
Ampliación de Fundamentos de Hardware

Angel Fabrizio Ullaguari Yanza

2ºASIR

Práctica IA. Fine tuning



Práctica	3
Configuración	3
Pruebas	4

Práctica

Configuración

Vamos a realizar un fine-tuning con el modelo Gemma-3-270M.

Primero, creamos la carpeta mi-finetuning para el proyecto. Luego, se crea un entorno virtual con python y lo activamos.

```
● ● ● mi-finetuning -- zsh -- 118x30
[aulaateca26@Mac-mini-de-aulaateca26 ~ % ls
Desktop Downloads mi-finetuning Music Public
Documents Library Movies Pictures
[aulaateca26@Mac-mini-de-aulaateca26 ~ % cd mi-finetuning
[aulaateca26@Mac-mini-de-aulaateca26 mi-finetuning % python -m venv venv
zsh: command not found: python
[aulaateca26@Mac-mini-de-aulaateca26 mi-finetuning % python3 -m venv venv
[aulaateca26@Mac-mini-de-aulaateca26 mi-finetuning % source venv/bin/activate
(venv) aulaateca26@Mac-mini-de-aulaateca26 mi-finetuning %
```

Ahora, se instala torch y sus dependencias para que gestionen el modelo.

```
(venv) aulaateca26@Mac-mini-de-aulaateca26 mi-finetuning % pip install torch torchvision torchaudio
Collecting torch
  Using cached torch-2.9.1-cp314-cp314-macosx_11_0_arm64.whl.metadata (30 kB)
Collecting torchvision
  Using cached torchvision-0.24.1-cp314-cp314-macosx_11_0_arm64.whl.metadata (5.9 kB)
Collecting torchaudio
  Using cached torchaudio-2.9.1-cp314-cp314-macosx_11_0_arm64.whl.metadata (6.7 kB)
Collecting filelock (from torch)
  Using cached filelock-3.20.3-py3-none-any.whl.metadata (2.1 kB)
Collecting typing_extensions>=4.10.0 (from torch)
  Using cached typing_extensions-4.15.0-py3-none-any.whl.metadata (3.3 kB)
Collecting setuptools (from torch)
  Using cached setuptools-63.5.1-py3-none-any.whl.metadata (10 kB)
```

También hay que instalar transformers (biblioteca de Hugging Face) para manejar los modelos de lenguaje, además de peft (es una herramienta para usar LoRA, que permite entrenar modelos grandes con poca memoria), y bitsandbytes (permite reducir el peso del modelo).

```
● ● ● mi-finetuning -- zsh -- 149x41
(venv) aulaateca26@Mac-mini-de-aulaateca26 mi-finetuning % pip install transformers datasets peft accelerate bitsandbytes
Requirement already satisfied: transformers in ./venv/lib/python3.14/site-packages (5.0.0rc1)
Collecting datasets
  Downloading datasets-4.5.0-py3-none-any.whl.metadata (19 kB)
Collecting peft
  Using cached peft-0.18.1-py3-none-any.whl.metadata (14 kB)
Collecting accelerate
  Using cached accelerate-1.12.0-py3-none-any.whl.metadata (19 kB)
Collecting bitsandbytes
  Downloading bitsandbytes-0.49.1-py3-none-macosx_14_0_arm64.whl.metadata (10 kB)
Requirement already satisfied: filelock in ./venv/lib/python3.14/site-packages (from transformers) (3.20.3)
Requirement already satisfied: huggingface-hub<2.0,>=1.2.1 in ./venv/lib/python3.14/site-packages (from transformers) (1.3.2)
```

En datasets editamos estos dos archivos.

train.jsonl (Entrenamiento): Son las preguntas y respuestas que la IA usa para estudiar. Es su libro de texto.

valid.jsonl (Validación): Son preguntas que la IA usa para hacerse un examen a sí misma y ver si está aprendiendo bien o si solo está memorizando.



```
UW PICO 5.09                               File: data/valid.jsonl
["messages": [{"role": "user", "content": "Explicame XAMPP."}, {"role": "assistant", "content": "XAMPP integra Apache, MySQL, PHP y Perl para desarrollar web local en Mac/Windows."}]

UW PICO 5.09                               File: data/train.jsonl
["messages": [{"role": "user", "content": "\u00e1Que es un ordenador?"}, {"role": "assistant", "content": "Un ordenador es una maquina electronica que procesa informacion media"}, {"messages": [{"role": "user", "content": "Explicame XAMPP."}, {"role": "assistant", "content": "XAMPP integra Apache, MySQL, PHP y Perl para desarrollo web local en Mac/Windows."}], {"messages": [{"role": "user", "content": "\u00e1Que es ejabberd?"}, {"role": "assistant", "content": "Ejabberd es servidor XMPP en Erlang/Ubuntu para chat escalable. Configura"}]
```

