3ème édition de la

Soirée du Test Logiciel Bordeaux

3 avril 2025

17h à 22h30

École de Turing, Bordeaux

Soirée du test logiciel Bordeaux

Conférence

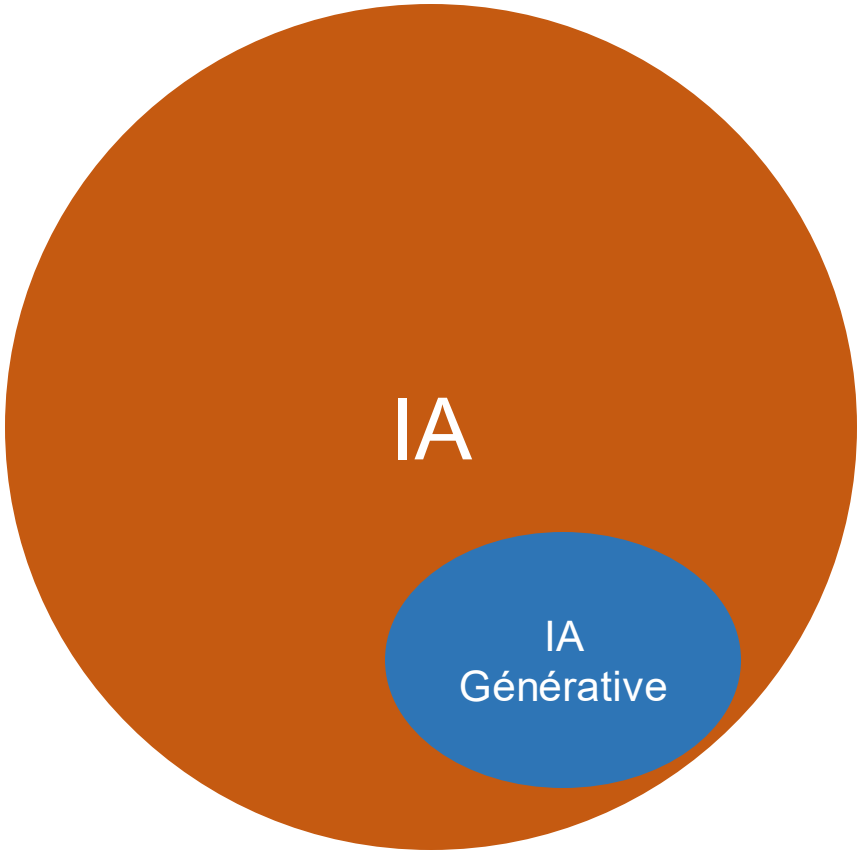
Évolution, fonctionnement et défis de l'IA Générative : comment le Test Logiciel en tire parti ?

Youssef TOUATI

Plan

1. Fonctionnement et Evolution de l'IA générative
2. IA Générative et le test
3. Défis de l'IA générative

Evolution de l'IA Générative



IA générative = un type d'IA

- Elle peut apprendre le langage humain, les langages de programmation, l'art, la chimie, la biologie ou tout autre sujet complexe.

- Elle réutilise ce qu'elle sait pour résoudre de nouveaux problèmes

Evolution de l'IA Générative

- **Avant 2017, les modèles d'IA générative reposaient principalement sur des architectures telles que :**
  - Les réseaux neuronaux récurrents (**RNN**, **LSTM**, **GRU**)
  - Les réseaux antagonistes génératifs (**GAN**)
  - Les auto-encodeurs variationnels (**VAE**)
- **Ces approches présentaient plusieurs obstacles techniques majeurs :**

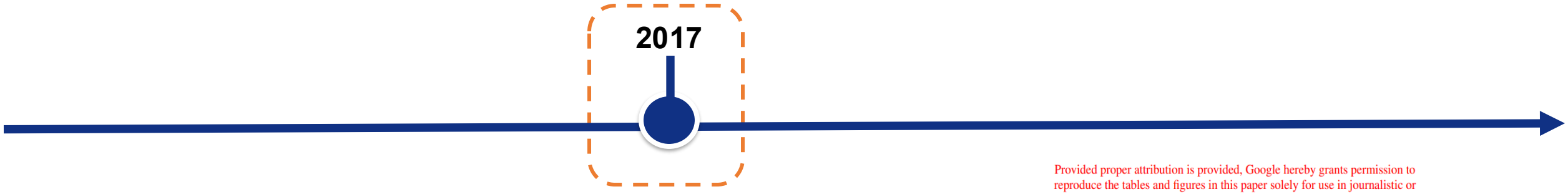
❌ Leurs mémoires est trop courte pour pouvoir traiter des paragraphes **trop longs**

❌ Traitent les données de façons séquentielle, et sont donc **difficilement parallélisable**

⇨ Entraîner un modèle de LSTM en utilisant les données consommées par GPT-3 aurait pris des décennies

Evolution de l'IA Générative

Conférence



L'année 2017 marque un tournant majeur avec l'introduction du modèle **Transformer** dans leur célèbre article ***"Attention is All You Need"***.
Cette nouvelle architecture propose :

- Un mécanisme innovant appelé attention, en particulier **l'auto-attention (self attention)**
- Un traitement **parallèle** des données, permettant une accélération significative de l'entraînement
- Une meilleure gestion des **dépendances** longues et du **contexte** global

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including

Evolution de l'IA Générative

2017

Après 2017

Après l'arrivée des Transformers, une nouvelle génération de modèles d'IA générative voit le jour

ChatGPT

DALL·E

Google

Les **Transformers** ont posé les **fondations techniques**, mais ce sont les **LLMs** qui ont véritablement démocratisé l'IA générative et transformé notre façon d'interagir avec la technologie.

Un Large Language Model (LLM), ou grand modèle de langage, est une forme avancée d'intelligence artificielle conçue pour **traiter**, **comprendre** et **générer** du langage naturel.

Exemples de LLM: GPT, Claude, Bert

Comment un Modèle de Langage Génère une Réponse ?

Exemple: Poser une question à ChatGPT

1 Tokenization (Découpage du texte en tokens) ✅

Le transformer commence par **découper** le texte en tokens.

