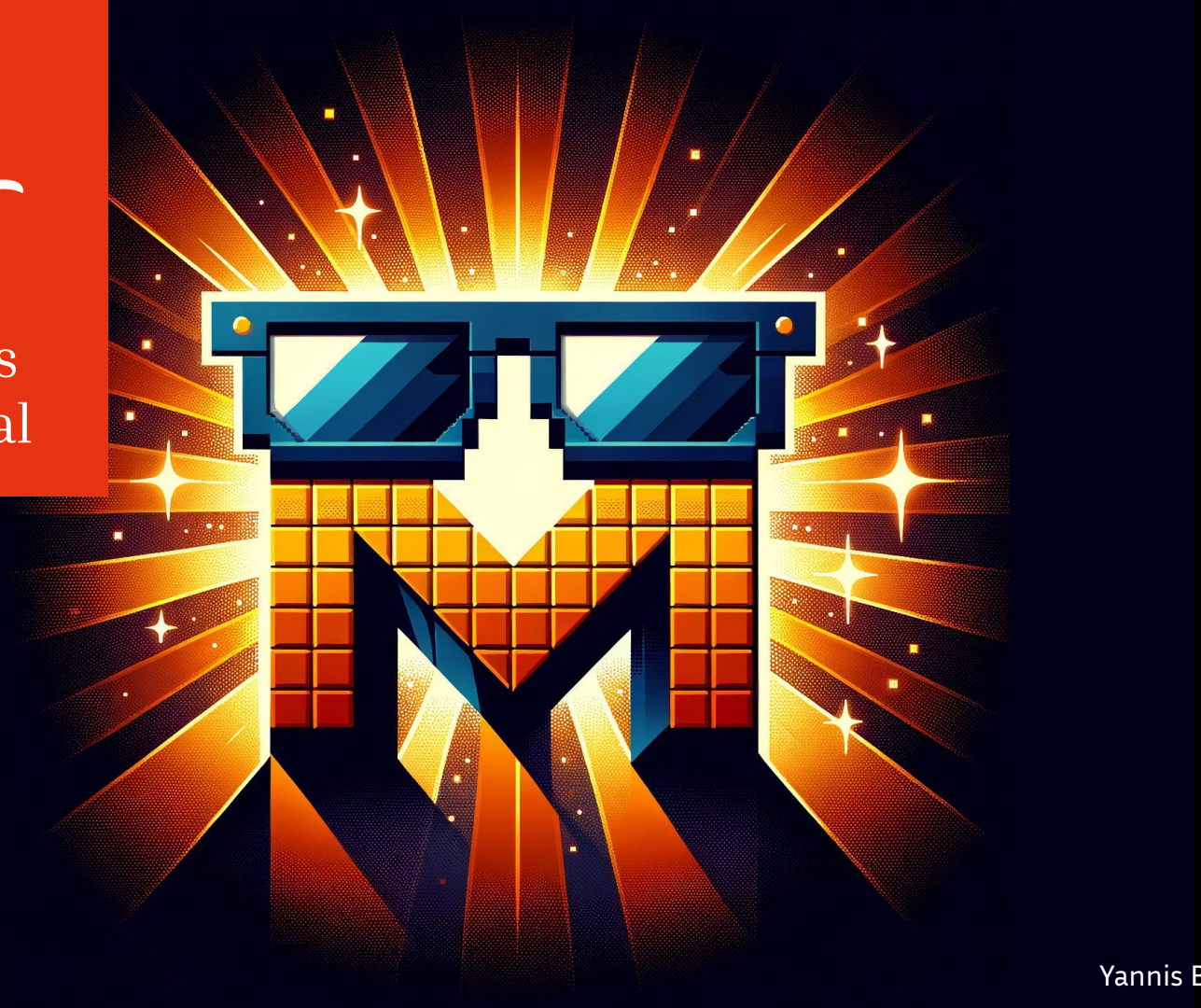
Déployer des LLMs en local

Yannis Bendi-Ouis, Xavier Hinaut

Déployer des LLMs en Local



Yannis Bendi-Ouis

Yannis Bendi-Ouis

Doctorant dans l'équipe Mnemosyne

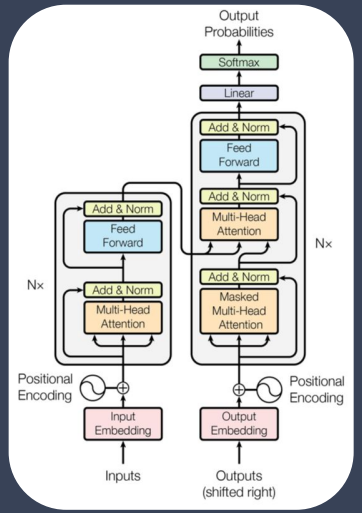
Encadré par Xavier Hinaut

Email :
yannis.bendi-ouis@inria.fr

Site web :
www.naowak.fr

Qu'est ce qu'un LLM ?

- ChatGPT
- LLM : Large Language Model (Gros modèle de langage)
- Génère du texte
- Basé sur une architecture : Transformers
- Beaucoup de paramètres/calculs



Pourquoi faire du local ?

- Pas besoin de connexion internet
- Force l'utilisation de modèles plus petits et moins énergivores
- Protection des données et de la vie privée
- Décentralisation du pouvoir (toute les données ne vont pas au même endroit)
- Réduction des risques de biais volontaires (propagande, manipulation de masse)
- Modèles ouverts (étudiables, on peut observer ses tendances/biais, maîtrise ses MAJ)
- Possibilités de finetune (contrôle du process de A à Z).

