

Homework 2 Report

1. Introduction

1.1 Purpose of this report

The purpose of this report is to explore open source Large Language Models by downloading and evaluating the Llama 3.2 model. Additionally, this project involves fine tuning Llama 3.2 using LoRA (Low Rank Adaptation), a parameter efficient training method, to improve performance on a selected dataset. The selected dataset and further details regarding the experiment will be detailed in later sections of this report.

1.2 Overview of OpenSource LLMs and LoRA Fine Tuning Basics

Open source LLMs are widely available and allow customization for specific tasks such as text summarization, classification, and reasoning. We find this to be particularly important in an age where proprietary software is leading and deemed most profitable. LoRA fine tuning optimizes these models by introducing low rank weights, reducing the overall computational cost while retaining the model's performance. This method is especially useful for efficiently adapting models to domain specific data without retraining the entire system.

1.3 Structure of the Report

This report is structured into two main sections. The first section discusses the Llama 3.2 model, including its architecture, download process, and evaluation on different tasks. The second section covers fine tuning Llama 3.2 using LoRA, detailing dataset selection, implementation, hyperparameters, and performance evaluation.

2. Problem 1: Understanding and Running Llama 3.2

2.1 Model Architecture

The Llama 3.2 3B model is a transformer based LLM optimized for efficiency and performance, and it differs from GPT3 in several ways. While GPT-3 consists of 175 billion parameters, Llama 3.2 3B is significantly smaller, making it more resource efficient while technically less powerful, meaning it is able to run on much smaller systems like our devices. Additionally, Llama 3.2 incorporates architectural improvements to enhance text generation and reasoning tasks, making it a viable alternative to GPT based models overall.

CSC 5991

Ammar Ahmed - HL5101, Rensildi, Taaseen - HL7392

Homework 2 3/14/25

2.2 Model Download and Setup

The Llama 3.2 1B & 3B models were downloaded from Meta's official website, Kaggle, and Ollama. The setup process involved installing Python 3.9+, PyTorch, the Transformers library, and Hugging Face's datasets module. Once installed, the model was verified by running test inputs to check for proper inference execution. The output of the installation confirmed that the model was successfully loaded and ready for evaluation. This can all be used to recreate the results of this report if need be.

2.3 Evaluating Pretrained Llama 3.2 3B

To assess the performance of Llama 3.2, we tested it on five different NLP tasks: text summarization, question answering, text classification, role playing, and reasoning. The model performed well on summarization and classification tasks, generating coherent and contextually relevant outputs. For question answering, the model accurately responded to factual queries but sometimes lacked nuanced details. In role playing scenarios, it successfully generated responses in different character personas, though some responses were repetitive. In reasoning tasks, the model demonstrated logical inference abilities but struggled with more complex multi step reasoning problems. Observations showed that while the model had strong general capabilities, fine tuning was necessary to improve performance on domain specific datasets.