⇨ décomposer des phrases complexes en éléments plus **simples** à **analyser**

- **Question originale** : "Pourquoi les tests logiciels sont-ils essentiels dans le cycle de développement ?“

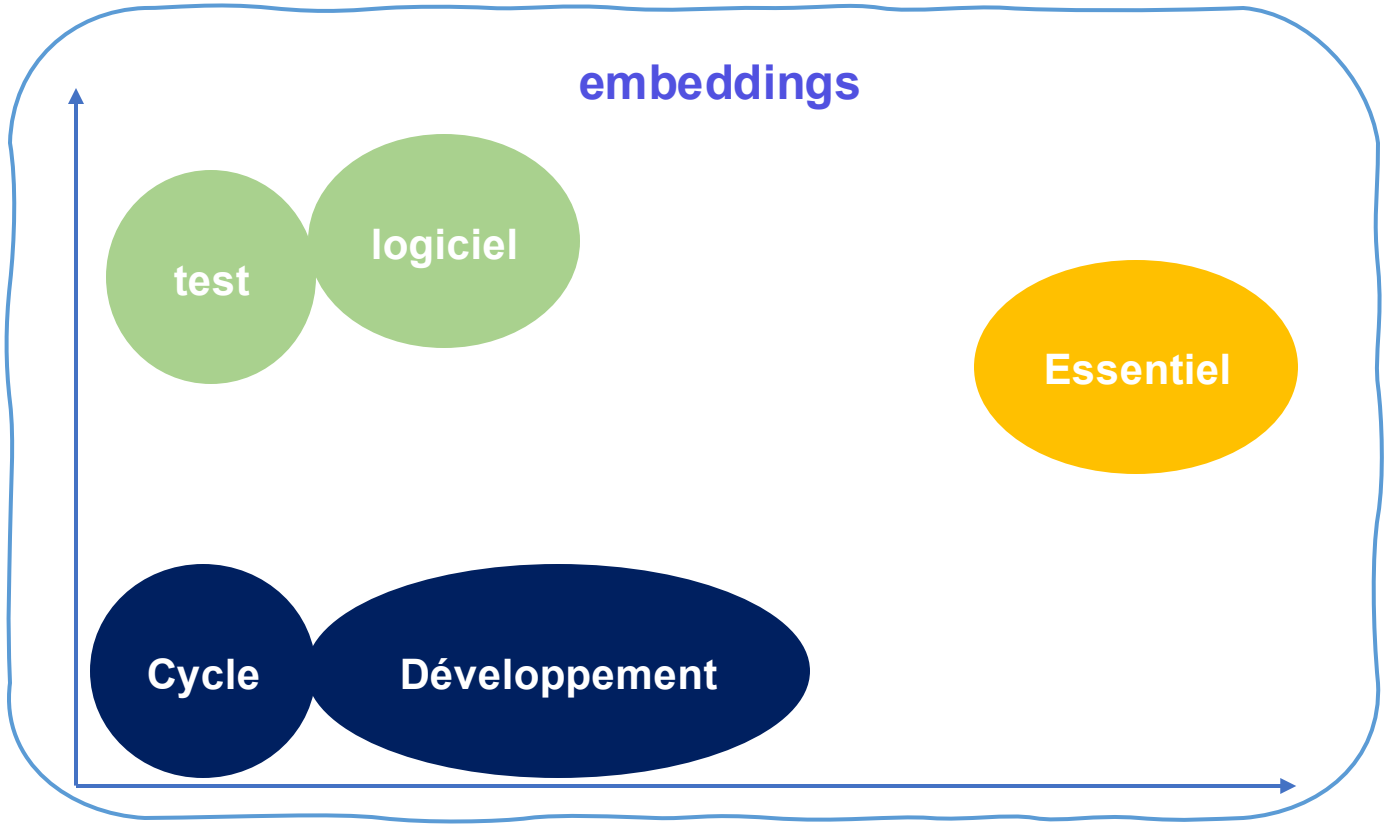
- **Après tokenization** :

```
Pourquoi les tests logiciels sont-ils essentiels dans le cy
cle de développement ?
```

🔷 **But** : Convertir le texte en une structure que le modèle peut analyser.

2 Transformation en embeddings (Représentation numérique) ✅

Chaque token est transformé en **vecteur numérique** dans un espace sémantique.
Les mots **similaires** sont placés **proches** dans cet espace.



📌 Exemple :

- "**tests**" et "**logiciels**" seront proches car ces concepts sont liés.
- "**développement**" sera proche de "**cycle**", mais loin de "**essentiels**"

🔷 **But : Permettre au modèle de capturer la relation entre les mots**.

3 Analyse du contexte et relations entre tokens (Self-Attention) ✅

Le modèle analyse les **liens entre les mots** pour comprendre le contexte et détermine les mots les plus **importants** dans la phrase.

“tests logiciels”	"cycle de développement"	Pourquoi
est une unité de sens forte (concept clé)	est un élément central du contexte	indique que l’utilisateur attend une explication causale

- But : Identifier la structure et le sens de la question.

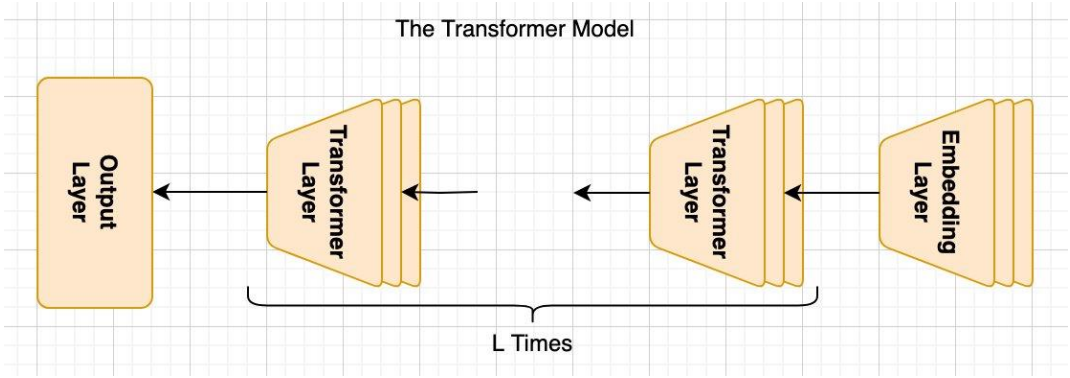
4 Passage dans plusieurs couches de neurones (Transformer Layers) ✅

✔ Chaque token **traverse plusieurs couches neuronales** pour **affiner** sa signification.

📌 Exemple :

- **Après plusieurs passes**, le modèle comprend que les tests logiciels servent à assurer la qualité et prévenir les erreurs.
- Il détecte que la réponse doit inclure des **avantages** et **objectifs** des tests.

