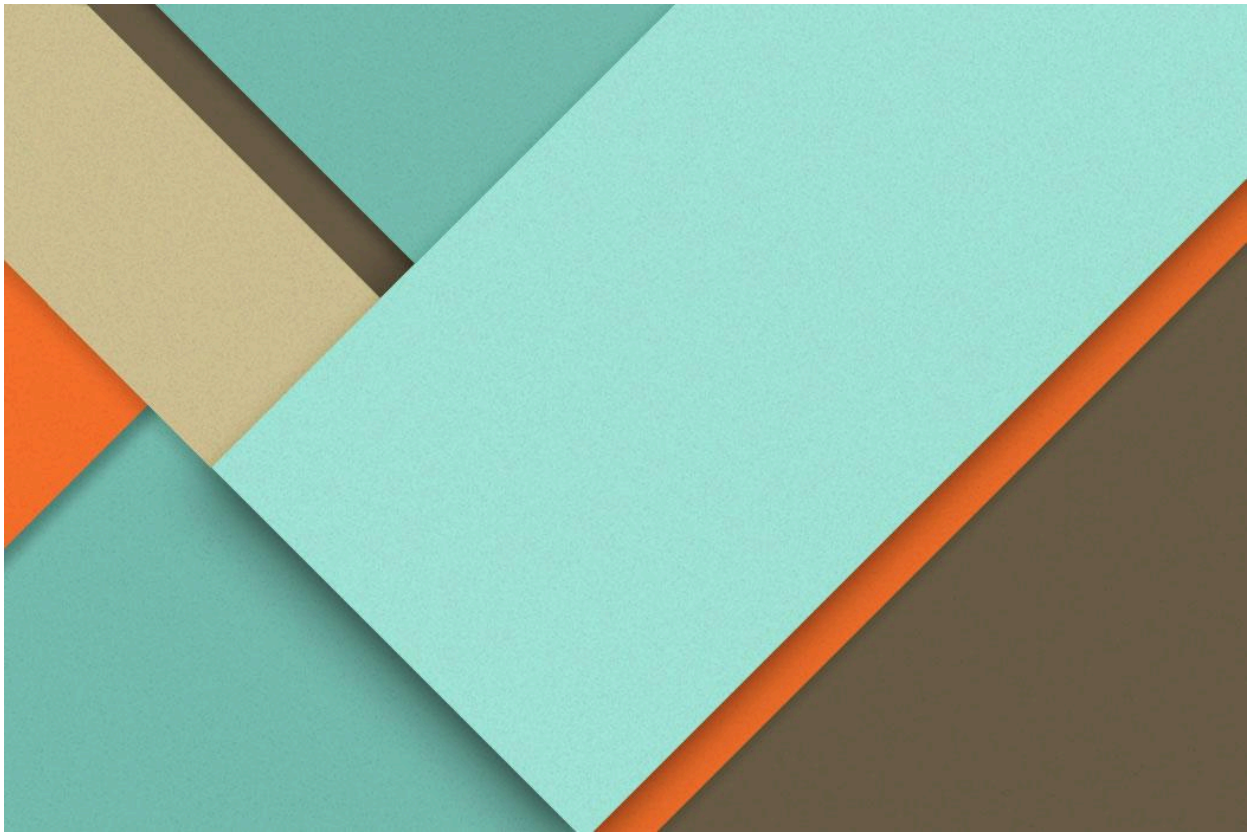

Project Report AIML based QnA Generation - Group 18

28.09.2024

Rohini Palanisamy Sengottaian

Prashant Kataria

Gudimella Sridhar

Anupreksha Jain

Overview **2**
Email Subject Line Generation **3**
Dataset 3
Models Fine-Tuned 3
Training Details 3
1. Mistral 3
Training Arguments: 4
2. Llama3 4
Training Arguments: 4
3. T5 5
Training Arguments: 5
4. Bart 6
Training Arguments: 6
Inference Results 7
Example of Email Body and Subject Line outputs: 7
Reference Outputs: 7
Model Evaluation Criteria 8
HuggingFace Demo & API 8
Code Files: 8
Observations 9
AIML Question and Answer Generation **10**
Dataset 10
Models Used and Training Details 10
Bart Model 10
GPT-2 Model 11
Llama3.1 12
Mistral7b 13
Qwen 13
Evaluation and Results 14
Observations: 15
Inference Example: 15
Key Learnings 15
API and Deployment 15
API Usage Example: 16
Conclusion 16

