
Introduction à l'IA

L'IA au service des développeurs



Lucas Delestre

Aly Ayinde

BUT GÉNIE ÉLECTRIQUE ET INFORMATIQUE INDUSTRIELLE

Année universitaire 2023/2024

Table des matières

1	Introduction	2
2	État de l'art	3
2.1	Un peu d'histoire	3
2.2	Chatbots	4
2.2.1	Problème de la sur-utilisation des chatbots	4
2.3	Âge du modèle et solutions	5
3	Mais aussi : les moteurs de recherche « amélioré » par l'IA	6
4	Réellement utile?	7
5	Conclusion	9

1 Introduction

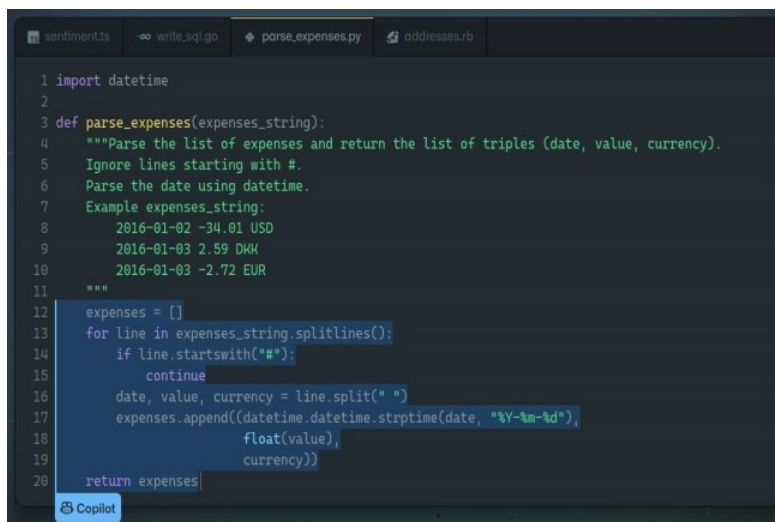
Ce court rapport a pour objectif une introduction aux IA pour les développeurs, une description de l'état de l'art actuellement, et une idée de ce que le futur peut attendre.

2 État de l'art

2.1 Un peu d'histoire

L'utilisation de l'IA pour aider les développeurs informatique dans leur métier est relativement récente. Le plus ancien système est GitHub Copilot, annoncé en mi-2021. Cela précède de quelques années tout l'engouement récent autour des IA génératives d'images et chatbots (LLM, *Large Language Model*), type ChatGPT. Cette application est basée sur modèle GPT-3, sorti par 2020 en OpenAI, qui a ensuite été entraîné sur tout le code source disponible sur GitHub. Cela pose donc des questions de respect du droit d'auteur sur le matériel d'entraînement.

Toutes les applications suivent la même formule : dans l'éditeur de code, l'IA analyse le contexte (le contenu du fichier, du projet, etc, et surtout la position du curseur) pour déterminer une suite potentielle, et l'affiche en gris à la suite du curseur. Si le code généré est correct selon le développeur, il suffit d'appuyer sur la touche Tab.



```
1 import datetime
2
3 def parse_expenses(expenses_string):
4     """Parse the list of expenses and return the list of triples (date, value, currency).
5     Ignore lines starting with #.
6     Parse the date using datetime.
7     Example expenses_string:
8     2016-01-02 -34.01 USD
9     2016-01-03 2.59 DKK
10    2016-01-03 -2.72 EUR
11    """
12    expenses = []
13    for line in expenses_string.splitlines():
14        if line.startswith("#"):
15            continue
16        date, value, currency = line.split(" ")
17        expenses.append((datetime.datetime.strptime(date, "%Y-%m-%d"),
18                        float(value),
19                        currency))
20    return expenses
```

FIGURE 1 – Image promotionnelle qui montre le fonctionnement de GitHub Copilot

L'IA est très efficace pour certains problèmes habituels ou notamment pour les tâches répétitives.