🔷 **But : Structurer une réponse logique basée sur les relations entre concepts.**



5 Génération d'une réponse (Prédiction du token suivant) ✅

Les LLMs utilisent les informations traitées par les transformers pour prédire les tokens suivants et générer une réponse complète :

📌 **Exemple** :

1. "Les"
2. "tests"
3. "logiciels"
4. "permettent"
5. "d'assurer"
6. "la"
7. "qualité"
8. "du"
9. "logiciel"
10. "et"
11. "de"
12. "réduire"
13. "les"
14. "risques"
15. "d'erreurs."

🔷 **But :**

Générer une réponse naturelle et cohérente en respectant le contexte.

6 Post-traitement et Vérification de cohérence ✅

- ✔ L’IA vérifie si la réponse est claire et grammaticalement correcte.
- ✔ Certains modèles appliquent des **filtres** pour **éviter les biais** ou informations **incorrectes**.

Résumé

Étape	Explication	Exemple (Pourquoi les tests logiciels sont essentiels ?)
1. Tokenization	Découpe en tokens	["Pourquoi", "les", "tests", "logiciels", "sont", "essentiels", "?"]
2. Embeddings	Conversion en vecteurs	"tests" et "logiciels" sont proches en sens
3. Self-Attention	Analyse des liens entre mots	"tests logiciels" est un concept clé, lié à "qualité" et "erreurs"
4. Passage dans les couches	Transformation progressive	Détection de l'importance des tests pour éviter les bugs
5. Génération des tokens	Prédiction mot par mot	"Les tests logiciels permettent d'assurer la qualité..."
6. Post-traitement	Vérification et reformulation	Réponse optimisée et plus fluide

Malgré leur puissance, les LLMs classiques rencontrent plusieurs défis ..

Connaissance statique

Une fois entraînés, **ils ne peuvent pas apprendre de nouvelles informations** sans une nouvelle phase d'entraînement coûteuse.

→ **Informations obsolètes ou inexactes**

⇩

RAG

Manque de spécialisation

•Un modèle pré-entraîné possède des connaissances générales mais peut ne pas être optimisé pour un **domaine précis** (médecine, finance, droit, etc.).

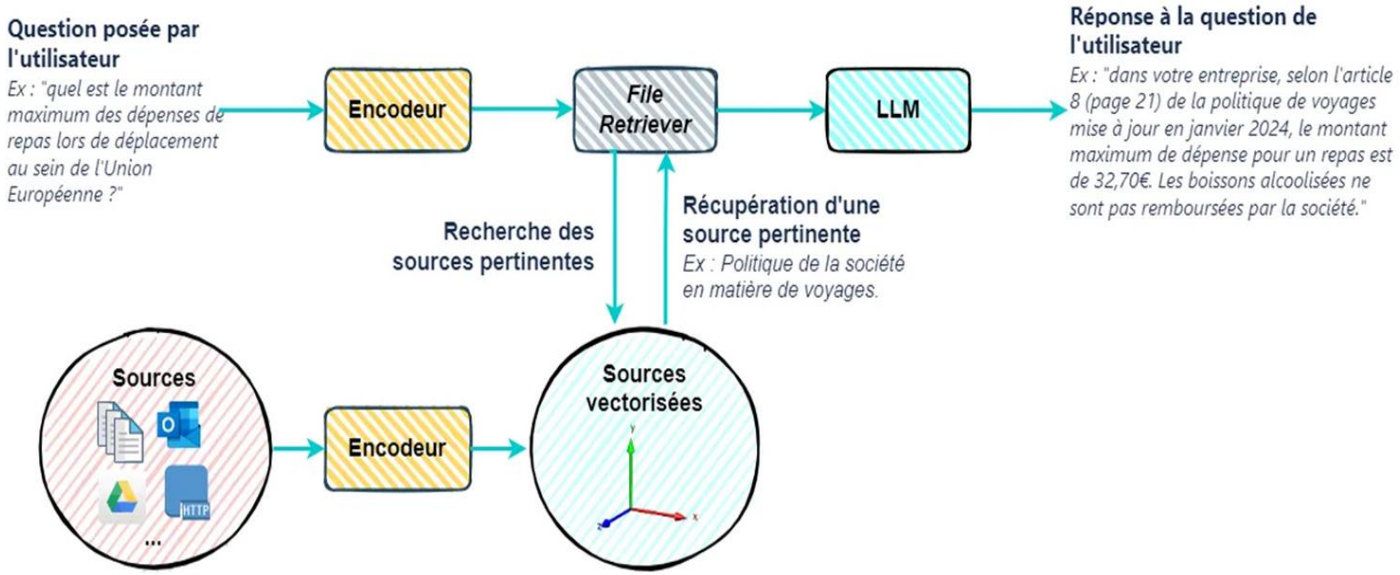
•Il peut générer des réponses trop vagues pour des cas d'usage nécessitant une **expertise pointue**.

⇩

Fine tuning

RAG : Une Solution Pour des Réponses Plus Fiables

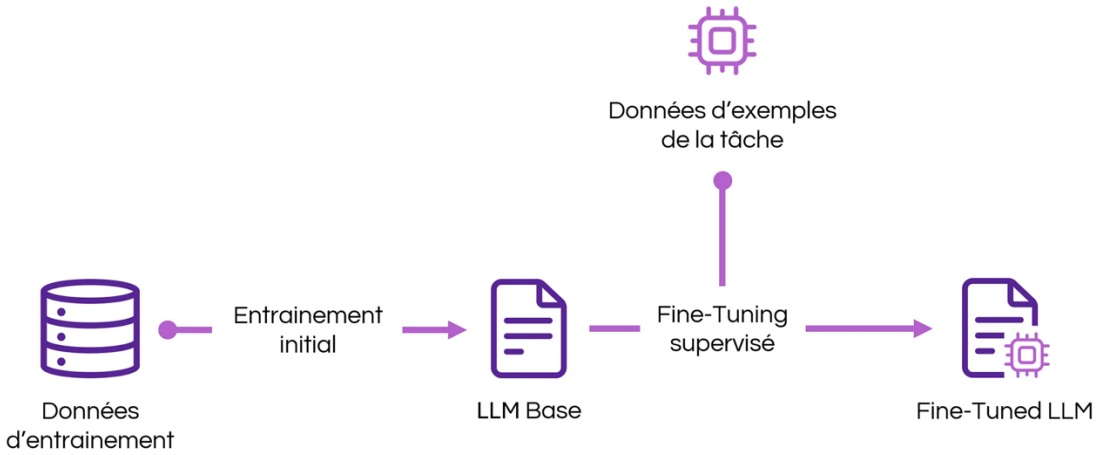
•Plutôt que de se baser uniquement sur **leurs connaissances internes**, le modèle **RAG (Récupération-Augmentée par Génération**) intègre une base de **données externe** ou un **moteur de recherche** pour enrichir leurs réponses avec des **informations actualisées et pertinentes**.



