MMLU Benchmark

This post evaluates a language model on the MMLU (Massive Multitask Language Understanding) benchmark.

The MMLU benchmark is a comprehensive test of a model's ability to perform various tasks across a wide range of subjects. It consists of multiple-choice questions covering diverse areas such as mathematics, history, law, and medicine.

Dataset Links:

- Papers with Code
- Hugging Face Datasets

```
import torch
from datasets import load_dataset
import requests
import json

# Load MMLU dataset
subject = "abstract_algebra"  # Choose your subject
dataset = load_dataset("cais/mmlu", subject, split="test")

# Format prompt with few-shot examples
def format_mmlu_prompt(example, few_shot_examples=5):
    prompt = "The following are multiple choice questions (with answers) about {}.\n\n".format(subject.replace

    # Add few-shot examples
    few_shot_dataset = load_dataset("cais/mmlu", subject, split="validation")
    for i in range(few_shot_examples):
        ex = few_shot_dataset[i]
        prompt += f"Question: {ex['question']}\n"
        prompt += "Choices:\nA. {}\nB. {}\nC. {}\nD. {}\n".format(*ex['choices'])
        prompt += f"Answer: {ex['answer']}\n\n"

    # Add current question
    prompt += f"Question: {example['question']}\n"
    prompt += "Choices:\nA. {}\nB. {}\nC. {}\nD. {}\n".format(*example['choices'])
    prompt += "Answer:"
    return prompt

# Evaluation loop
```

```python
correct = 0
total = 0

for example in dataset:
    prompt = format_mmlu_prompt(example)

    # Send request to llama-server
    url = "http://localhost:8080/v1/chat/completions"
    headers = {"Content-Type": "application/json"}
    data = {
        "messages": [{"role": "user", "content": prompt}],
        "max_tokens": 5,
        "temperature": 0,
    }

    response = requests.post(url, headers=headers, data=json.dumps(data))

    if response.status_code == 200:
        output_text = response.json()["choices"][0]["message"]["content"]
        predicted_answer = output_text.strip()[0] if len(output_text.strip()) > 0 else ""
    else:
        predicted_answer = ""
        print(f"Error: {response.status_code} - {response.text}")

    # Compare with ground truth
    if predicted_answer.upper() == example["answer"]:
        correct += 1
    total += 1

# Calculate accuracy
accuracy = correct / total
print(f"Subject: {subject}")
print(f"Accuracy: {accuracy:.2%} ({correct}/{total})")
```