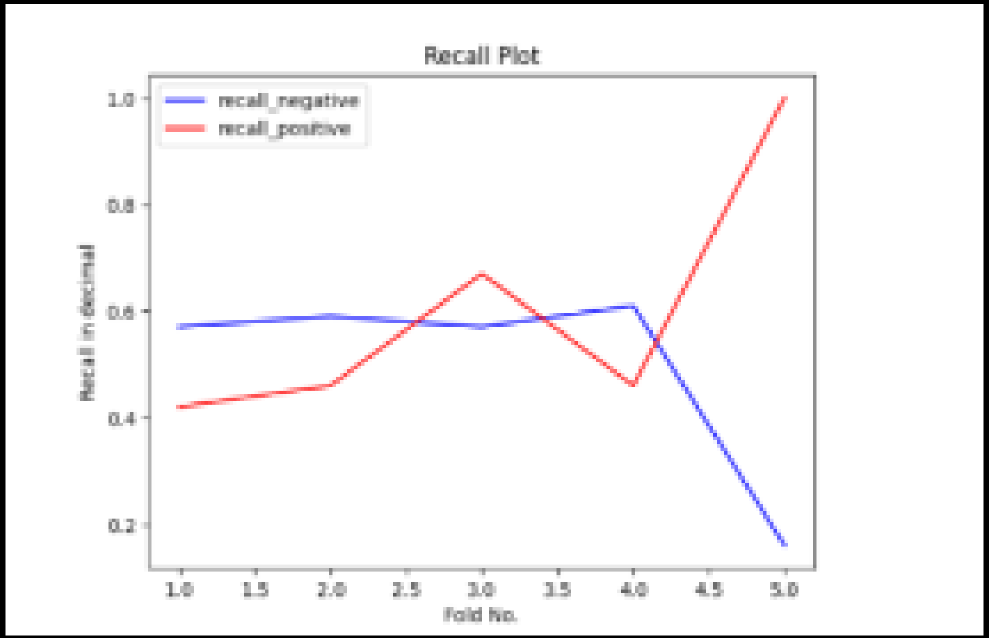
▶Unfinetuned model direct inferencing using prompt tuning performance.

```
Average accuracy_overall:  0.5308
Average accuracy_negative:  0.49660000000000004
Average accuracy_positive:  0.6
Average precision_negative:  0.7700000000000001
Average precision_positive:  0.356
Average recall_negative:  0.5
Average recall_positive:  0.6020000000000001
Average f1_score_negative:  0.568
Average f1_score_postive:  0.44000000000000006
```