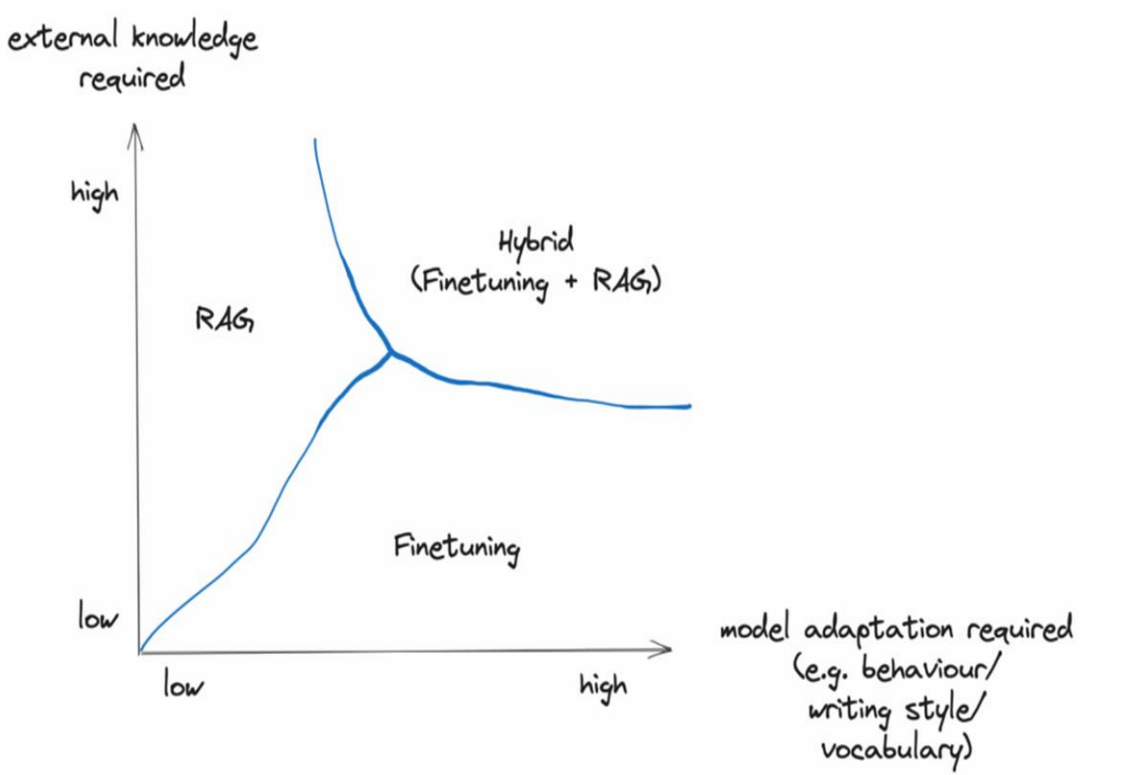
Fine tuning : Une Solution Pour des Réponses Plus ajustées

•**Fine-tuning** : Il s'agit du processus qui consiste à prendre un LLM **pré-entraîné** et à le **réentraîner** sur un ensemble de données plus petit et **spécifique** afin de l'adapter à **une tâche particulière** ou d'améliorer ses performances.

Exemples : documents juridiques, dossiers médicaux, bases de connaissances internes d'une entreprise.



RAG VS Fine tuning



Récapitulatif rapide 💡 :

➡️ Besoin d'accéder à des connaissances externes ? → **RAG**
➡️ Besoin d'un alignement rédactionnel précis ? → **Fine-tuning**

Comment communiquer avec IA … ?

Via **API** ou via **Interface graphique (GUI)**

GUI

Permettre aux utilisateurs d'interagir **directement** avec l'IA via une **application** web ou mobile.

⟹ Interface intuitive et conviviale.

⟹ Idéal pour les tâches ponctuelles ou créatives.

API

Intégrer directement les fonctionnalités de l'IA dans vos propres applications ou systèmes

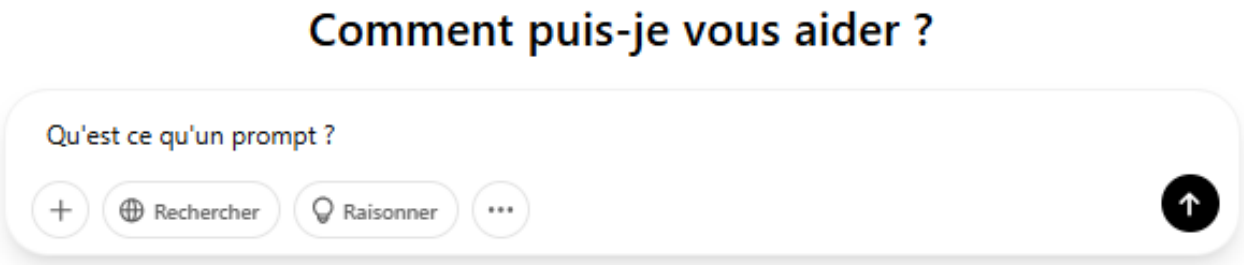
⟹ **Intégration** dans des workflows existants.

⟹ Personnalisation **avancée**.

Que ce soit via l'API ou l'interface graphique (GUI), les bonnes pratiques de création de prompts sont indispensables pour interagir efficacement avec l'IA.

Qu’est-ce qu’un prompt ?

Un prompt est une instruction ou une question donnée à un modèle d’IA pour obtenir une réponse ou exécuter une tâche.