Initialement, il n'y avait pas vraiment d'alternative. Cependant, il existe maintenant plusieurs entreprises concurrentes qui ont développé leur propre produit :

- JetBrains, développeur des IDE IDEA, WebStorm, etc. propose son AI Assistant depuis décembre 2023 ¹
- Amazon CodeWhisperer, proposé dans le cadre d'Amazon Web Services
- Tabnine, par une entreprise indépendante

1. Présentation de JetBrains AI et de l'AI Assistant dans les IDE <https://blog.jetbrains.com/fr/blog/2023/12/06/presentation-de-jetbrains-ai-et-de-l-ai-assistant-dans-les-ide/>

2.2 Chatbots

Autrement que des outils dédiés, il est aussi possible de s'aider d'un Chatbot par IA, le plus célèbre étant ChatGPT. N'ayant pas le contexte du code, il faut apprendre à décrire ses problèmes. Cette solution est très populaire pour des problèmes « one shot », c'est-à-dire un problème sur lequel on bute, et sur lequel on aimerait de l'aide. On utilise l'IA pour aider à combler le un petit gap de connaissance occasionnel.

Cette solution est aussi très abordable : pas besoin d'installer un plugin sur un éditeur, juste d'un navigateur.

2.2.1 Problème de la sur-utilisation des chatbots

L'utilisation d'un outil d'IA pour combler un petit gap de connaissance est très pratique, et permet d'amener un regard extérieur au problème, de manière similaire à la méthode du canard en plastique².

Cependant, un autre problème peut se poser si le développeur utilise cette solution dans un domaine qu'il ne connaît que très mal, ou le gap de connaissance est trop grand.

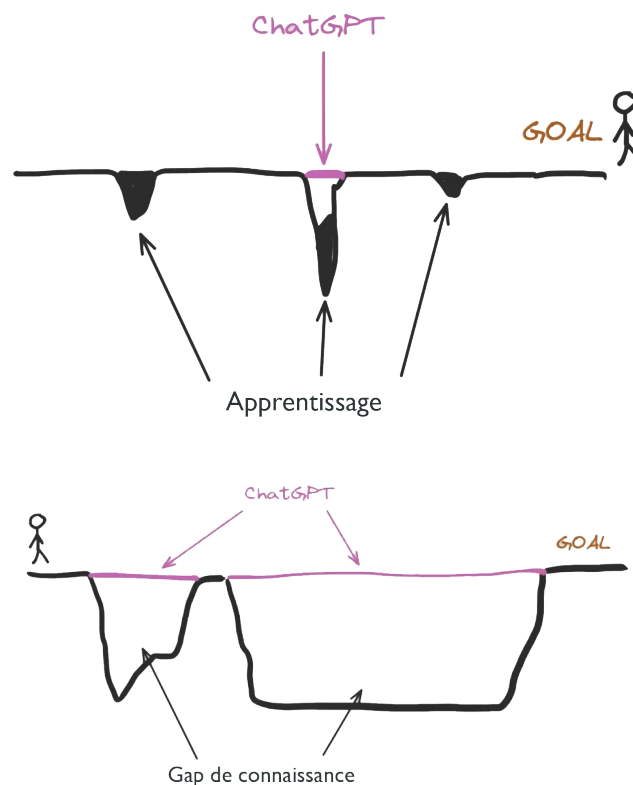


FIGURE 2 – Schéma du problème de l'utilisation de ChatGPT

2. Méthode du canard en plastique https://fr.wikipedia.org/wiki/M%C3%A9thode_du_canard_en_plastique

En figure 2, un schéma³ expliquant le problème.

Les chatbots, n'ayant pas le contexte de l'application, essayeront de générer une solution la plus générique possible. Cependant, elle sera parfois bien plus complexe que réellement nécessaire. C'est là où la connaissance aide, et là où le bat blesse si ces connaissances manquent.

2.3 Âge du modèle et solutions

Un grand problème des LLM subsiste : l'âge parfois avancé des données d'entraînement. En effet, si un modèle a été entraîné jusqu'à janvier 2022 (ChatGPT par exemple), il ne connaîtra aucune technologie disponible depuis cette date. En informatique notamment, l'évolution est rapide. Ainsi, le modèle risque plus d'inventer des fausses informations.