Below are the asked for outputs of the requested tasks.

```
C:\Users\ekalaz\ollama run llama3.2
>>> Antibiotics are a type of medication used to treat bacterial infections. They work by either killing the bacteria or p
... preventing them from reproducing, allowing the body's immune system to fight off the infection. Antibiotics are usuall
... y taken orally in the form of pills, capsules, or liquid solutions, or sometimes administered intravenously. They ar
... e not effective against viral infections, and using them inappropriately can lead to antibiotic resistance. Explain
... the above in one sentence.
Antibiotics are a type of medication used to treat bacterial infections by either killing or preventing
bacterial reproduction, allowing the body's immune system to fight off the infection, but they are not
effective against viral infections and should be used responsibly to avoid promoting antibiotic resistance.

>>> Answer the question based on the context below. Keep the answer short and concise. Respond "Unsure about answer" if not sure about the answer. Context: Tepilizumab traces its roots to a New Jersey drug company called Ortho Pharmaceuti
... cal. There, scientists generated an early version of the antibody, dubbed OKT3. Originally sourced from mice, the molecule was able to bind to the surface of T cells and limit their cell-killing potential. In 1986, it was approved to
... help prevent organ rejection after kidney transplants, making it the first therapeutic antibody allowed for human use. Question: What was OKT3 originally sourced from?
Mice.

>>> Classify the text into neutral, negative or positive. Text: I think the food was okay.
Neutral.

>>> The following is a conversation with an AI research assistant. The assistant tone is technical and scientific. Hu... man: Hello, who are you?
Greetings human entity. I am an artificial intelligence research assistant, a cognitive model designed to
process and generate human-like linguistic output based on the inputs provided to me. My architecture is
grounded in machine learning principles, utilizing advanced neural network algorithms to analyze and respond to
complex queries. I am currently operating within the realm of knowledge sharing and information dissemination,
serving as a resource for users seeking accurate and concise answers to their inquiries.

>>> Can you tell me about the creation of blackholes?
The formation of black holes is a complex process that involves the collapse of massive stars or other compact
objects with extremely high densities.

According to our current understanding, black holes are formed when a star with a mass at least 3-4 times that
of the sun undergoes a supernova explosion. During this event, the star's core collapses under its own gravity,
causing a massive amount of matter to be compressed into an incredibly small space.

If the mass of the collapsing object is sufficient (typically above 2-3 solar masses), it will create a
singularity - a point of infinite density and zero volume. The gravitational pull of this singularity is so
strong that not even light can escape, making it invisible to us.

The exact process of black hole formation is still an active area of research, but the general consensus among
astrophysicists is as follows:

1. Stellar collapse: A massive star runs out of fuel and collapses under its own gravity.
2. Core contraction: The core of the star contracts, causing a massive amount of matter to be compressed into
an incredibly small space.
3. Singularity formation: If the mass is sufficient, a singularity forms at the center of the collapsed object.
4. Event horizon formation: The event horizon, which marks the boundary beyond which nothing can escape the
gravitational pull of the black hole, forms around the singularity.

(It's worth noting that there are different types of black holes, including stellar-mass black holes formed from
individual stars, supermassive black holes found at the centers of galaxies, and intermediate-mass black holes
with masses that fall between these two categories.)
```

CSC 5991

Ammar Ahmed - HL5101, Rensildi, Taaseen - HI7392

Homework 2 3/14/25

```
>>> The odd numbers in this group add up to an even number: 15, 32, 5, 13, 82, 7, 1. Solve by breaking the problem in... to steps. First, identify the odd numbers, add them, and indicate whether the result is odd or even.  
To solve this problem, I will break it down into steps as follows:  
  
Step 1: Identify the odd numbers in the group.  
The odd numbers are: 15, 5, 13, 7, 1.  
  
Step 2: Add up the odd numbers to find their total sum.  
Odd numbers sum:  $15 + 5 + 13 + 7 + 1 = 41$   
  
Step 3: Determine whether the result is odd or even.  
The result (41) is an odd number.  
  
>>> Send a message (/? for help)
```

Include task specific results here, make sure to have a paragraph for all 5

3. Problem 2: Fine-Tuning Llama 3.2 Using LoRA

3.1 Dataset Selection

For fine-tuning, we selected a dataset from hugging face. The dataset was chosen based on its relevance to our targeted task, ensuring sufficient data samples for meaningful improvements and it looking the easiest to work with in our opinion. Before fine tuning, the model's baseline accuracy was measured, revealing some inconsistencies in handling certain prompts, reinforcing the need for fine tuning.