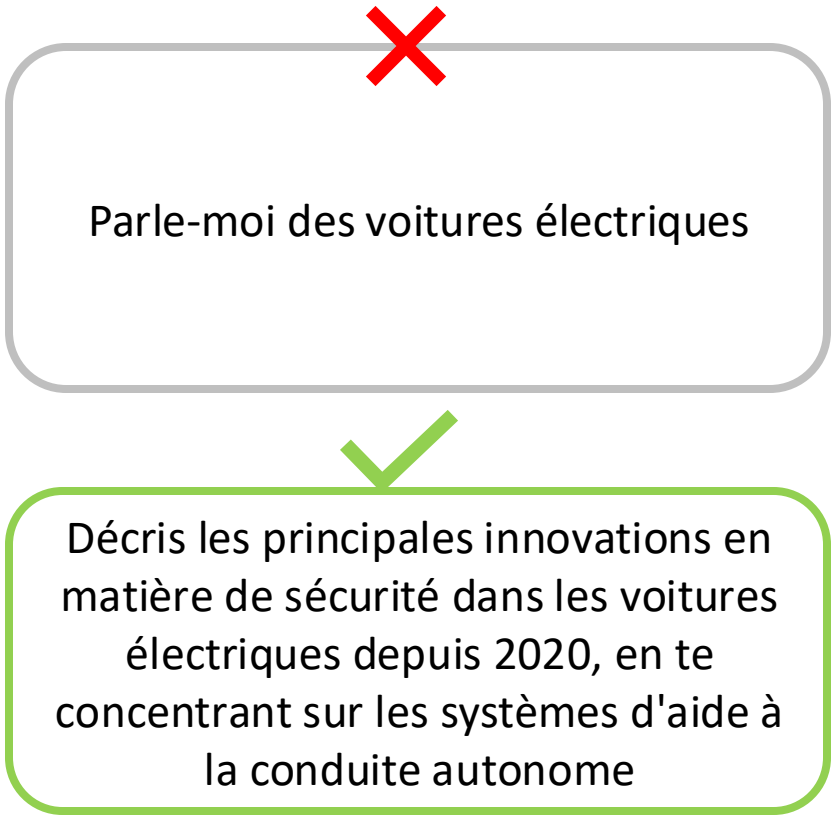
Soirée du test logiciel

Conférence

Comment créer un prompt efficace pour ChatGPT .. ?

1. Soyez précis

Fournissez des instructions détaillées et claires



2. Donnez au modèle de l’espace pour réfléchir

Encouragez le modèle à raisonner étape par étape avant de fournir une réponse finale

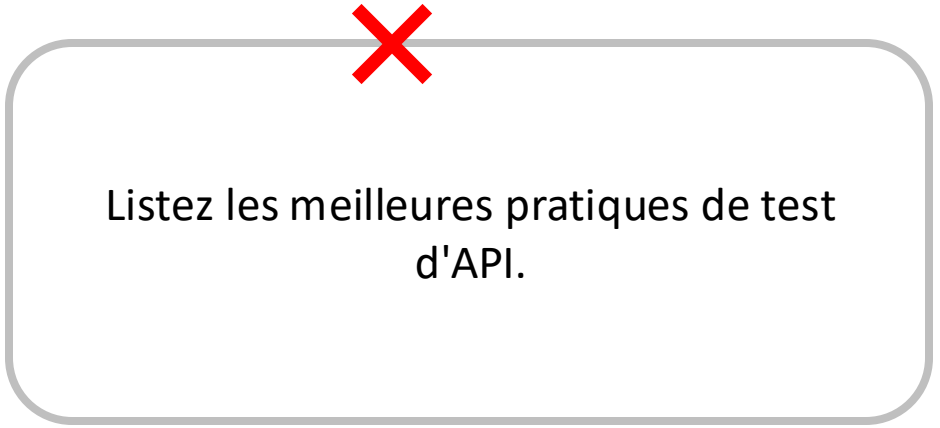
Au lieu de lancer une demande globale et d'espérer que l'IA comprenne vos attentes, vous lui donnez une feuille de route.

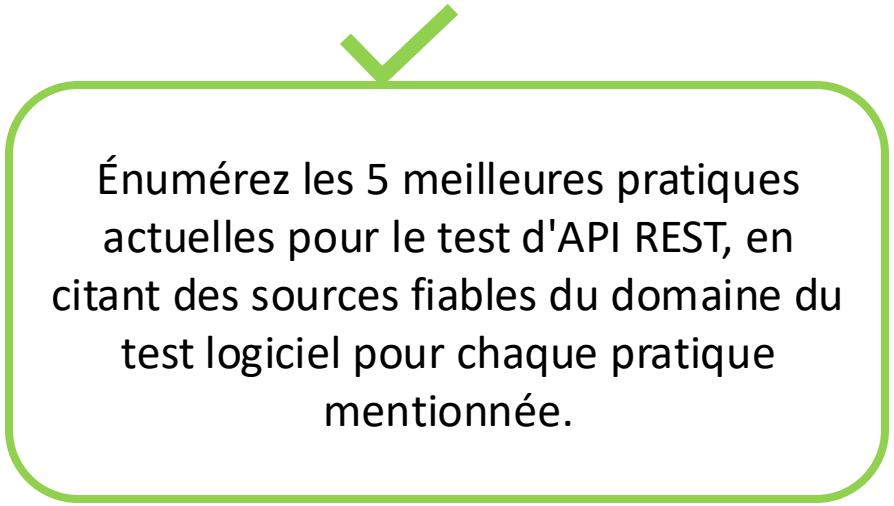
Est-ce que l'intelligence artificielle va remplacer les humains ?

Analysons étape par étape les implications potentielles de l'intelligence artificielle sur l'emploi humain. Considère les avantages, les inconvénients et les domaines susceptibles d'être les plus affectés avant de conclure