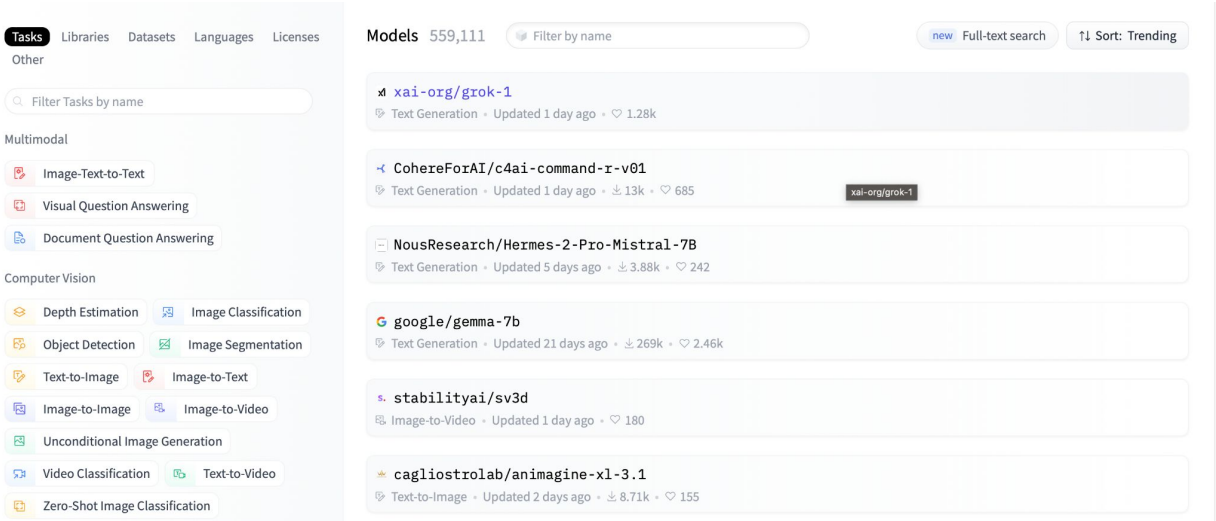
01

De nombreux modèles

Du choix pour tous

Hugging Face

Hugging Face



Llama 2 & Mistral

Meta

Llama 2

Meta AI

7B, 13B, 70B

MISTRAL AI_

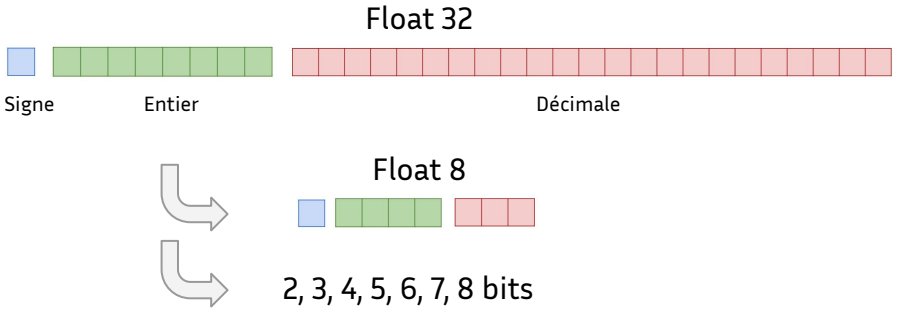
Mistral 7B

Mixtral 8x7B

Llama 2 & Mistral

Rank	Model	☆ Arena Elo	Organization	License
1	GPT-4-1106-preview	1251	OpenAI	Proprietary
1	GPT-4-0125-preview	1249	OpenAI	Proprietary
1	Claude 3 Opus	1247	Anthropic	Proprietary
4	Bard (Gemini Pro)	1202	Google	Proprietary
4	Claude 3 Sonnet	1190	Anthropic	Proprietary
5	GPT-4-0314	1185	OpenAI	Proprietary
7	GPT-4-0613	1159	OpenAI	Proprietary
7	Mistral-Large-2402	1155	Mistral	Proprietary
8	Qwen1.5-72B-Chat	1146	Alibaba	Qianwen LICENSE
8	Claude-1	1145	Anthropic	Proprietary
8	Mistral Medium	1145	Mistral	Proprietary
12	Claude-2.0	1126	Anthropic	Proprietary
12	Mistral-Next	1123	Mistral	Proprietary
12	Gemini Pro (Dev API)	1118	Google	Proprietary
13	Claude-2.1	1115	Anthropic	Proprietary
13	Mixtral-8x7b-Instruct-v0.1	1114	Mistral	Apache 2.0