CSC 5991

Ammar Ahmed - HL5101, Rensildi, Taaseen - HL7392

Homework 2 3/14/25

```
Checking performance by task type/subtask:
```

```
Overall Accuracy: 48.54%
```

```
task_name
```

```
mmlu_chat    0.485401
```

```
Name: is_correct, dtype: float64
```

```
Analyzing output metrics:
```

```
<bound method NDFrame.head of 0      {'acc': 0.0, 'correct_format': 1.0}
```

```
1      {'acc': 1.0, 'correct_format': 1.0}
```

```
2      {'acc': 1.0, 'correct_format': 1.0}
```

```
3      {'acc': 0.0, 'correct_format': 1.0}
```

```
4      {'acc': 0.0, 'correct_format': 1.0}
```

```
...
```

```
14037    {'acc': 1.0, 'correct_format': 1.0}
```

```
14038    {'acc': 1.0, 'correct_format': 1.0}
```

```
14039    {'acc': 1.0, 'correct_format': 1.0}
```

```
14040    {'acc': 1.0, 'correct_format': 1.0}
```

```
14041    {'acc': 1.0, 'correct_format': 1.0}
```

```
Name: output_metrics, Length: 14042, dtype: object>
```

```
Empty DataFrame
```

```
Columns: [parsed_accuracy]
```

```
Index: []
```

```
Extract Accurace from output metrics:
```

```
0      0.0
```

```
1      1.0
```

```
2      1.0
```

```
3      0.0
```

```
4      0.0
```

```
5      0.0
```

```
6      1.0
```

```
7      1.0
```

```
8      0.0
```

```
9      1.0
```

```
Name: parsed_accuracy, dtype: float64
```

```
Compute overall model accuracy:
```

```
Overall Model Accuracy: 48.54%
```

CSC 5991

Ammar Ahmed - HL5101, Rensildi, Taaseen - HL7392

Homework 2 3/14/25

```
Model Mistakes:

      input_question input_correct_responses \
0  Blood clots are responsible for repeatedly blo... [B]
3              What is the future of WMD? [D]
4  In which ways does realism differ from social ... [C]
5              What is 'malware'? [C]
8  Which of the following statements does not des... [C]
10 What preventative security measures might a go... [A]
14 Within Critical Security Studies, what is the ... [A]
15 Why is coercive diplomacy hard? What are the i... [C]
21 The concept of gemeinschaft developed by Ferdi... [B]
22 There was a growth in income inequality in the... [D]

      output_prediction_text
0          [ D.]
3          [ B.]
4          [ B.]
5          [ D.]
8          [ D.]
10         [ D.]
14         [ D.]
15         [ D.]
21         [ A.]
22         [ C.]
Most common mistakes:
```

```
Finding the most difficult subtasks:

subtask_name
mmlu_chat.college_physics      0.284314
mmlu_chat.moral_scenarios      0.288268
mmlu_chat.machine_learning     0.294643
mmlu_chat.econometrics         0.298246
mmlu_chat.high_school_mathematics 0.300000
Name: is_correct, dtype: float64
```

3.2 Fine Tuning Process

The LoRA fine tuning method was implemented by freezing most of the Llama 3.2 model's parameters and training only a small set of low-rank adaptation matrices. This approach allowed efficient fine tuning while preserving the original model's knowledge. The fine tuning process involved setting Rank (r) to 1, epochs. The model was trained using PyTorch and the Hugging Face Trainer API, with code modifications to integrate LoRA layers. Once fine-tuning was complete, the model was evaluated to determine improvements in performance.