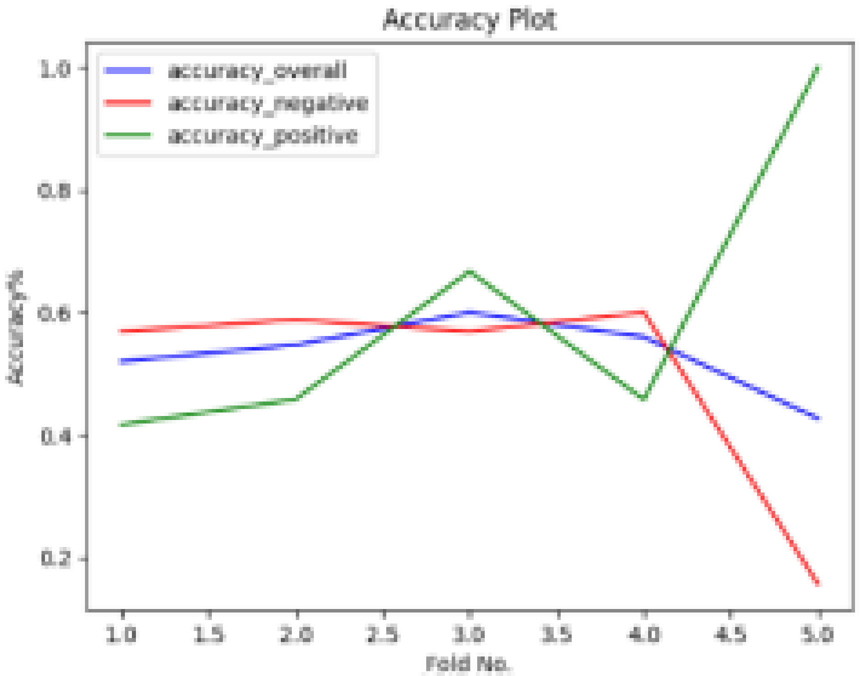


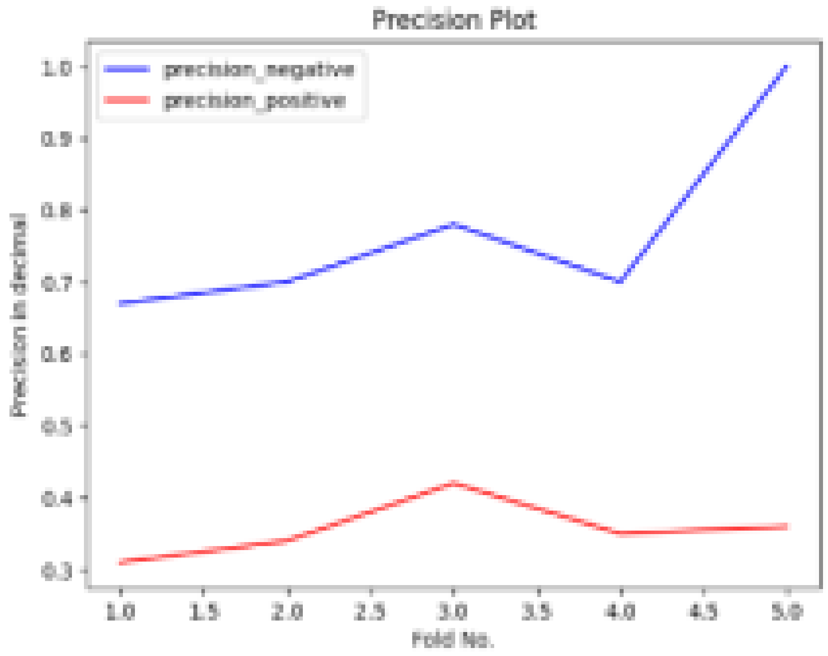

F1 Score Plot
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.30
F1 Score in decimal
f1_score_negative
f1_score_postive
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Fold No.

