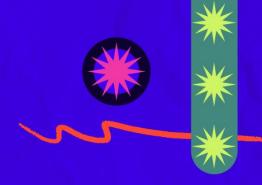


# Laboratorio Local de IA &

https://github.com/marisbotero/pycon







## ¿Por qué correr modelos localmente?

- Sin depender de la nube
- Privacidad y control total de tus datos
- 💸 Cero costos por tokens
- III Funciona sin conexión a internet
- Perfecto para aprender cómo funciona un LLM por dentro





# ¿Cómo funciona un LLM por dentro?

Un Large Language Model (LLM) como LLaMA, GPT o Mistral es un tipo de modelo de redes neuronales entrenado para predecir la siguiente palabra (o token) dada una secuencia de entrada.







#### Cuando tú escribes:

"¿Cuál es la capital de Colombia?"

```
Input → ["¿", "Cuál", "es", "la", "capital", "de", "Colombia", "?"]
Tokens → [29871, 2741, 338, 263, 5043, 309, 18135, 29991]
return go(f, seed, [])
}
```







#### Embeddings: darle forma y magia a las runas



token\_embedding = embedding\_matrix[token\_id] # e.g., vector de 4096 dimensiones







## El Transformer: la torre mágica de atención

Q: consulta (query) del token actual

K/V: claves y valores de tokens pasados

Se multiplica todo para decidir qué tokens anteriores tienen más "peso mágico".

 $Attention(Q,K,V) = softmax(QK^T/\sqrt{d})*V$ 







#### Predicción de la próxima palabra

Un Large Language Model (LLM) como LLaMA, GPT o Mistral es un tipo de modelo de redes neuronales entrenado para predecir la siguiente palabra (o token) dada una secuencia de entrada.

```
next_token_logits = linear_layer(output_vector)
probs = softmax(next_token_logits)
e.g., vector de 4096 dimensiones
```







#### ¿Qué haremos en este taller?

Ejecutar un modelo de lenguaje en tu computador sin conexión.

Crear un chatbot con Streamlit.

Explorar cómo funciona Llama internamente.

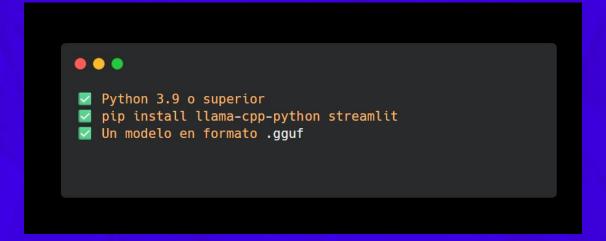
Y al final... invocar un entorno mágico con Docker 🐳







### **Requisitos**









#### Estructura del proyecto Python



```
llama_lab/
    app.py # Chatbot en Streamlit
    main.py # Interfaz por consola
    models/ # Carpeta para archivos .gguf
    requirements.txt
```









Magia	Término técnico
Piedra rúnica	Token
Conjuro	Predicción
Torre mágica	Transformer
Cristal pulido	Capa feedforward
Bola de cristal	Salida del modelo







#### ¿Qué es Llama.cpp?

Es una biblioteca ligera para correr modelos LLM de Meta localmente en CPU/GPU.

Ideal para:

Laboratorios educativos

Demostraciones sin conexión

Experimentación sin depender de la nube







#### Docker

Docker está simplificando la creación de apps agenticas: Modelos curados en Docker Hub Ejecución local con Model Runner Seguridad integrada Herramientas familiares







### Herramientas mágicas necesarias

- Docker Desktop (versión 4.40 o superior)
- Model Runner de Docker activado
- Puerto TCP para conectarte desde Python







#### ¿Qué es Model Runner?

Docker Model Runner es como un bibliotecario mágico que te entrega modelos LLM listos para usar con un solo comando.

- Carga modelos de lenguaje dentro de contenedores
- Te da un endpoint tipo OpenAI (/v1/chat/completions)
- Corre en tu máquina, sin internet, sin nube, sin llaves secretas







#### ¿Por qué usar Model Runner?

- Fácil para principiantes
- Rápido y sin configuración
- Reutilizable desde cualquier lenguaje
- 🔒 Local, privado y sin APIs externas
- / Ideal para pruebas, demos y experimentos creativos







- # <u>1</u> 1. Activar Docker Model Runner docker desktop enable model-runner --tcp 12434
- # \( \to 2. \) Descargar modelo liviano (Gemma o Smollm2) docker model pull ai/gemma3-qat:1B-Q4\_K\_M docker model pull ai/smollm2
- # **4** 3. Ver modelos disponibles docker model list
- # # 4. Ejecutar el modelo directamente (modo consola)
  docker model run ai/gemma3-qat:1B-Q4\_K\_M
- # 5. Navegar a tu proyecto local cd ~/Desktop/ia-local-docker
- # 💆 6. Levantar la WebUI (si tienes docker-compose.yml)
  docker compose up
- # # 7. Abrir la app en navegador http://localhost:3000
- # / 8. Ver contenedores activos y detenerlos
  docker ps
  docker stop <container\_id>
  # Carpeta para archivos .gguf
  \_\_ requirements.txt











Modelos para descargar
in TheBloke en Hugging Face (modelos GGUF):
huggingface.co/TheBloke

Modelos oficiales de Meta (LLaMA): meta.com/llama

Docker y Model Runner
Guía oficial de Docker Model Runner:
docs.docker.com/genai

Nost: Run LLMs Locally con Docker: Docker Blog - Run LLMs Locally





### Recursos mágicos para seguir explorando



- 🐍 Python y librerías útiles
- Ilama-cpp-python: github.com/abetlen/llama-cpp-python
- Aprendizaje continuo
- Curso de Hugging Face sobre Transformers: huggingface.co/course
- Notebook de ejemplo en Colab:Google Colab: llama-cpp-python + GGUF







¡Muchas gracias por compartir este espacio conmigo!

**@marisbotero** 



