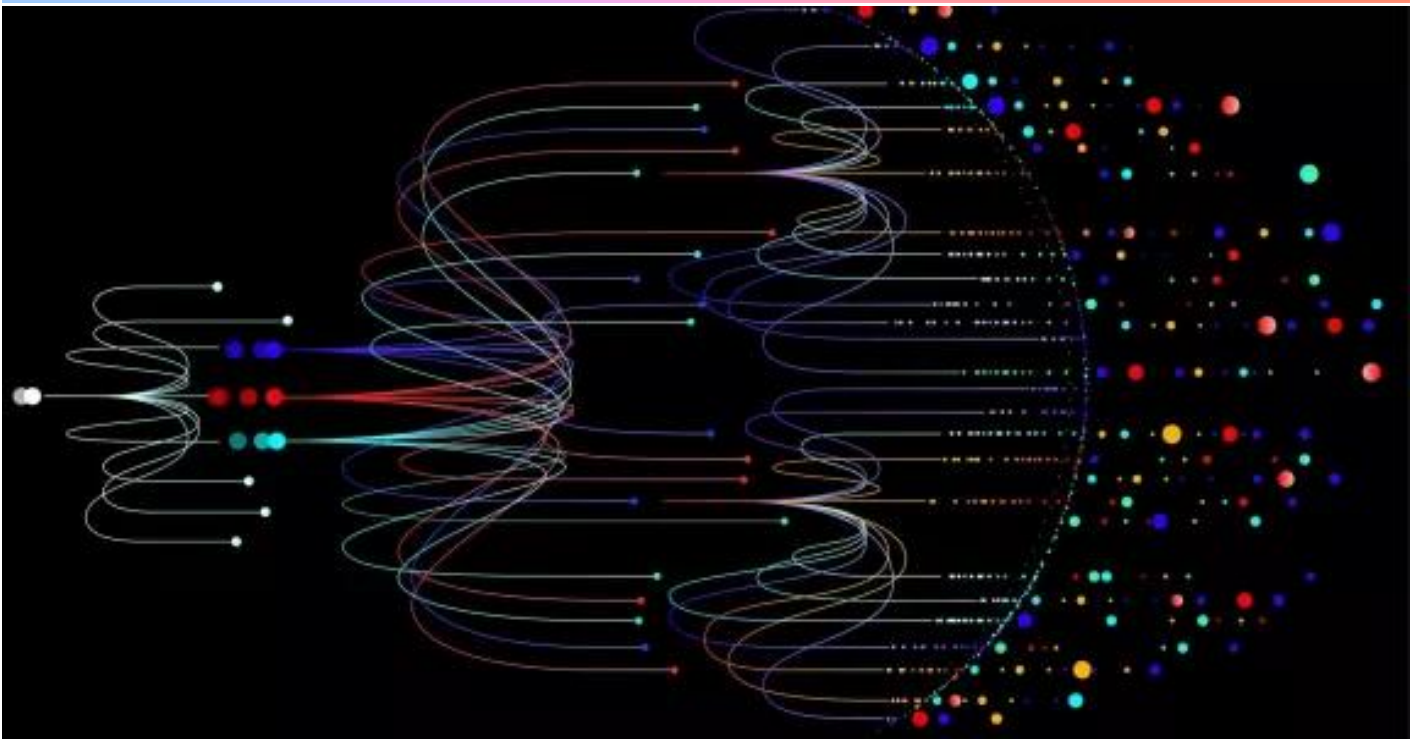




# Quatre mois d'expérimentation avec l'IA générative

25 juin 2024, 13:42

Nick Boucart

Marouene Oueslati

## Expérience acquise dans le domaine de l'élaboration d'applis reposant sur de grands modèles de langage (LLM)

L'élaboration d'une appli d'IA générative est-elle ardue ? Cette question méritait d'être étudiée : nous avons travaillé avec des applications reposant sur de grands modèles de langage (LLM). Quatre mois plus tard, il est temps de partager quelques-unes de nos expériences.

C'est en collaboration avec Niels Holvoet et Marouene Oueslati, experts Sirris, qu'ont été menées diverses expériences en matière d'élaboration d'applis reposant sur de grands modèles de langage (LLM). Nous avons déployé plusieurs démonstrations de faisabilité et découvert que l'itinéraire qui mène de la démonstration de faisabilité à la production est semé d'embûches... Nous avons conçu un chercheur de startups qui, sur la base d'un argumentaire de haut niveau élaboré par un créateur, se met à la recherche en ligne de concullègues et résume ses conclusions dans un rapport. Nous avons créé un robot conversationnel afin de clavarder avec un certain nombre de fichiers PDF ; il s'agit à n'en pas douter du cas d'utilisation par excellence de la RAG (génération augmentée d'informations contextuelles). La crewAI nous a même permis d'élaborer des applications multiagents (lesquelles communiquent entre elles par le biais de de grands modèles de langage (LLM) pour procéder ensemble à l'exécution de tâches plus complexes).