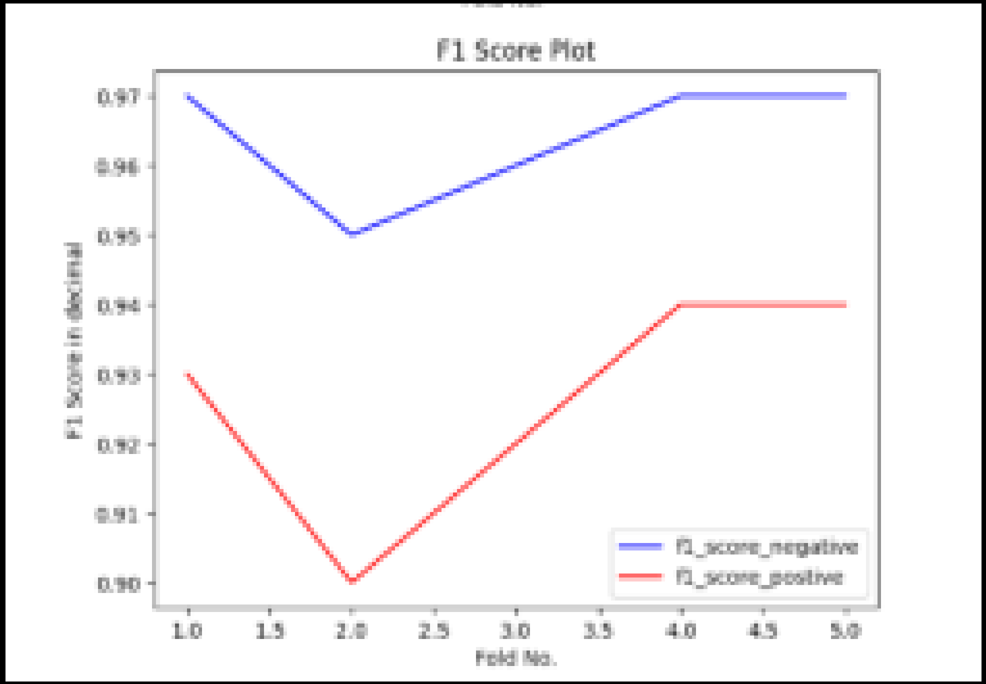
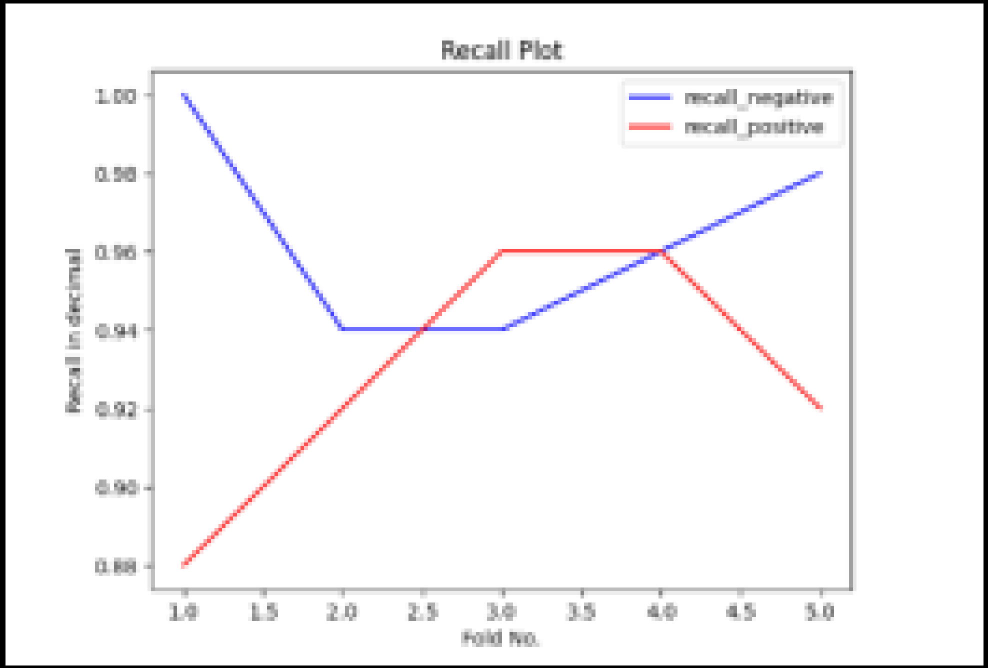

Recall Plot
recall_negative
recall_positive
1.0
0.8
0.6
0.4
0.2
Recall in decimal
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Fold No.

▶Fine tuned model performance

```
Average accuracy_overall:  0.952
Average accuracy_negative:  0.9645999999999999
Average accuracy_positive:  0.925
Average precision_negative:  0.9640000000000001
Average precision_positive:  0.9279999999999999
Average recall_negative:  0.9640000000000001
Average recall_positive:  0.9279999999999999
Average f1_score_negative:  0.9639999999999999
Average f1_score_postive:  0.9259999999999999
```






F1 Score Plot
0.97
0.96
0.95
0.94
0.93
0.92
0.91
0.90
F1 Score in decimal
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Fold No.
f1_score_negative
f1_score_positive

Recall Plot
1.00
0.98
0.96
0.94
0.92
0.90
0.88
Recall in decimal
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Fold No.
recall_negative
recall_positive

Future Work

In the future, I will try to explore other parameter efficient fine-tuning ways like prefix tuning and check model performance on it In future, I would try explore other parameter efficient fine-tuning ways like prefix tuning and check model performance on it.

Explore topics like reinforcement learning with human feedback for improving the accuracy

Thank you