Quantification



> Taille d'un modèle quantifié

7B ~ 30 Go → 4 bits → ~ 4 Go

> RAM / VRAM nécessaire

7B ~ 30 Go → 4 bits → ~ 7 Go

02

LM Studio

Une interface graphique pour du local

LM Studio

LM Studio

> Logiciel propriétaire

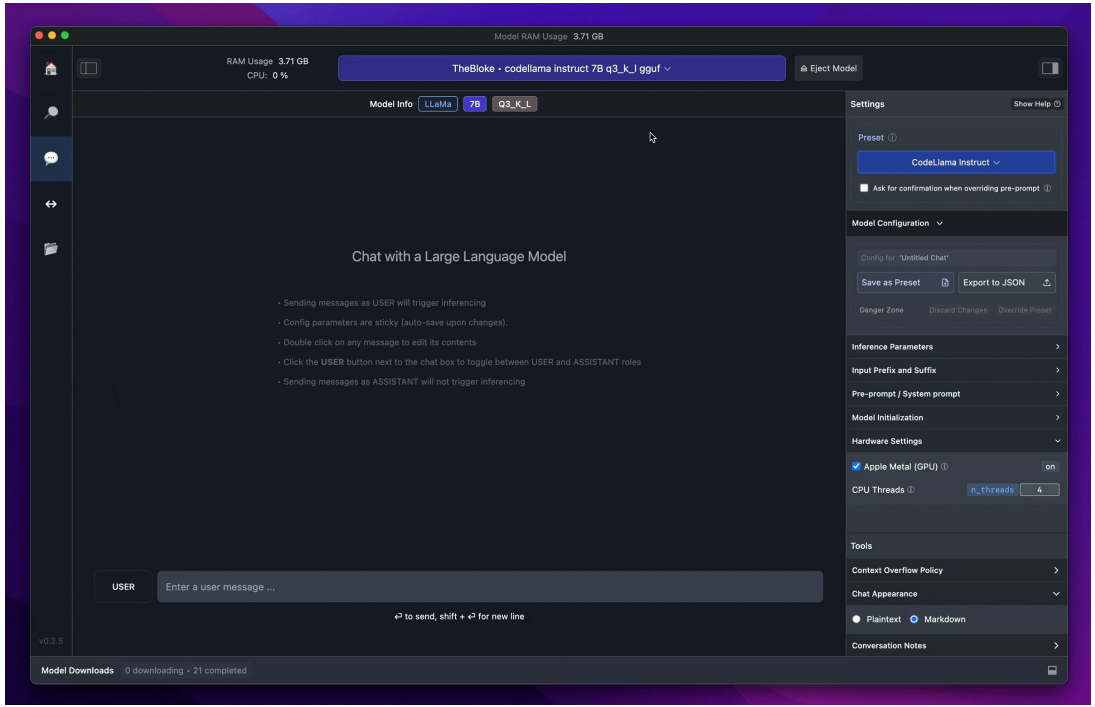
> Gratuit

> Facile

> Aucune connaissance en programmation requise

> Installation d'un modèle quantifié à partir de HuggingFace

> Configuration Automatique / Manuelle



Inria

03

Ollama

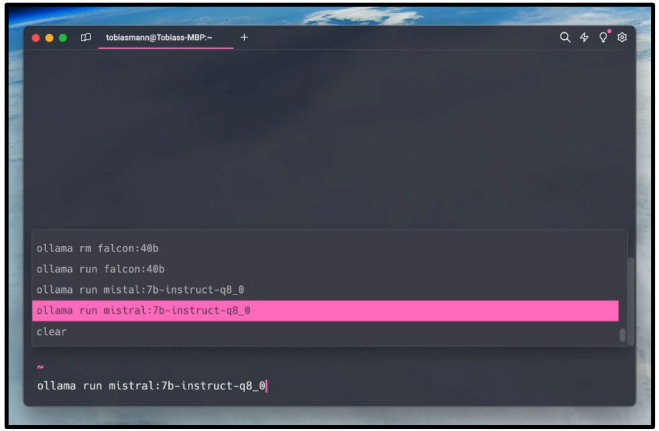
Intégration facile dans le terminal
et plus encore

Ollama

> Open source

> Gratuit

> Facile



> Installation et utilisation facile dans un terminal

> Lancement facile d'un serveur

> Plusieurs interfaces graphiques disponibles

> Connexion facile à d'autres applications

04

Transformer & Llama.cpp

Du développement en python

Transformers

> Open source

> Gratuit

> Facile mais développement python

> Implémentation Pytorch & Tensorflow

> Mis à jour très régulièrement

> Accepte énormément d'architecture de LLM

Hugging Face

```
>>> from transformers import AutoModelForCausalLM, AutoTokenizer
>>> device = "cuda" # the device to load the model onto

>>> model = AutoModelForCausalLM.from_pretrained("mistralai/Mixtral-8x7B-v0.1")
>>> tokenizer = AutoTokenizer.from_pretrained("mistralai/Mixtral-8x7B-v0.1")

>>> prompt = "My favourite condiment is"

>>> model_inputs = tokenizer([prompt], return_tensors="pt").to(device)
>>> model.to(device)

>>> generated_ids = model.generate(**model_inputs, max_new_tokens=100, do_sample=True)
>>> tokenizer.batch_decode(generated_ids)[0]
"The expected output"
```

Llama.cpp

> Open source

> Gratuit

> Facile mais développement C++/python

> Permet l'utilisation de modèle quantifié (GGUF)

> Mis à jour très régulièrement

> Accepte énormément d'architecture de LLM

Communauté sur Reddit

05

vLLM

Déploiement à grande échelle

vLLM

> Librairie open-source

> Fonctionne sur les GPU NVIDIA

> Optimise les calculs et permet de traiter plusieurs utilisateurs en simultané

A100

Mixtral

x20

~ 20 tokens/seconde

Merci !

Suivez-nous sur www.inria.fr