## Quelques observations

- Diverses bibliothèques telles que LangChain (bibliothèque Python facilitant la réalisation d'une interface avec les LLM) et Gradio (bibliothèque Python autorisant l'élaboration rapide d'interfaces Web) permettent à une personne possédant quelques compétences en Python d'interagir sans peine avec les LLM.
- L'usage d'outils tels qu'Ollama ou lmstudio vous permettra de travailler en toute simplicité avec des LLM locaux, à telle enseigne que vous n'aurez plus besoin d'acheter continuellement des jetons auprès d'OpenAI. Au demeurant, LangChain permet de changer, sans difficulté, de LLM.
- D'aucuns continuent à s'étonner de voir un système informatique exécuter des tâches qui, jusqu'à une date récente, semblaient relever du domaine exclusif des humains.
- Cela étant, la production d'une démonstration de faisabilité de l'IA générative n'est pas pour demain.

## Quelques constatations moins réjouissantes

Bien qu'au niveau technique, LangChain permette de travailler sans peine avec différents LLM, ces langages ne sont pas interchangeables : les invites aptes à produire de bons résultats avec un LLM donné ne le font pas nécessairement avec un autre LLM. La génération d'invites s'avère particulièrement chronophage. Dans la pratique, cet effort ne va pas sans son lot de tâtonnements.

Cette approche empirique consiste bien évidemment à vérifier régulièrement si le système produit les résultats souhaités. Comme ces résultats sont le fruit de tâches moins triviales (p. ex., résumer un texte, consulter les résultats de recherches sur Google, ...), leur évaluation s'avère assez difficile : d'une part, notre capacité à construire un système informatique apte à la production de synthèses, et ce en dépit de vagues notions de traitement automatique du langage naturel (NLP) est proprement étourdissante. D'autre part, et non sans un brin de scepticisme : les résultats sont souvent satisfaisants, pour autant que les questions aient été formulées avec soin. Si l'on s'écarte un tant soit peu de cette exigence, la qualité des réponses risque d'en pâtir immédiatement. Cette faiblesse fragilise le système et compromet le déploiement à grande échelle de telles démonstrations de faisabilité. En tant qu'ingénieur en logiciels, vous vous demandez comment faire subir un « essai unitaire » à un système basé sur des LLM.

Comme dans bien d'autres domaines, l'expérimentation d'applis reposant sur des LLM n'est pas gratuite. L'exploitation locale de grands modèles de langage (GPT-3.5 ou GPT-4, modèles en source libre comportant davantage de 7 billions de paramètres, ...) est inconcevable à moins de disposer d'une carte graphique des plus puissantes. Par conséquent, vous ne pouvez en envisager l'exploitation qu'avec des fournisseurs de LLM tels qu'OpenAI ou les déployer vous-même qu'avec le concours de Runpods, d'AWS Bedrock ou d'autres fournisseurs qui proposent de puissantes unités de traitement graphique (GPU). Mais ce n'est pas gratuit ! Le coût de quelques heures d'expérimentation menées avec crewAI et un modèle LLM plus conséquent et affectées exclusivement à la génération d'invites s'élève facilement à quelques dizaines d'euros. Sans parler du déploiement de votre solution auprès de plusieurs utilisateurs. Par conséquent, vous devrez impérativement surveiller vos coûts et veiller à ce que le retour sur investissement de votre application reste malgré tout positif.

Alors que les modèles de moindre envergure sont plus sensibles, les modèles LLM plus conséquents (p. ex. GPT-4) s'en sortent peu mieux avec des invites sous-optimales. À l'inverse, si vous consacrez plus de temps à l'élaboration d'invites adéquates (p. ex., few-shot prompting, ...)

pour certaines tâches, un modèle LLM plus modeste permet parfois d'atteindre la précision requise, et ce à un coût très inférieur.

L'abréviation LLMOps ne vous est peut-être pas étrangère ? En tout état de cause, nous sommes à même d'en imaginer la portée : suivi de l'assurance de qualité des réponses et des entrées, déploiement en temps réel des LLM adéquats (avec les invites correspondantes) en fonction de la tâche requise, et ce en veillant à maintenir un équilibre raisonnable entre les coûts et les avantages, ... Nous nous attendons à en apprendre davantage dans les mois à venir.

Si vous souhaitez partager avec nous vos expériences personnelles ou relever certains défis

[Faites-le nous savoir !](#)

Comme nous assistons à l'émergence de cette problématique dans bon nombre d'entreprises, [nous avons lancé le projet collectif de recherche LISA \(LLM Implementation, Security & Adaptation\)](#), avec le concours de DistriNet, pour travailler sur ce thème. Si ce projet vous intéresse

[N'hésitez pas à prendre contact avec nous !](#)

## Auteurs



Nick Boucart



Marouene Oueslati