3. Citez les sources

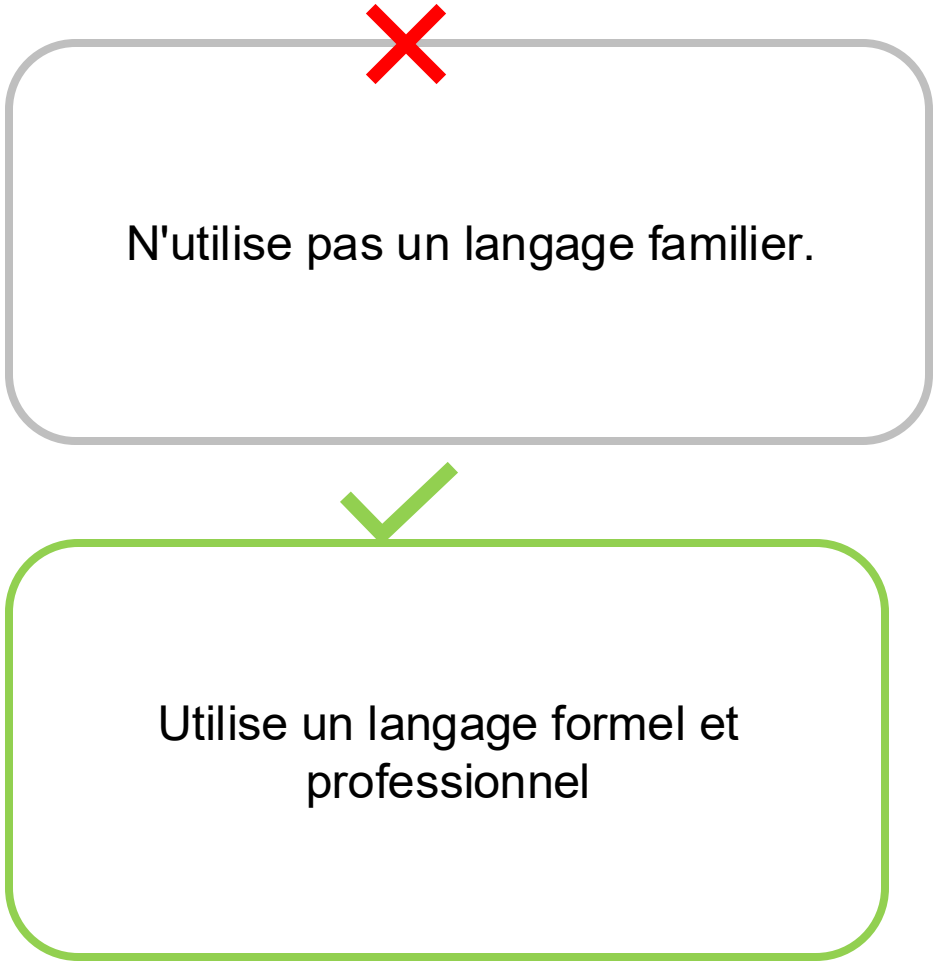
Demander au modèle de citer ses sources peut réduire le risque d’hallucinations.





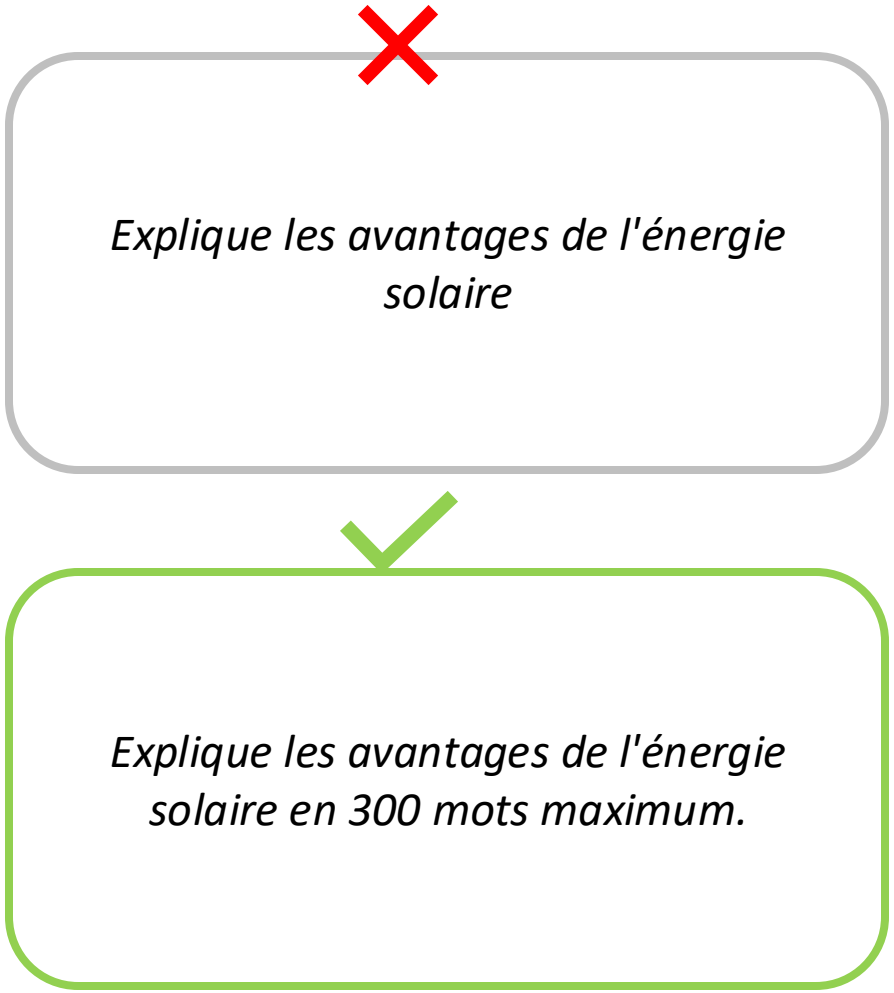
4. Instructions positives

•Concentrez-vous sur ce que le modèle *doit faire* plutôt que sur les restrictions.



5. Cadrage temporel et format

- Indiquez clairement les contraintes :
  - Longueur de la réponse (*"En 300 mots maximum"*)
  - Structure souhaitée (*"En tableau Markdown avec en-têtes"*)



Soirée du test logiciel

Conférence

utiliser l'IA pour créer des prompts

AI-Powered Prompt Writer

Base Prompt*

Looking for a tool for content creation? Try out our Blog Post Generator to create SEO-rich content effortlessly.

Je suis testeur logiciel je veux savoir comment éviter les flaky tests

12/200 words

Generate

Junia

Results

788 characters · 122 words

En tant que testeur logiciel, je cherche des conseils détaillés et des meilleures pratiques pour gérer et éliminer les tests instables (flaky tests) dans mes suites de tests automatisés. Veuillez inclure :

- Les causes communes des tests instables (problèmes de timing, dépendances externes, état des données)
- Les stratégies concrètes pour identifier et isoler les tests flaky
- Les techniques de conception de tests robustes
- Les bonnes pratiques d'implémentation pour réduire l'instabilité
- Des exemples de refactoring de tests instables
- Les outils et frameworks recommandés pour la détection et la gestion des tests flaky
- Les métriques à suivre pour évaluer la stabilité des tests
- Les stratégies de documentation et de communication autour des tests instables dans une équipe

Copy

Le prompt type contient 6 parties :