CSC 5991

Ammar Ahmed - HL5101, Rensildi, Taaseen - HI7392

Homework 2 3/14/25

```
# Load dataset from Hugging Face
dataset = load_dataset("meta-llama/Llama-3.2-1B-Instruct-evals",
                       name="Llama-3.2-1B-Instruct-evals__mmlu__details",
                       split="latest")
```

```
# Split dataset into train and test sets
dataset = dataset.train_test_split(test_size=0.1)
train_data = dataset["train"]
test_data = dataset["test"]
```

```
# Load LLaMA model and tokenizer
MODEL_NAME = "meta-llama/Llama-3.2-1B-Instruct"
tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, token=HF_TOKEN)
model = AutoModelForCausalLM.from_pretrained(MODEL_NAME,
                                              device_map="auto",
                                              torch_dtype=torch.float16,
                                              token=HF_TOKEN)

# Set up LoRA configuration
lora_config = LoraConfig(
    r=1,          # LoRA rank
    lora_alpha=32, # Scaling factor
    lora_dropout=0.1, # Dropout to prevent overfitting
    bias="none",    # No bias updates
    task_type="CAUSAL_LM"
)

# Apply LoRA
model = get_peft_model(model, lora_config)
model.print_trainable_parameters()
```

```
# Ensure padding token is set
if tokenizer.pad_token is None:
    tokenizer.pad_token = tokenizer.eos_token # Use EOS token for padding

# Function to format dataset into Q&A style
def format_example(example):
    question = example["input_question"]
    answer = example["output_parsed_answer"] # Use the parsed answer
    if answer is None:
        answer = "" # Avoid training on NaN values
    formatted_text = f"Q: {question}\nA: {answer}"
    return {"text": formatted_text}

# Apply formatting
train_data = train_data.map(format_example)
test_data = test_data.map(format_example)

# Tokenization function
def tokenize_function(examples):
    return tokenizer(examples["text"], padding=True, truncation=True, max_length=128)
```

```
# Tokenize datasets
tokenized_train = train_data.map(tokenize_function, batched=True)
tokenized_test = test_data.map(tokenize_function, batched=True)

# Custom collate function to add labels (shifted input_ids)
def collate_fn(batch):
    input_ids = [example['input_ids'] for example in batch]
    attention_mask = [example['attention_mask'] for example in batch]

    # Shift input_ids to create labels (shifted by 1 for causal LM)
    labels = [example['input_ids'][1:] + [tokenizer.pad_token_id] for example in batch]
    labels = torch.tensor(labels)

    # Ensure the tensors are padded correctly
    return {
        'input_ids': torch.tensor(input_ids),
        'attention_mask': torch.tensor(attention_mask),
        'labels': labels # Pass the labels for loss calculation
    }
```

```
# Custom Trainer with time-based logging
class CustomTrainer(Trainer):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.last_log_time = time.time() # Initialize last log time

    def log(self, logs, start_time=None): # Accept start_time
        # Check if 5-10 seconds have passed since the last log
        current_time = time.time()
        time_elapsed = current_time - self.last_log_time

        if time_elapsed >= 5: # Log every 5 seconds
            super().log(logs, start_time) # Pass the start_time along
            self.last_log_time = current_time # Update last log time
            print(f"Step: {self.state.global_step}, Loss: {logs.get('loss', 'N/A')}")

    def on_epoch_end(self):
        print(f"Epoch {self.state.epoch} finished!")
        super().on_epoch_end()
```


CSC 5991

Ammar Ahmed - HL5101, Rensildi, Taaseen - HL7392

Homework 2 3/14/25

```
# Define training arguments with updated parameter names
training_args = TrainingArguments(
    output_dir="./lora-finetuned",
    eval_strategy="epoch", # Updated from `evaluation_strategy`
    save_strategy="epoch",
    num_train_epochs=10,
    per_device_train_batch_size=4,
    per_device_eval_batch_size=4,
    learning_rate=2e-4,
    weight_decay=0.01,
    logging_dir="./logs",
    logging_steps=1, # Log every step for debugging (this is now overridden by time-based logging)
    save_total_limit=2,
    push_to_hub=False,
    report_to="none", # Disable external logging services
)
```

```
# Create the trainer instance
trainer = CustomTrainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_train,
    eval_dataset=tokenized_test,
    data_collator=collate_fn # Pass custom collator function
)
```