Overview

This project aims to explore and learn the generative text systems through two key tasks. In the first task, team will fine-tune a GPT models on a pre-prepared dataset, focusing on subject line generation. The second task involves deploying a trained Question and Answer model to test its ability to answer new AIML queries.

The Email Subject Line Generation task focuses on creating concise and meaningful subject lines from the body of an email. This project involves working with generative models in Natural Language Processing (NLP), particularly GPT-2 variants, and exploring various metrics for evaluating text generation. The goal is to identify key information within emails and abstract it into a brief subject line, providing users with an accurate summary of the email content.

The Question and Answering task involves modeling a domain-specific GPT-variant model tailored to answer questions related to the Artificial Intelligence and Machine Learning (AIML) course. The goal is to fine-tune models on a specialized and manually curate datasets to improve the model's ability to answer AIML questions with greater precision and accuracy.

For more detailed project insights, the capstone project proposal is documented in the following file:

- [Capstone Project Proposal - Google Docs.pdf](Group 18)

Email Subject Line Generation

The goal here is to use the annotated enron subject line corpus and train it on various models to generate concise and contextually correct subject line for the email. Four different LLM models were trained and tested on the data the section describes the process and results.

Dataset

The project used **The Annotated Enron Subject Line Corpus** from the repository below to fine-tune models:

- Dataset Repository: AESLC

Models Fine-Tuned

Multiple models were fine-tuned as part of the project, with varying architectures and training steps. Each model was evaluated using the **ROUGE score**, which measures the overlap between the generated and reference subjects.

Model	Framework	Model Type	Training Steps	Evaluation Method
Mistral	unsloth	4-bit quantized	60	ROUGE Score
Llama3	unsloth	4-bit quantized	60	ROUGE Score
T5	Transformer	Base model	200	ROUGE Score
Bart	Transformer	Base model	200	ROUGE Score

Training Details

1. Mistral

- **Code File**: `Group18EmailDataSetTrainingMistral.ipynb`

- **Model**: `unsloth/mistral-7b-v0.3-bnb-4bit`
- **Training Framework**: Huggingface `trl SFTrainer`
- **Training Steps**: 60

Training Arguments:

```
TrainingArguments(
    per_device_train_batch_size=2,
    per_device_eval_batch_size=2,
    gradient_accumulation_steps=4,
    evaluation_strategy="steps",
    warmup_steps=5,
    num_train_epochs=3,
    max_steps=60,
    learning_rate=2e-4,
    logging_steps=1,
    optim="adamw_8bit",
    weight_decay=0.01,
    seed=3407,
    output_dir="outputs"
)
```

2. Llama3

- **Code File**: `Group18FineTuneLlama3EmailSubjectFinal.ipynb`
- **Model**: `unsloth/llama-3-8b-bnb-4bit`
- **Training Steps**: 60

Training Arguments:

```
TrainingArguments(
    per_device_train_batch_size=2,
    gradient_accumulation_steps=4,
    warmup_steps=5,
    max_steps=60,
    learning_rate=2e-4,
    logging_steps=1,
    optim="adamw_8bit",
    weight_decay=0.01,
```

```
    seed=3407,
    output_dir="outputs"
)
```

3. T5

- **Code File**: `Group18FineTuningT5EmailSubject.ipynb`
- **Model**: `t5-base`
- **Training Framework**: Transformer `Seq2SeqTrainer`