Partie	Exemple
Rôle	Agis comme un expert QA spécialisé dans les tests d'applications mobiles
Contexte	Nous travaillons sur une application de paiement mobile permettant aux utilisateurs de transférer de l'argent
Instructions	Génère 5 cas de test pour vérifier la fonctionnalité de validation 3D-Secure lors du paiement en ligne
Contraintes	Les cas de test doivent être écrits au format Gherkin avec les étapes 'Étant donné', 'Quand', et 'Alors'
Format	Présente les cas sous forme de tableau Markdown avec ID, Titre, Étapes et Résultat attendu
Données	Utilise des spécifications techniques fournies et inclut des données fictives réalistes pour simuler les transactions

Soirée du test logiciel

Conférence

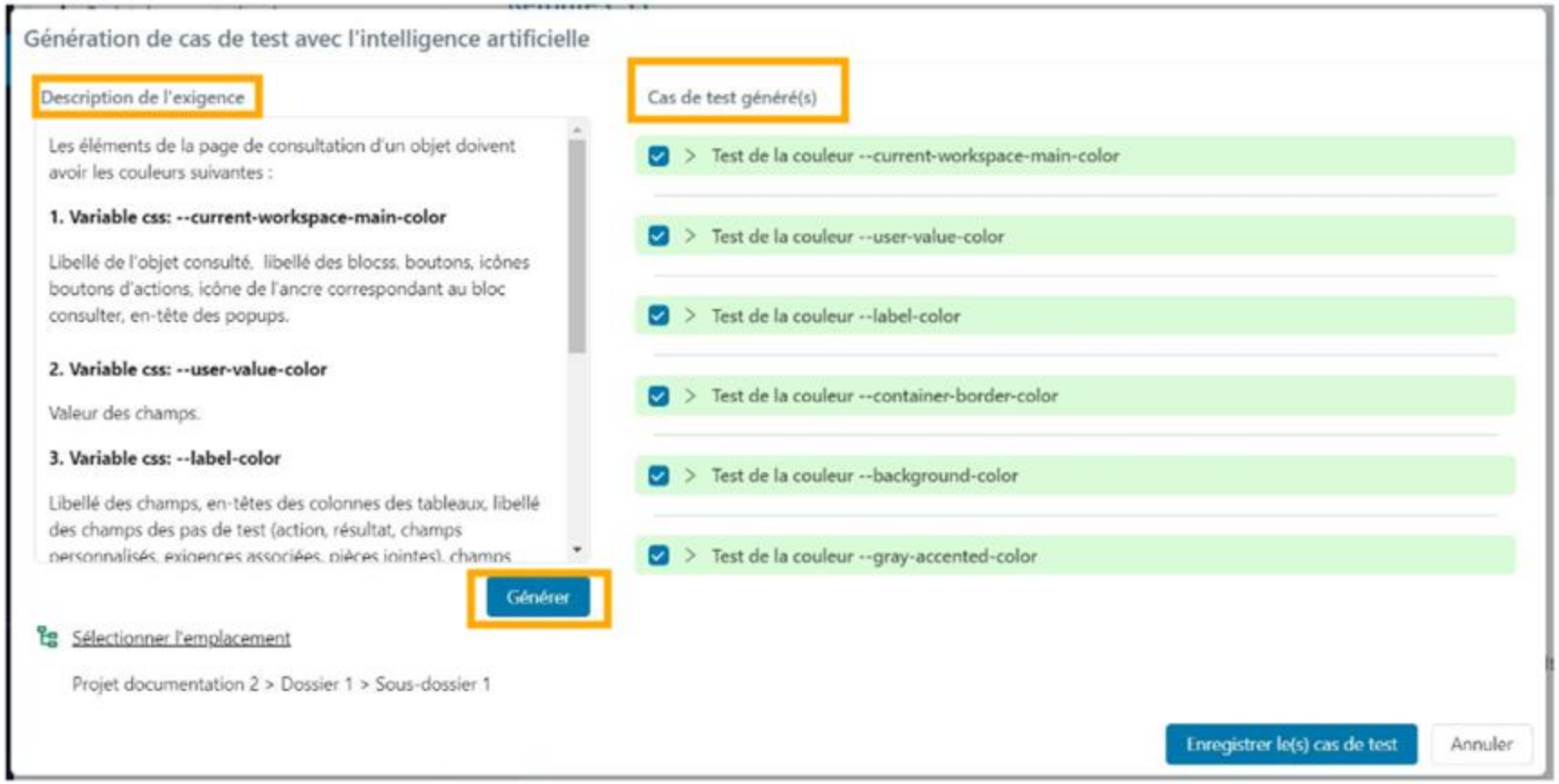
Exemples d'utilisation de l'IA Générative dans le domaine du test logiciel ..

Soirée du test logiciel

Conférence

Automatisation de la rédaction de cas de test

L'IA permet de transformer automatiquement des spécifications fonctionnelles ou des user stories en cas de test détaillés, éliminant la rédaction manuelle.



Squash TM

Avantages clés :

1.Gain de temps :
Génération instantanée de scénarios multiples.

2.Couverture exhaustive :
Inclusion de cas limites (ex. : champs vides, formats incorrects

3.Réduction des erreurs :
Suppression des oublis ou biais humains.

Création de données de test réalistes

L'IA générative transforme la création de jeux de données pour les tests logiciels :

Simulation réaliste	Diversité et volume	Tests approfondis
Génère des données **imitant** fidèlement le **monde réel**	Produit de vastes ensembles de **données variées**	Inclut les cas **limites**

Soirée du test logiciel

Conférence

Automatisation de l’analyse des logs d’erreur

L'IA générative révolutionne l'analyse des logs d'erreurs logiciels :

Avantages	Lecture et catégorisation rapide des logs
	Réduction significative du temps d'analyse
	Équipes concentrées sur la résolution plutôt que l'analyse
Fonctionnalités	Suggestion de **causes probables** et **solutions potentielles**

Démo Fichier Logs

Création de scripts de test automatisés

L'IA transforme la création de tests automatisés en simplifiant le processus :

Fonctionnement	Génération de scripts à partir de descriptions simples
Avantages	**Accessibilité** : Tests automatisés pour non-programmeurs
	Gain de temps : Réduction du temps de développement des tests
	Flexibilité : Adaptation facile aux différents parcours utilisateurs

Tester la fonctionnalité de connexion. L'utilisateur entre un nom d'utilisateur et un mot de passe valides, clique sur le bouton de connexion, et devrait être redirigé vers la page d'accueil

(En selenium)

```python
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

# Initialiser le navigateur
driver = webdriver.Chrome()
driver.get("https://example.com/login")  # Remplacer par l'URL réelle

# Entrer le nom d'utilisateur
username_field = driver.find_element(By.ID, "username")
username_field.send_keys("utilisateur_valide")

# Entrer le mot de passe
password_field = driver.find_element(By.ID, "password")
password_field.send_keys("mot_de_passe_valide")

# Cliquer sur le bouton de connexion
login_button = driver.find_element(By.ID, "login-button")
login_button.click()

# Attendre et vérifier la redirection vers la page d'accueil
WebDriverWait(driver, 10).until(
    EC.url_contains("/home")
)

# Vérifier un élément spécifique de la page d'accueil
assert "Bienvenue" in driver.page_source

# Fermer le navigateur
driver.quit()
```

Synergie Humain et IA ?