3.3 Performance Evaluation

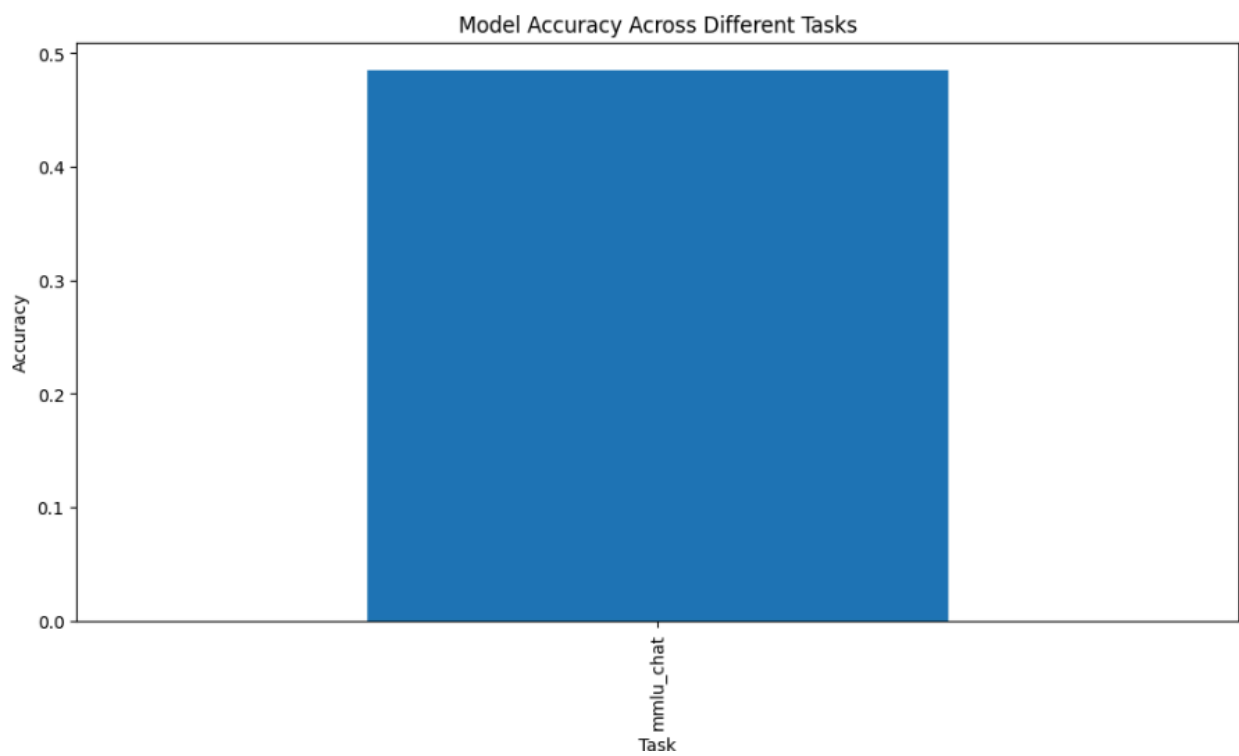
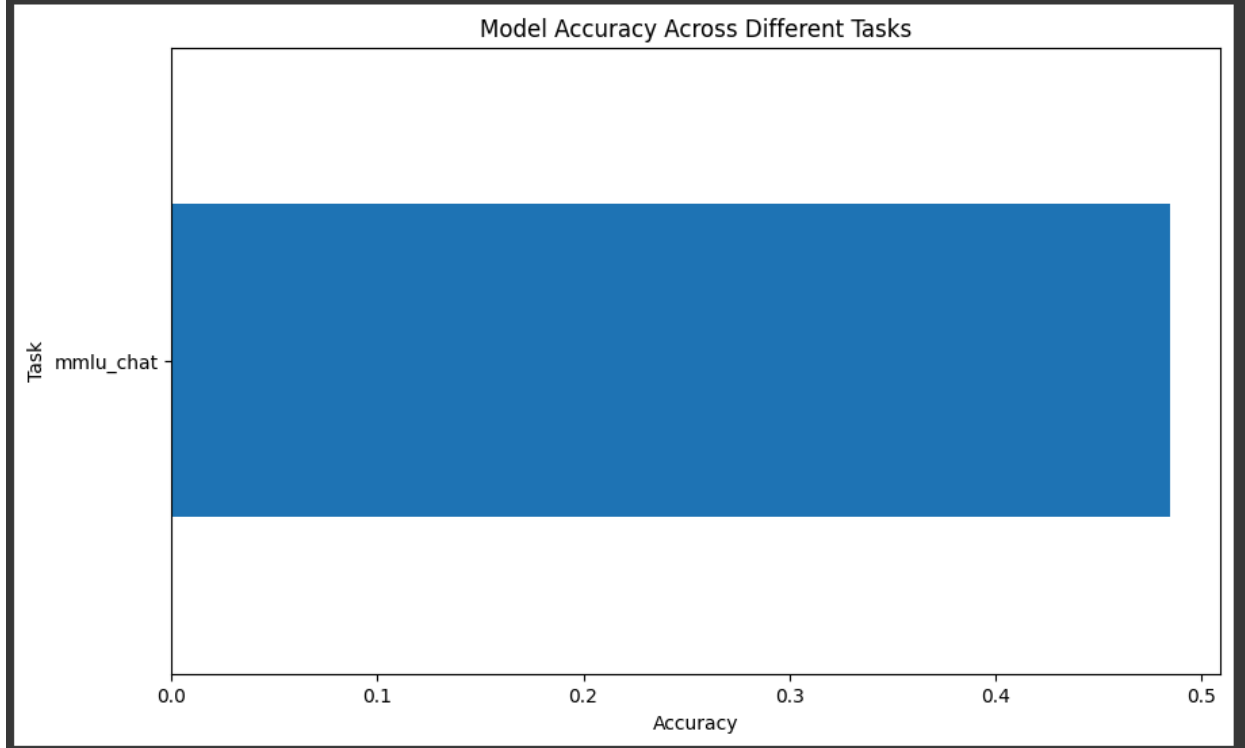
Full Info can be found here: [Homework2 Dont forget to save files.ipynb - Colab](#)