Training Arguments:

```
Seq2SeqTrainingArguments(
    evaluation_strategy="steps",
    eval_steps=200,
    logging_strategy="steps",
    logging_steps=100,
    save_strategy="steps",
    save_steps=200,
    learning_rate=4e-5,
    per_device_train_batch_size=8,
    weight_decay=0.01,
    num_train_epochs=2,
    predict_with_generate=True,
    fp16=True,
    load_best_model_at_end=True,
    metric_for_best_model="rouge1",
    report_to="tensorboard"
)
```

4. Bart

- **Code File**: `Group18FineTuneBartEmailSubjectFinal.ipynb`
- **Model**: `facebook/bart-large-xsum`
- **Training Framework**: Transformer `Seq2SeqTrainer`

Training Arguments:

```
Seq2SeqTrainingArguments(
    evaluation_strategy="steps",
    eval_steps=200,
    logging_strategy="steps",
    logging_steps=100,
    save_strategy="steps",
    save_steps=200,
    learning_rate=4e-5,
    per_device_train_batch_size=8,
    weight_decay=0.01,
    num_train_epochs=2,
    predict_with_generate=True,
    fp16=True,
    load_best_model_at_end=True,
    metric_for_best_model="rouge1",
    report_to="tensorboard"
)
```

Inference Results

Examples of email bodies, their corresponding reference outputs, and the generated subject lines from each model are provided below.

Example of Email Body and Subject Line outputs:

Please help summarize the provided email body and generate email subject*:*

"Kevin Presto is requesting that you attend a meeting regarding Organizing an Action Plan for the Start-up of Netco. The meeting will be held in ECS 06716 at 9:30 am, Wednesday, January 2, 2002.

For Tim and Chris, could you please call 713-584-2067. This is the telephone number in the conference room.

If you should have any questions, please call T Jae Black at 3-5800.

Thanks"

Reference Outputs:

- **Llama3 Output**: Netco Action Plan Meeting
- **Mistral Output**: Meeting on Netco Startup
- **T5 Output**: Organizing an Action Plan for the Start
- **Bart Output**: Organizing an Action Plan

Model Evaluation Criteria

The **ROUGE score** was used to evaluate the models. It compares the generated subject line to the reference, with higher values indicating better overlap.

Model	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-LSum
Mistral	0.04175	0.01531	0.03866	0.04011
Llama3	0.04454	0.01628	0.03984	0.04157
T5	0.14457	0.07031	0.14026	0.14112
Bart	0.26737	0.13460	0.24999	0.25001

Model Deployment

HuggingFace Demo & API

- **Gradio App**: Gradio App Demo
- **Fast API**: Fast API Demo

Code Files:

- **Gradio App Notebook**: `GradioAppWithModelSelection.ipynb`
- **Fast API Code**: Stored in the `api/` folder, including Dockerfile, requirements.txt, and main.py

Observations

1. **Quantized Models**: Due to resource limitations, quantized versions of generative models were used for training.
2. **Generative vs. Seq2Seq Models**:
    - Generative models often generate synonyms of important words, leading to a lower ROUGE score.
    - Seq2Seq models like Bart and T5 capture the exact words from the email content, leading to higher ROUGE scores.
3. **Bart vs. T5**: While both models are encoder-decoder based, Bart outperformed T5 due to differences in their training corpus, token dropout strategies, and parameter initialization.

AIML Question and Answer Generation

The project involves modeling a domain-specific GPT-variant model tailored to answer questions related to the Artificial Intelligence and Machine Learning (AIML) course. The challenge lies in the fact that pretrained models, although proficient in generating general textual outputs, often fall short in domain-specific contexts. The goal was to fine-tune models on a specialized dataset to improve their ability to answer AIML questions with greater precision and accuracy.

