Akshay Syal

5. Write a detailed report covering the model choice, dataset preparation, fine-tuning process, and an analysis of the summarization performance before and after fine-tuning.

Model choice: Selected the lowest performing model google-t5/t5-small with the hope of improving its performance.

Dataset preparation:

1. Prefixed the input with prompt ‘summarize’ so T5 knows this is a summarization task. Some models capable of multiple NLP tasks require prompting for specific tasks.
2. Used the keyword text_target argument when tokenizing labels.
3. Truncated sequences to be no longer than the maximum length set by the max_length parameter.
4. Used dataset’s map method to apply the preprocessing function over the entire dataset. Speeded up the map function by setting batched=True to process multiple elements of the dataset at once.
5. Created a batch of examples using DataCollatorForSeq2Seq. It’s more efficient to dynamically pad the sentences to the longest length in a batch during collation, instead of padding the whole dataset to the maximum length.

Fine-tuning process:

Fine-Tuned the model on the dataset for 3 epochs. Saved the model checkpoint every 100 steps in google drive. Kept batch_size as 1 and fp16 to true avoid GPU overflow issues.

```
batch_size = 1
num_train_epochs = 3

args = TrainingArguments(
    output_dir="/content/drive/MyDrive/LLMPA3-4",
    evaluation_strategy="no",  # Disable evaluation during training
    learning_rate=1e-5,
    per_device_train_batch_size=batch_size,
    per_device_eval_batch_size=batch_size,
    weight_decay=0.01,
    save_total_limit=3,
    num_train_epochs=num_train_epochs,
    gradient_accumulation_steps=16,
    save_steps=100,
    fp16=True,
    eval_steps=None,  # Ensure no evaluation is performed at any specific steps
    # push_to_hub=True,
```

```
    # save_strategy='epoch'
)

trainer = Trainer(
    model=model,
    args=args,
    train_dataset=tokenized_datasets["train"],
    eval_dataset=None, # no evaluation
    data_collator=data_collator,
    tokenizer=tokenizer,
    compute_metrics=compute_metrics,
)
```

Analysis of the summarization performance before and after fine-tuning:

Fine tuning the model improved the model's ability to summarize the text for both train and test data. The original model was not good at understanding the underlying semantics. Fine tuning the model didn't help in that aspect.

However, the fine-tuned model's summary is generally more coherent, at least able to summarize the content and not include the dialog itself. Also the number of words produced is generally lesser than that of the original model.

After Fine tuning the Rouge Scores improved on the test set.

Before Fine Tuning:
{'rouge1': 0.2839869931922432,
'rouge2': 0.07813235417373512,
'rougeL': 0.2146845115342596,
'rougeLsum': 0.2149136580655594}

After Fine Tuning:
{'rouge1': 0.35386344587365093,
 'rouge2': 0.13875659543532526,
 'rougeL': 0.29093864767787203,
 'rougeLsum': 0.2911483466753183}

Summaries of Train Dataset sample

Sr	google_t5_small_summary	fine_tuned_google_t5_small_summary
1	Missed essential points	Missed essential points

2	Covered some points of conversation	Missed or misunderstood
3	Missed essential points	Missed essential points
4	Missed essential points	Covered some points of conversation
5	Covered some points of conversation	Covered more points of conversation

Summaries of Test dataset sample:

Sr	google_t5_small_summary	fine_tuned_google_t5_small_summary
1	Missed essential points	Missed essential points
2	Covered few points	Covered few points
3	Covered few points	Missed essential points
4	Covered few points	Covered few points
5	Covered few points	Covered few points

References:

https://huggingface.co/docs/transformers/en/tasks/summarization#load-billsum-dataset