Il est très coûteux d'entraîner un modèle : il est nécessaire d'avoir du matériel très puissant, et très énergivore.

Selon une étude de Ben Cottier pour EpochAI⁴, le coût pour entraîner un modèle n'a fait qu'augmenter depuis 2009, et ce d'environ 0.5 ordre de grandeur par année. Cela signifie que le prix est multiplié par dix tous les deux ans.

Pour contrer ce problème majeur, il existe plusieurs solutions. Une des plus utilisée est d'autoriser le modèle à aller chercher sur Google et naviguer sur Internet à notre place pour aller chercher des informations. ChatGPT permet cela, mais un abonnement payant est nécessaire.

Hugging Face, une entreprise franco-américaine qui propose des outils et des ressources pour faciliter l'utilisation et le développement de transformeurs et de modèle. Ils ont aussi un produit similaire à ChatGPT, HuggingChat, qui permet d'utiliser plusieurs modèles différents (Mixtral par exemple, par l'entreprise française Mistral AI). La navigation sur Internet est disponible gratuitement.

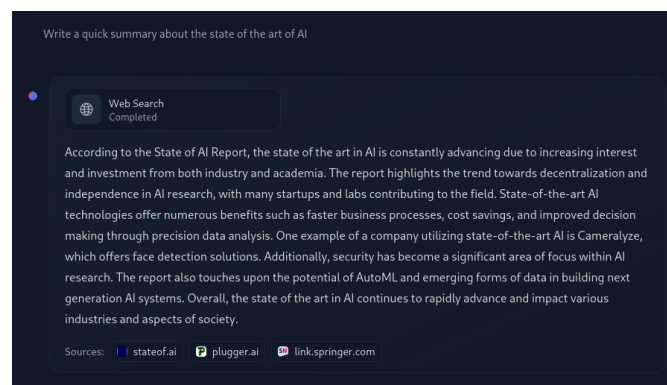


FIGURE 3 – Exemple d'un chatbot avec recherche sur Internet

3. Source du schéma « A cautionary tale about ChatGPT for advanced developers » https://www.youtube.com/watch?v=6CGtwF_5kzY

4. Trends in the Dollar Training Cost of Machine Learning Systems <https://epochai.org/blog/trends-in-the-dollar-training-cost-of-machine-learning-systems>

Des solutions plus ou moins spécialisés apparaissent, notamment les *RAG* (pour Retrieval-Augmented Generation, génération avec récupération améliorée, litt.). C'est un terme très récent, inventé par Patrick Lewis, un des auteurs principaux d'une sur ce sujet ⁵. Un article résumant le sujet ⁶ explique le principe de fonctionnement.