CSC 5991

Ammar Ahmed - HL5101, Rensildi, Taaseen - HI7392

Homework 2 3/14/25

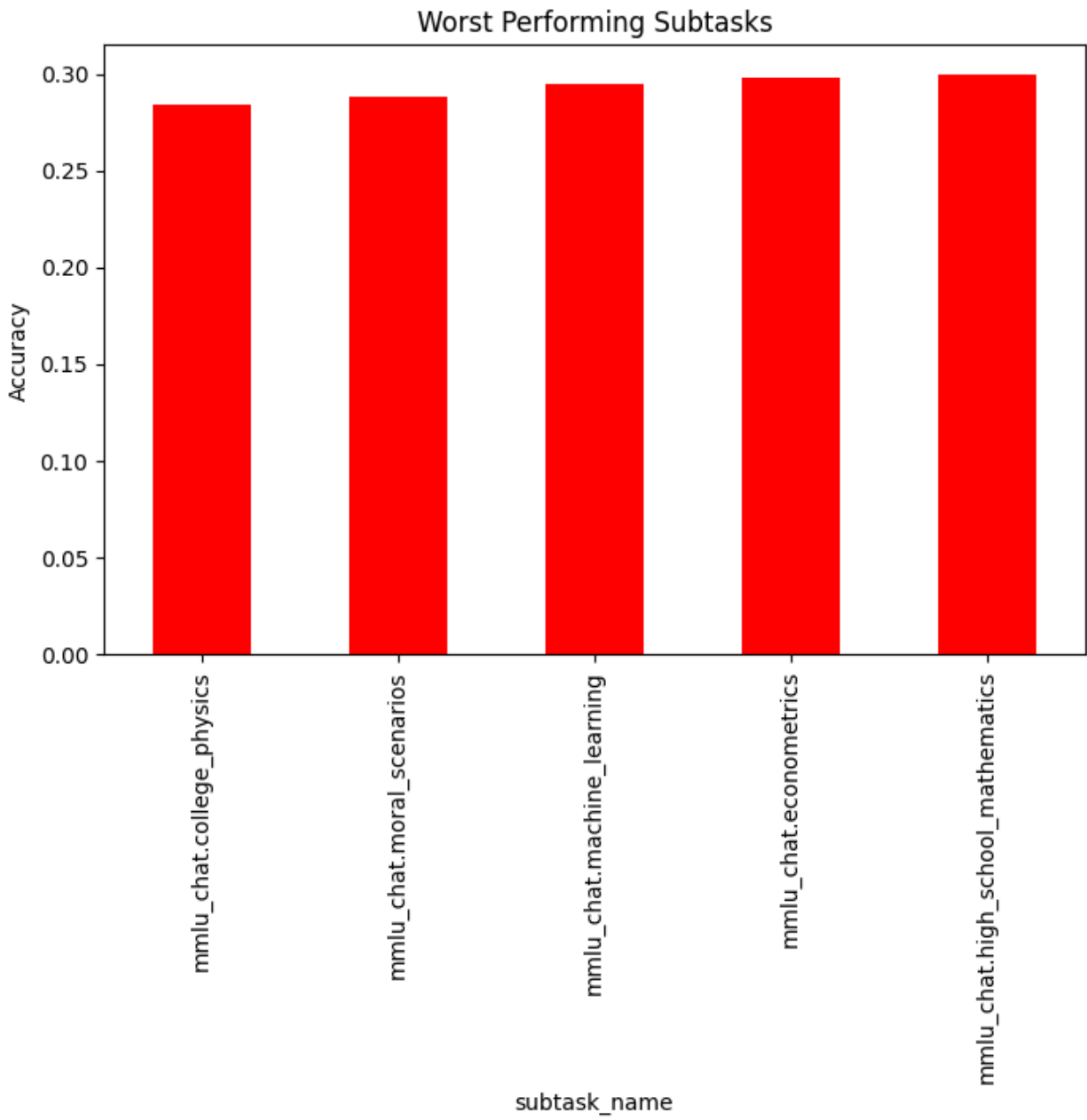
Vizualization:

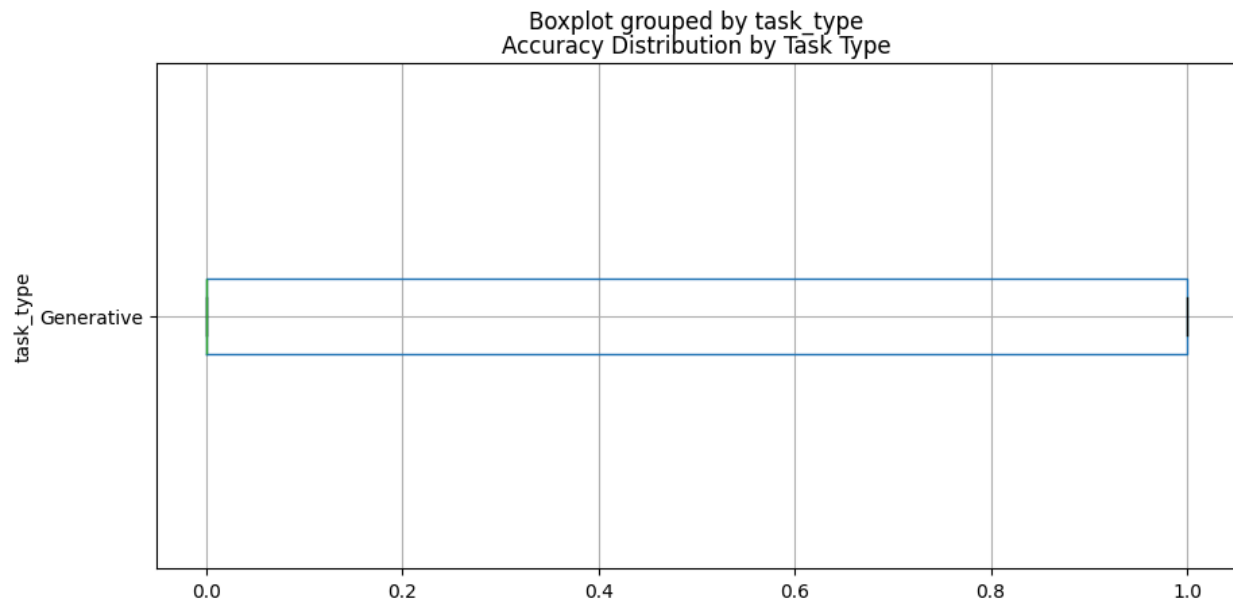


CSC 5991

Ammar Ahmed - HL5101, Rensildi, Taaseen - HI7392

Homework 2 3/14/25





As can be seen, the performance did not change noticeably before and after training for our experiment. This could be as a result of the chosen procedure and/or dataset, but not likely due to a lack of outliers in the results.

Again, please check the colab files located in the references section, they contain all of our info and notes and can be used for clarification.

4. References

- [LoRA.ipynb - Colab](#)
- [Homework2 Dont forget to save files.ipynb - Colab](#) , These two google collab files have all of our extra notes and code used for the experiment, please take a look
- Llama 3.2 Model & Documentation:
 - [Meta's Official Site](#)
 - [Ollama Llama 3.2](#)
 - [Hugging Face Repository](#)
- LoRA Fine-Tuning Resources:
 - [LoRA Paper](#)
 - [Microsoft LoRA Code](#)
 - [Google Colab LoRA Tutorial](#)
- Dataset Resources:
 - [HuggingFace Datasets](#)

CSC 5991

Ammar Ahmed - HL5101, Rensildi, Taaseen - HI7392

Homework 2 3/14/25