

Benchmark MMLU

Esta publicación evalúa un modelo de lenguaje en el benchmark MMLU (Massive Multitask Language Understanding).

El benchmark MMLU es una prueba exhaustiva de la capacidad de un modelo para realizar diversas tareas en una amplia gama de materias. Consiste en preguntas de opción múltiple que cubren áreas diversas como matemáticas, historia, derecho y medicina.

Enlaces al conjunto de datos:

- [Papers with Code](#)
- [Hugging Face Datasets](#)

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

# Cargar el conjunto de datos MMLU
subject = "abstract_algebra" # Elige tu materia
dataset = load_dataset("cais/mmlu", subject, split="test")

# Formatear el prompt con ejemplos few-shot
def format_mmlu_prompt(example, few_shot_examples=5):
    prompt = "Las siguientes son preguntas de opción múltiple (con respuestas) sobre {}. \n\n".format(subject)

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

    # Añadir la pregunta actual
    prompt += f"Pregunta: {example['question']} \n"
    prompt += "Opciones: \nA. {} \nB. {} \nC. {} \nD. {} \n".format(*example['choices'])
    prompt += "Respuesta: "
    return prompt

# Bucle de evaluación
```

```

correct = 0
total = 0

for example in dataset:
    prompt = format_mmlu_prompt(example)

    # Enviar solicitud a 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}")

    # Comparar con la respuesta correcta
    if predicted_answer.upper() == example["answer"]:
        correct += 1
    total += 1

# Calcular la precisión
accuracy = correct / total
print(f"Materia: {subject}")
print(f"Precisión: {accuracy:.2%} ({correct}/{total}")

```