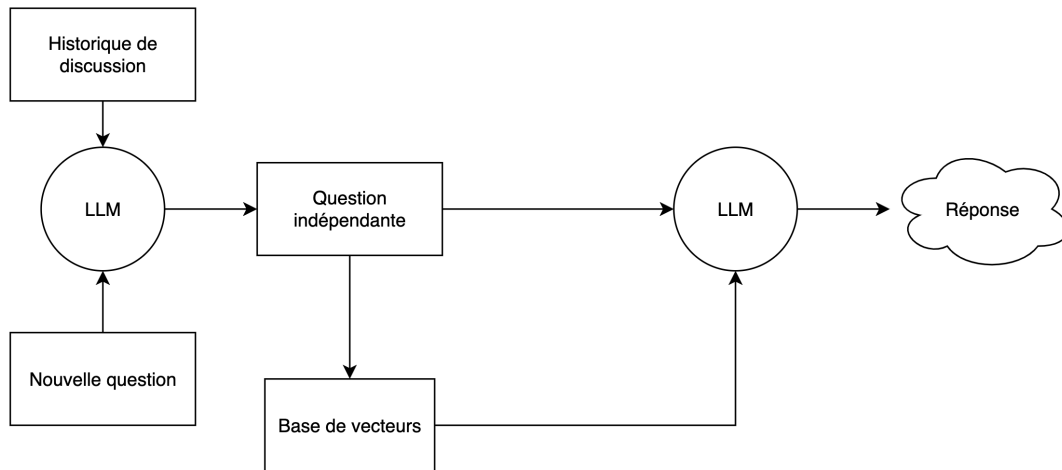


FIGURE 4 – Schéma de fonctionnement d'un RAG

3 Mais aussi : les moteurs de recherche « amélioré » par l'IA

Il existe un moteur de recherche pour développeurs logiciels qui utilise l'IA : Phind ⁷. Il utilise l'IA pour « comprendre » le contexte de la recherche de l'utilisateur, il fait en suite des recherches sur Internet pour trouver des sources et des exemples (afin d'éviter de créer des fausses informations), qu'il compile en une liste d'étapes ou de points clés, grâce à un LLM.

5. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks <https://arxiv.org/pdf/2005.11401.pdf>

6. What Is Retrieval-Augmented Generation, aka RAG? <https://blogs.nvidia.com/blog/what-is-retrieval-augmented-generation/>

7. Site web de Phind <https://www.phind.com/search>

The screenshot shows a search result on the Phind platform. The search query is "How to make calculate the best itinerary in a graph using NetworkX?". The result is titled "ANSWER | PHIND V10 MODEL" and provides a step-by-step guide. Step 1, "Create a Graph", involves initializing an empty graph object using `nx.Graph()`. Step 2, "Add Edges with Weights", involves adding edges to the graph with weights representing cost or distance. Step 3, "Find the Shortest Path", involves using the `nx.shortest_path()` function. Two code snippets are provided: one for creating the graph and adding edges, and another for finding the shortest path. The right sidebar shows "SOURCES" with links to NetworkX documentation and a Medium article.

phind Search Chat 🔍 ☰ Sign In

How to make calculate the best itinerary in a graph using NetworkX?

ANSWER | PHIND V10 MODEL

To calculate the best itinerary in a graph using NetworkX, you would typically want to find a route that minimizes cost or distance. Here's a step-by-step guide on how to achieve this:

- 1. Create a Graph:** First, initialize an empty graph object using `nx.Graph()`. This will be used to construct your network where nodes represent locations and edges represent paths between them.

```
import networkx as nx
G = nx.Graph()
```

- 2. Add Edges with Weights:** Add edges to the graph along with weights representing the cost or distance between the nodes. For example, if you are modeling a road network, the weight could be the distance or time taken to travel between cities.

```
G.add_edge("A", "B", weight=3)
G.add_edge("B", "C", weight=2)
G.add_edge("C", "A", weight=1)
```

- 3. Find the Shortest Path:** Use the `nx.shortest_path()` function to find the shortest path between two nodes based on the weight attribute. The `weight` parameter specifies the

Ask a followup question 🔍 →

SOURCES

- networkx.org > documentation > stabl...
Tutorial — NetworkX 3.2.1
documentation
Graph (edgelist) # create a graph from an edge list >>> list (H. edges ()) [(0, 1), (1, 2), (2, 3)] >>> adjacency_dict = {0:...
- networkx.org > documentation > stabl...
Shortest Paths — NetworkX 3.2.1
documentation
Compute the shortest paths and path lengths between nodes in the graph. These algorithms work with undirected and directed graphs. Advanced Interface # Shortest...
- medium.com > @tushar_aggarwal > n...
NetworkX: A Comprehensive Guide to Mastering Network ...
Oct 4, 2023 - NetworkX offers a wide range of graph analysis functions, allowing you to compute various graph properties and metrics. In this section, we...
- www.datacamp.com > tutorial > netw...
Python NetworkX for Graph Optimization Tutorial | DataCamp
NetworkX provides basic functionality for visualizing graphs, but its main goal is to

FIGURE 5 – Capture d'écran d'une recherche sur Phind

En figure 5 vous pouvez trouver un exemple de recherche avec Phind.

Ce moteur est une autre forme de solution au problème précédent. En allant chercher sur internet des sources à jour et fiables (avec un peu de chance).

4 Réellement utile ?

Une étude par GitClear⁸, une entreprise qui propose une plateforme code review sur Internet — donc à prendre avec des pincettes —, est sortie récemment, et remet en question le succès et voire le bien fondé de l'utilisation des assistants par IA pour le développement.

