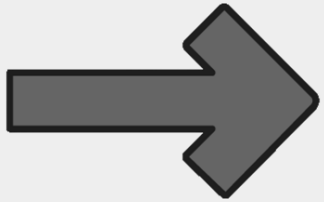
Google Developer Group
Universidad Autónoma de Madrid

Google Developer Group on Campus UAM

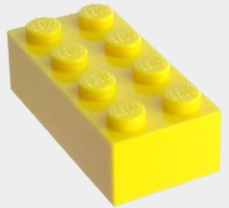
Introducción a Hugging Face



¿Qué es Hugging Face?

Permite Construir cosas complejas con piezas simples.

Plataforma de código abierto para construir IA



No necesitas ser un ingeniero, solamente una idea.

Miles de modelos pre-entrenados - Tus piezas de Lego

Generar texto, entender idiomas, crear imágenes, analizar audio... ¡listo para usar!

¿Qué es Hugging Face?

El modelo que aprendió a entender el contexto

Surge la librería **Transformers**

aws

J.P.Morgan

Azure

Open Source

LLaMA by Meta

deepseek

MISTRAL AI_

GPT-2

STABLE DIFFUSION

¿Cómo lo puedo usar?

1. Instala las librerías en Python

```
!pip install transformers
!pip install torch
```

2. Importa la librería

```
from transformers import pipeline
```

Ejemplos de uso

Análisis de Sentimiento:

```python
from transformers import pipeline

# 1. Carga la "pieza de LEGO" para análisis de sentimiento
clasificador = pipeline("sentiment-analysis")

resultado = clasificador("Brrr Brrr Bebesita!")
resultado2 = clasificador("Hugging Face es un duro")

# 3. Imprimimos el resultado
print(resultado)
print(resultado2)
```

```
No model was supplied, defaulted to distilbert/distilbert-base-
Using a pipeline without specifying a model name and revision
Device set to use cpu
[{'label': 'POSITIVE', 'score': 0.9636847972869873}]
[{'label': 'POSITIVE', 'score': 0.9959320425987244}]
```

Clasificación de Imágenes:

```python
from transformers import pipeline

# 1. Cargamos la "pieza de LEGO" para etiquetado de imágenes
image_classifier = pipeline("image-classification", model="google/vit-base-patch16-224")

# Ponemos una URL
imagen_url = "https://tse4.mm.bing.net/th/id/OIP.EhC-dFu48sefZXQLvAdH-gHaFd?rs=1&pid=ImgDetMain&o=7&rm=3"

# 3. La IA analiza la imagen
resultados_vision = image_classifier(imagen_url)

# 4. Imprimimos lo que la IA ve
print(resultados_vision)
```