Dataset

A novel dataset specific to the AIML domain was used for fine-tuning. The dataset can be accessed at the following repository:

- Dataset: Google Drive

Data Collection and Preprocessing:

- Compiled a dataset of AIML-related questions and answers from the AIML course materials.
- ~3600 AIML Questions and Answers were curated by multiple teams
- Ensure the dataset covers all units within AIML course.
- Preprocessed the data to remove punctuation marks, lower case the text for consistency to ensure quality and consistency.

Models Used and Training Details

Bart Model

- **Model Type:** `facebook/bart-large-xsum`
- **Framework:** Transformer (Seq2SeqTrainer)
- **Training Steps:** 600
- **Training Parameters:**

```
Seq2SeqTrainingArguments(
    model_dir,
    evaluation_strategy="steps",
```

```
    eval_steps=100,
    logging_strategy="steps",
    logging_steps=100,
    save_strategy="steps",
    save_steps=100,
    learning_rate=4e-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=6,
    predict_with_generate=True,
    fp16=True,
    load_best_model_at_end=True,
    metric_for_best_model="rouge1",
    report_to="tensorboard"
)
```

- **Results:**
  - **Best Performance:** Step 500
  - **ROUGE-1:** 36.23
  - **ROUGE-2:** 16.21
  - **ROUGE-L:** 29.72

GPT-2 Model

- **Model Type:** `gpt2` (LoRa)
- **Framework:** Transformer (LoRa-based adaptation)
- **Training Steps:** 600
- **Training Parameters:**

```
TrainingArguments(
    output_dir="./gpt3-lora-qa",
    overwrite_output_dir=True,
    evaluation_strategy="steps",
    eval_steps=100,
    logging_strategy="steps",
    logging_steps=100,
    num_train_epochs=5,
    per_device_train_batch_size=2,  # Lower batch size
    per_device_eval_batch_size=2,
    gradient_accumulation_steps=4,  # Adjust batch size based on GPU memory
```

```
    save_steps=500,
    save_total_limit=2,
    fp16=True,  # Use mixed precision training for efficiency
    report_to="none",
    dataloader_pin_memory=True
)
```

- **Results:**
    - **Best Performance:** Step 500
    - **ROUGE-1:** 48.23
    - **ROUGE-2:** 17.87
    - **ROUGE-L:** 39.41

Llama3.1

- **Model Type:** `Llama3.1` (4-bit Quantized)
- **Framework:** Unsloth
- **Training Steps:** 60
- **Training Parameters:**

```
TrainingArguments(
        per_device_train_batch_size = 2,
        gradient_accumulation_steps = 4,
        warmup_steps = 5,
        max_steps = 60,
        learning_rate = 2e-4,
        fp16 = not is_bfloat16_supported(),
        bf16 = is_bfloat16_supported(),
        logging_steps = 1,
        optim = "adamw_8bit",
        weight_decay = 0.01,
        lr_scheduler_type = "linear",
        seed = 3407,
        output_dir = "outputs",
    )
```

- **Results:**
    - **Best Performance:** Step 60
    - **ROUGE-1:** 0.20
    - **ROUGE-2:** 0.10
    - **ROUGE-L:** 0.22

Mistral7b

- **Model Type:** `Mistral7b` (4-bit Quantized)
- **Framework:** Unsloth
- **Training Steps:** 60
- **Training Parameters:**

```
TrainingArguments(
        per_device_train_batch_size = 2,
        gradient_accumulation_steps = 4,
        warmup_steps = 5,
        max_steps = 60,
        learning_rate = 2e-4,
        fp16 = not is_bfloat16_supported(),
        bf16 = is_bfloat16_supported(),
        logging_steps = 1,
        optim = "adamw_8bit",
        weight_decay = 0.01,
        lr_scheduler_type = "linear",
        seed = 3407,
        output_dir = "outputs",
    )
```

- **Results:**
    - **Best Performance:** Step 60
    - **ROUGE-1:** 0.26
    - **ROUGE-2:** 0.08
    - **ROUGE-L:** 0.21