Conférence

ChatGPT and Human Synergy in Black-Box Testing: A Comparative Analysis

Hiroyuki Kirinuki
hiroyuki.kirinuki@ntt.com
NTT Software Innovation Center
Tokyo, Japan

Haruto Tanno
haruto.tanno@ntt.com
NTT Software Innovation Center
Tokyo, Japan

Applications testées

1- Vérificateur de robustesse de mot de passe

2- Convertisseur d'unités

3- Planificateur de budget

Participants

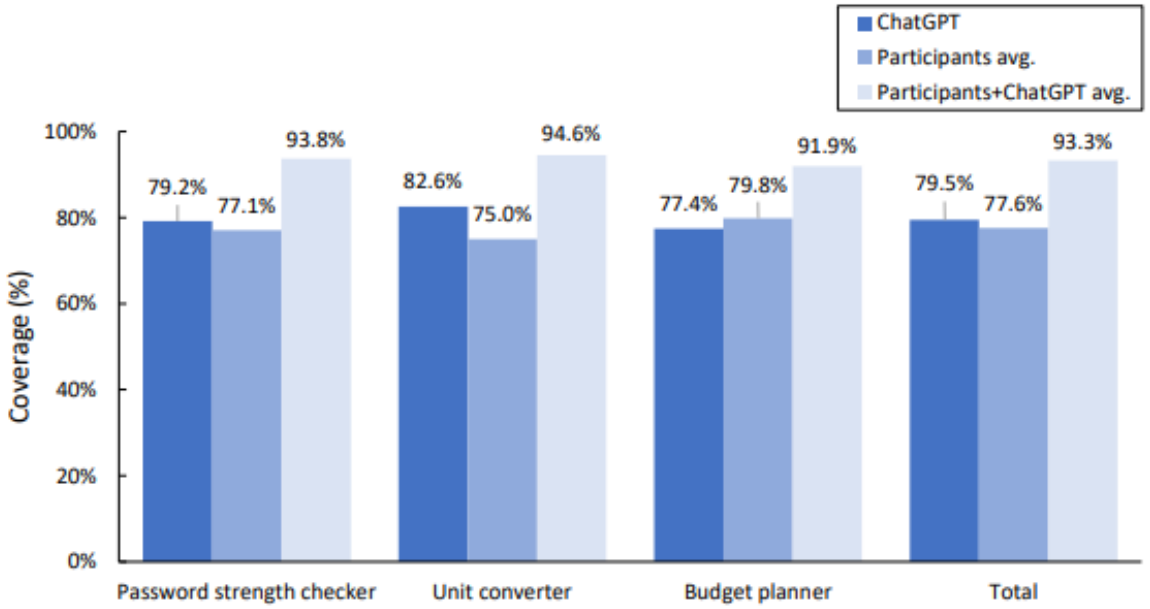
- **ChatGPT** (GPT-4)

- **Quatre** testeurs humains

Critères d'évaluation

- **Taux de couverture** des tests

Résultats:



Performance comparative :
ChatGPT a couvert autant, voire plus, de points de vue que les humains dans deux des trois applications.

Temps requis :
Les participants humains ont pris en moyenne **198 minutes** pour concevoir leurs suites de tests

Limites identifiées pour ChatGPT :
Incohérences occasionnelles entre les descriptions des cas, les entrées et les résultats attendus.

Points Clés

- La collaboration entre humains et ChatGPT a permis une couverture significativement plus large (93,3 % des points de vue effectifs)
- Nécessité d'une supervision humaine pour combler les lacunes (valeurs limites, cohérence)
- ChatGPT peut générer rapidement des cas de test comparables à ceux créés par des humains

Défis majeur de l'IA générative

L'IA peut être… menteuse (et avec confiance en plus !)

Elle a tendance à "meubler".

Elle répond à tout, même si c'est n'importe quoi.

💡 **Morale de l'histoire** : L'IA générative est un excellent outil pour accélérer nos projets, mais son travail demande toujours une validation humaine.

En somme, elle est l'assistant, pas le chef.

Dans un message publié sur X, Grok, le chatbot d'Elon Musk, a accusé à tort la star de la NBA **Klay Thompson** d'avoir **jeté des briques** dans les **fenêtres de plusieurs maisons**.

Klay Thompson Accused in Bizarre Brick-Vandalism Spree

In a bizarre turn of events, NBA star Klay Thompson has been accused of vandalizing multiple houses with bricks in Sacramento. Authorities are investigating the claims after several individuals reported their houses being damaged, with windows shattered by bricks. Klay Thompson has not yet issued a statement regarding the accusations. The incidents have left the community shaken, but no injuries were reported. The motive behind the alleged vandalism remains unclear.

Grok is an early feature and can make mistakes. Verify its outputs.

Jeter des briques
=
tirs ratés ou maladroits

Pour aller plus loin :

LLM Visualization

Site interactif pour comprendre facilement comment marchent les modèles de langage (LLM).

Références:

Transformer Architecture: Key Components, Use Cases & Future

Transformer AI : la révolution dans le traitement du langage naturel

Getting started with prompts for text-based Generative AI tools | Harvard University Information Technology

Comment communiquer? - Faire un prompt - Université Numérique – UNIGE

Comment créer des prompts efficaces : guide du débutant - francenum.gouv.fr

Getting started with prompts for text-based Generative AI tools | Harvard University Information Technology

Écrire des cas de test avec l'aide de l'IA - Documentation Squash TM

Les Modèles Génératifs (Partie 3) | Les Transformers | Hugo Michel

3ème édition de la

Soirée du Test Logiciel Bordeaux

3 avril 2025

17h à 22h30

École de Turing, Bordeaux

Soirée du test logiciel Bordeaux

Merci de votre écoute !