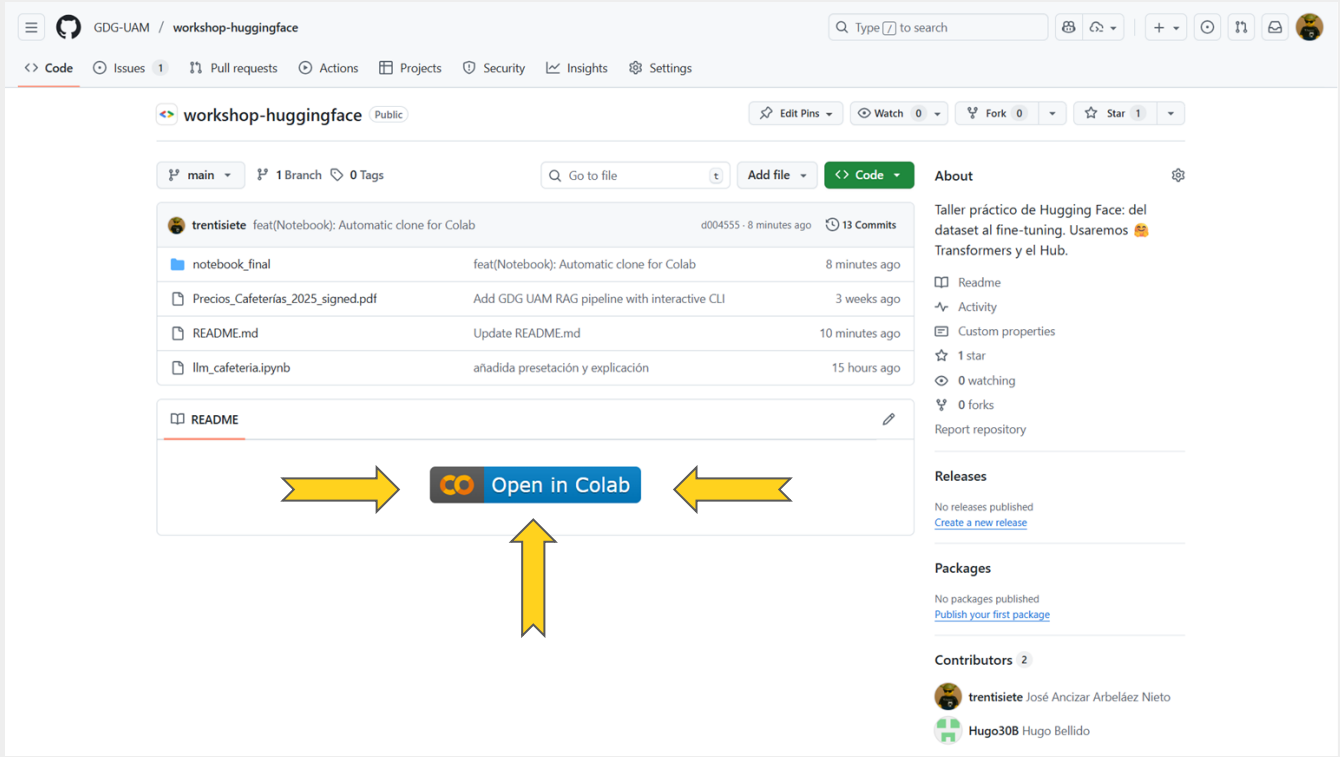
```
[{'label': 'pelican', 'score': 0.9853333830833435},
{'label': 'American egret, great white heron, Egretta albus', 'score': 0.0026109481696039044},
 {'label': 'albatross, mollymawk', 'score': 0.0016224891878664494},
{'label': 'black stork, Ciconia nigra', 'score': 0.0009927289793267846}, {'label': 'goose', 'score': 0.0007832114933989942}]
```