Qwen

- **Model Type:** `Qwen1.5-0.5B` (Quantized model)
- **Framework:** NA - Using Qwen2CausalML
- **Training Steps:** 100
- **Results:**
    - **Best Performance:** Step 100
    - **ROUGE-1:** 0.60
    - **ROUGE-2:** 0.31
    - **ROUGE-L:** 0.53
- **Code File**: `Group18-AIMLQnA-Qwen_HuggingFace_Sep26.ipynb`
- **Model**: `Qwen1.5-0.5B`
- **Evaluation steps**: 100

- **Training Arguments:**

```
TrainingArguments(
    eval_strategy="steps",
    eval_steps=100,
    logging_strategy="steps",
    logging_steps=100,
    num_train_epochs=10,
    per_device_train_batch_size=4,
    per_device_eval_batch_size=2,
    gradient_accumulation_steps=2,
    save_steps=500,
    save_total_limit=2,
    gradient_checkpointing=True,
    fp16=True,
    report_to="none",
    dataloader_pin_memory=True
)
```

Evaluation and Results

The primary evaluation criterion for the fine-tuned models was the ROUGE score, which measures the similarity between the model's generated responses and the ground truth answers.

Model	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-LSum
Bart	0.38	0.17	0.31	0.40
GPT-2	0.48	0.17	0.39	0.42
Llama3.1	0.20	0.10	0.22	0.23
Mistral7b	0.26	0.08	0.21	0.20
Qwen1.5-0.5B	0.58	0.28	0.51	0.54

Observations:

- **GPT-2** achieved the best performance, with a **ROUGE-1 score of 48.23**, making it the most effective model for answering AIML-related questions.
- **Bart** followed closely, with a decent performance, while **Llama3.1** and **Mistral7b** did not perform as well, likely due to their smaller training size and quantization.

Inference Example:

Question: "What is a linear classifier?"

- **Bart:** "A linear classifier is a model used to classify data points along a line."
- **Qwen: "**A linear classifier is a type of classifier that predicts the class of a given input based on the input's features."
- **GPT2:** Linear classifiers are a way to represent a set of values in a linear way. They are used to represent a set of values in a linear way.
- **Mistral:** A linear classifier is a machine learning algorithm that predicts class labels by assigning weights to features. It uses the weighted sum of feature values as input for classification
- **Llama 3.1:** A linear classifier is a machine learning algorithm that predicts class labels by assigning weights to features. It uses the weighted sum of feature values as input for classification.', 'Linear classifiers are models used in machine learning to classify data points into different classes based on their feature values.

Key Learnings

- The project revealed the challenges of fine-tuning large generative models for domain-specific tasks. Due to the resource limitations, quantized versions of models were employed to make the training feasible.
- PEFT (Parameter Efficient Fine-Tuning) techniques like LoRa (Low-Rank Adaptation) proved highly effective in enhancing GPT-2's performance.

API and Deployment

The fine-tuned models were made accessible through a Hugging Face Gradio app, allowing users to ask AIML-related questions and receive model-generated responses. The code for deploying the models via FastAPI is available in the `api` folder, and a Dockerfile was included for easy deployment.

- **Gradio App**: anukvma/Question_Answers
- **FastAPI**: Deployed via Hugging Face Space

API Usage Example:

```
curl --location --request GET 'https://anukvma-aimlqnaapi.hf.space' \
--header 'Content-Type: application/json' \
--data-raw '{
    "question": "what is linear regression?"
}'
```

Response: "Linear regression is a statistical method used to forecast the probability of a dependent variable using a linear equation."

Conclusion

Both projects leveraged fine-tuning of large language models using specific datasets for tasks like email subject generation and AIML question answering. The quantized versions of models (like Mistral and Llama) were necessary due to hardware limitations, and models like Bart and GPT-2 emerged as the best performers in their respective tasks. Both projects demonstrate the potential of large models fine-tuned on domain-specific tasks, although computational efficiency remains a challenge.