Pruebas

Entrenamos a nuestro modelo.

```
(venv) aulaateca26@Mac-mini-de-aulaateca26 mi-finetuning % TOKENIZERS_PARALLELISM=false mlx_lm.lora --model mlx-community/gemma-3-270m-it-4bit --train --data data --iters 100 --batch-size 1 --adapter-path checkpoints_ordenador --save-every 50

Loading pretrained model
Fetching 10 files: 100%[=====] 10/10 [00:00<00:00, 128266.18it/s]
Download complete: : 0.00B [00:00, ?B/s] | 0/10 [00:00<?, ?it/s]
Loading datasets
Training
Trainable parameters: 0.629% (1.688M/268.098M)
Starting training..., iters: 100
Calculating loss...: 100%[=====] 1/1 [00:00<00:00, 8.54it/s]
Iter 1: Val took 0.119s
Iter 10: Train loss 4.417, Learning Rate 1.000e-05, It/sec 10.006, Tokens/sec 497.281, Trained Tokens 497, Peak mem 0.370 GB
Iter 20: Train loss 1.042, Learning Rate 1.000e-05, It/sec 27.101, Tokens/sec 1336.075, Trained Tokens 990, Peak mem 0.370 GB
Iter 30: Train loss 0.308, Learning Rate 1.000e-05, It/sec 27.015, Tokens/sec 1323.753, Trained Tokens 1480, Peak mem 0.370 GB
Iter 40: Train loss 0.245, Learning Rate 1.000e-05, It/sec 27.038, Tokens/sec 1343.799, Trained Tokens 1977, Peak mem 0.370 GB
Iter 50: Train loss 0.201, Learning Rate 1.000e-05, It/sec 27.038, Tokens/sec 1332.955, Trained Tokens 2470, Peak mem 0.370 GB
Iter 50: Saved adapter weights to checkpoints_ordenador/adapters.safetensors and checkpoints_ordenador/0000050_adapters.safetensors.
Iter 60: Train loss 0.162, Learning Rate 1.000e-05, It/sec 26.936, Tokens/sec 1319.844, Trained Tokens 2960, Peak mem 0.379 GB
Iter 70: Train loss 0.132, Learning Rate 1.000e-05, It/sec 27.011, Tokens/sec 1331.620, Trained Tokens 3453, Peak mem 0.379 GB
Iter 80: Train loss 0.099, Learning Rate 1.000e-05, It/sec 27.018, Tokens/sec 1342.801, Trained Tokens 3950, Peak mem 0.379 GB
Iter 90: Train loss 0.069, Learning Rate 1.000e-05, It/sec 27.010, Tokens/sec 1323.467, Trained Tokens 4440, Peak mem 0.379 GB
Calculating loss...: 100%[=====] 1/1 [00:00<00:00, 58.03it/s]
Iter 100: Val took 0.019s
Iter 100: Train loss 0.045, Learning Rate 1.000e-05, It/sec 26.987, Tokens/sec 1330.449, Trained Tokens 4933, Peak mem 0.379 GB
Iter 100: Saved adapter weights to checkpoints_ordenador/adapters.safetensors and checkpoints_ordenador/0000100_adapters.safetensors.
Saved final weights to checkpoints_ordenador/adapters.safetensors.
(venv) aulaateca26@Mac-mini-de-aulaateca26 mi-finetuning %
```

```
(venv) aulaateca26@Mac-mini-de-aulaateca26 mi-finetuning % TOKENIZERS_PARALLELISM=false mlx_lm.lora --model mlx-community/gemma-3-270m-it-4bit --train --data data --iters 1000 --batch-size 1 --adapter-path checkpoints_ordenador --save-every 50
```

Con la ayuda de nano, configuraremos el archivo test_gemma.py para poder hablar con el modelo recién entrenado.



The screenshot shows a terminal window titled "mi-finetuning — nano test_gemma.py — 82x48". The window contains the following Python code:

```
UW PICO 5.09                                         File: test_gemma.py

model, tokenizer = load("mlx-community/gemma-3-270m-it-4bit", adapter_path="check$"

while True:
    prompt_text = input("\nTú: ")
    if prompt_text.lower() in ['salir', 'exit', 'q']: break
    messages = [{"role": "user", "content": prompt_text}]
    prompt = tokenizer.apply_chat_template(messages, add_generation_prompt=True)
    response = generate(model, tokenizer, prompt=prompt, max_tokens=300, temp=0.1)
    print("Gemma: " + response)
```