Manos a la obra

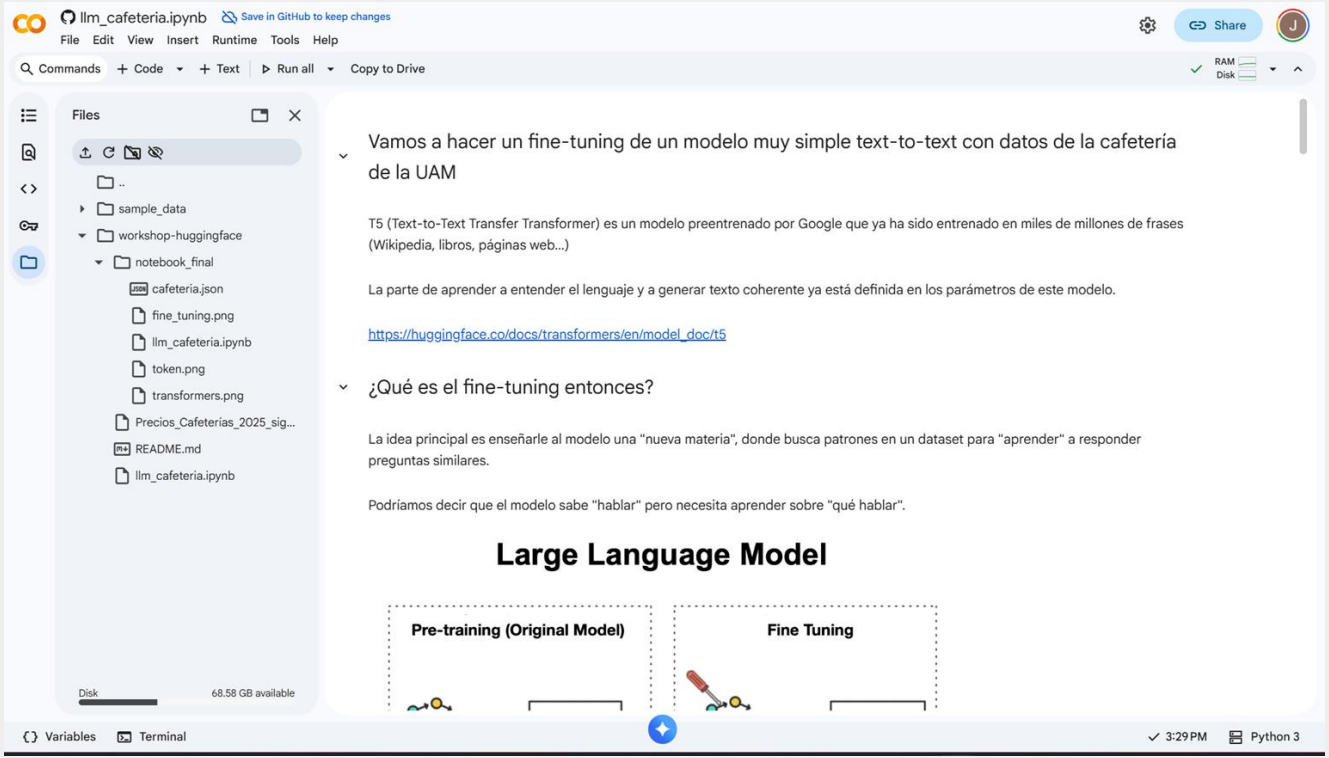
Accede al repositorio del Taller

https://github.com/GDG-UAM/workshop-huggingface

Manos a la obra



Manos a la obra



Te toca a ti

https://huggingface.co/spaces

Modelos que podemos usar

Los modelos se pueden usar de 2 formas:

1. **En Remoto con la clase InferenceClient:**
   Nos permite hacer peticiones al servidor de Hugging Face que aloja el modelo.
   (Lo que hemos hecho con Spaces)

2. **Localmente con libreria Transformers:**
   Descargas los pesos del modelo (a veces GBs de datos) a tu propio PC, servidor o Google Colab.
   (Lo que hemos hecho con Colab)

Modelos que podemos usar

Texto

Chatbots:

- `meta-llama/Meta-Llama-3-8B-Instruct:` Estándar Open Source
- `google/gemma-3-4b-it:` Modelo mas nuevo de Google.
- `mistralai/Mistral-7B-Instruct-v0.2:` Pequeño y potente

Clasificación de Texto (Zero-Shot):

- `MoritzLaurer/mDeBERTa-v3-base-mnli-xnli:` Rápido y Multilingüe
- `facebook/bart-large-mnli:` Estándar pero a veces da Timeout.

Modelos que podemos usar

Traducción:

- `Helsinki-NLP/opus-mt-en-es:` Modelos especializados para pares de idiomas.

Visión (Computer Vision)

Generación de Imagen:

- `stabilityai/stable-diffusion-xl-base-1.0:` Estándar de oro.
- `stabilityai/stable-diffusion-3-medium-diffusers-hf:` Mas nuevo y potente.

Clasificación de Imagen:

- `google/vit-base-patch16-224:` El Vision Transformer estándar de Google.