Ce rapport pose la question sur la nécessité d'écrire du « mauvais » code, certes plus vite, mais est-ce que le temps gagné à l'écriture ne sera pas perdu en réessayant de comprendre ce que fait le code plus tard ?

Selon leurs statistiques sur le code disponible sur GitHub, il y a eu une augmentation de la signification du « churn » (c'est à dire la part de code qui est écrit, puis édité ou supprimé moins de deux semaines après, dans cette étude).

8. Coding on Copilot <https://gitclear-public.s3.us-west-2.amazonaws.com/Coding-on-Copilot-2024-Developer-Research.pdf>

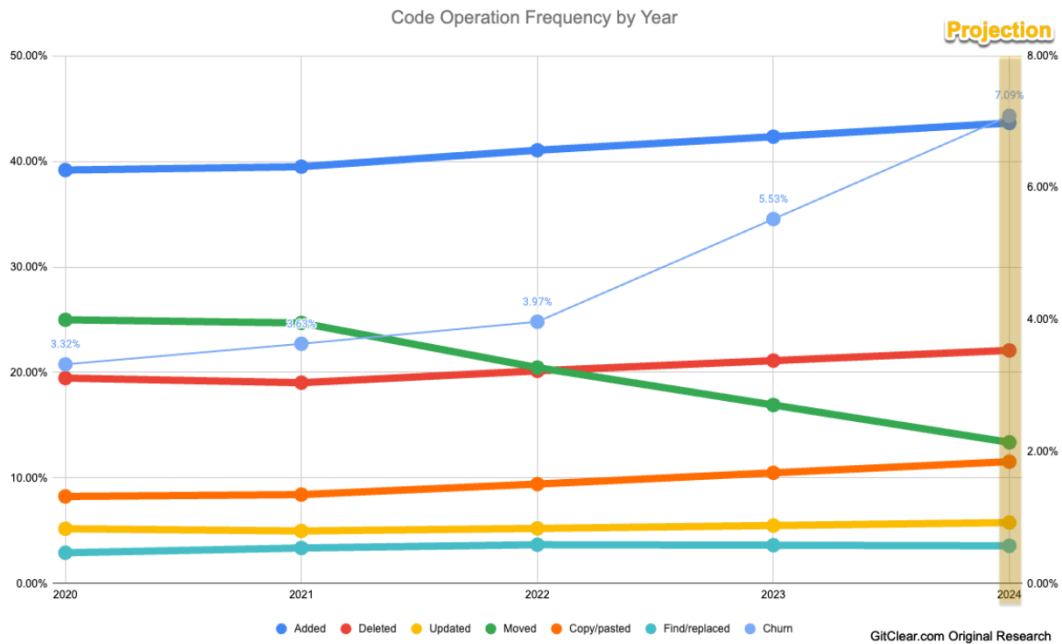


FIGURE 6 – Fréquence des opérations sur le code par année

En figure 6, on peut observer le pourcentage pour chaque type de changement à un code sur GitHub, par année. L'année 2024 est une estimation.

L'ajout en bleu étant évidemment le plus grand, avec 40% environ, en légère hausse depuis quelques années.

La ligne en bleu clair est celle qui nous intéresse : le code ajouté, puis modifié ou supprimé très vite après. De 3.3% en 2020, nous sommes à 5.5% en 2023, et GitClear estime que nous allons atteindre 7.1% en 2024.

De même, le temps passé à lire le code proposé avant de l'approuver ou non peut représenter un certain temps, voire un gêne.

5 Conclusion

En conclusion, l'utilisation de l'IA par les développeurs dans leur métier est grandissante. Bien qu'il soit certain que l'on peut développer bien plus vite avec l'aide de l'intelligence artificielle, il faut toujours traiter son entrée comme une aide, un outil, plutôt qu'une réponse universelle qui fonctionnera à coup sûr, vu qu'« une machine ne fait pas d'erreur ».

Ce serait aussi une erreur de se priver, ne serait-ce que par principe, d'un outil qui peut être très bien utilisé, ou très mal. Comme dit l'adage, « si vous avez un marteau, tout ressemble à un clou ».