Modelos que podemos usar

Visión (Computer Vision)

Detección de Objetos:

- `facebook/detr-resnet-50:` El modelo DETR original de Facebook.
- `YOLO:` Los Yolo también están en HF pero se suele usar más su librería propia `ultralytics`.

Modelos que podemos usar

Audio

Transcripción de Audio (Voz-a-Texto):

- `openai/whisper-large-v3:` El mejor modelo open-source para transcribir audio, es increíblemente preciso.
- `stabilityai/stable-diffusion-3-medium-diffusers-hf: Mas nuevo y potente.`

Generación de Texto-a-Voz (TTS):

- `facebook/mms-tts-spa:` Útil para generar voz en Español.

Modelos que podemos usar

Multimodal (Texto + Imagen)

Transcripción de Audio (Voz-a-Texto):

- `llava-hf/llava-1.5-7b-hf`: Puedes darle una imagen y hacerle preguntas sobre ella

Generación de Texto-a-Voz (TTS):

- `facebook/mms-tts-spa`: Útil para generar voz en Español.

Modelos que podemos usar

Multimodal (Texto + Imagen)

Transcripción de Audio (Voz-a-Texto):

- `llava-hf/llava-1.5-7b-hf`: Puedes darle una imagen y hacerle preguntas sobre ella

Generación de Texto-a-Voz (TTS):

- `facebook/mms-tts-spa`: Útil para generar voz en Español.

Usalo en Hugging Face Spaces con python:

```
from huggingface_hub import InferenceClient
import os

HF_TOKEN = os.environ.get("HF_TOKEN")

client = InferenceClient(
    "meta-llama/Meta-Llama-3-8B-Instruct", # Elige el modelo
    token=HF_TOKEN
)
response = client.chat_completion(
    messages=[{"role": "user", "content": "Escribe un poema sobre un
robot"}],
    max_tokens=100
)
print(response.choices[0].message.content)
```

Usalo en tu web con curl o fetch desde JS, JAVA o otro lenguaje:

```
curl [https://api-
inference.huggingface.co/models/EL_MODELO](https://api-
inference.huggingface.co/models/EL_MODELO) \
    -X POST \
    -d '{"inputs": "Tu texto de entrada"}' \
    -H "Authorization: Bearer $HF_TOKEN" \
    -H "Content-Type: application/json"
```

Usalo como lo hemos enseñado en el taller para hacer Fine-Tuning con la librería Transformers:

```
from transformers import pipeline
HF_TOKEN = os.environ.get("HF_TOKEN")

clasificador = pipeline(
    task="zero-shot-classification",
    model="MoritzLaurer/mDeBERTa-v3-base-mnli-xnli"
    # device=0

letra = "Me cansé de llorar, no quiero sufrir más"
etiquetas = ["fiesta", "tristeza", "amor"]

resultado = clasificador(letra, etiquetas)
print(resultado)
{'sequence': 'Me cansé de llorar, no quiero sufrir más',
#  'labels': ['tristeza', 'amor', 'fiesta'],
#  'scores': [0.95, 0.03, 0.02]}
```

Usalo como lo hemos enseñado en el taller para hacer Fine-Tuning con la librería Automodel:

```
from transformers import AutoTokenizer, AutoModelForCausalLM
import torch

model_id = "gpt2"
tokenizer = AutoTokenizer.from_pretrained(model_id)
model = AutoModelForCausalLM.from_pretrained(model_id)

inputs = tokenizer("El futuro de la IA es", return_tensors="pt",
padding=True)

outputs = model.generate(input_ids=inputs["input_ids"],
max_new_tokens=20)
texto_generado = tokenizer.decode(outputs[0], skip_special_tokens=True)
```

¡Registrate!

Nuestra Web

Comunidad Oficial

puede que venga con regalo...

Muchas gracias y bienvenidos a la comunidad.

Google Developer Group
Universidad Autónoma de Madrid