Atelier (3h) : Créez votre premier agent IA avec OpenAI et Agno

Objectif

Apprendre à concevoir, développer et exécuter un agent IA en utilisant OpenAI SDK et Agno/Phidata.

Format

- Présentation rapide
- Démonstrations en direct
- Activités pratiques
- Questions-réponses

Matériel requis :

- Ordinateurs portables avec Python 3.11+ installé.
- Accès à Internet stable.
- IDE (recommandé : VS Code).
- Clé API OpenAI active (les participants doivent en créer une à l'avance si possible).
- Bibliothèques Python installées : Voir le fichier Requirements.txt
- Accés au github repo préparés pour les démonstrations et l'activité pratique.

Note Importante : Cet atelier constitue une base solide pour la création d'agents IA. Les participants sont encouragés à explorer les ressources fournies et à continuer à expérimenter pour développer des agents plus sophistiqués et adaptés à leurs besoins spécifiques.

Introduction et Bases des Agents IA

1.0 Définition des concepts: LLM, Token, RAG etc

Concepts Fondamentaux des LLMs et des Agents IA pour votre Workshop

Large Language Models (LLMs)

Un **Large Language Model (LLM)** est une IA spécialisée dans la compréhension et la génération de langage naturel grâce à des réseaux de neurones profonds. Ces modèles, qualifiés de "larges", sont entraînés sur d'immenses corpus de données et possèdent un grand nombre de **paramètres** (ex. GPT-3 avec 175 milliards), leur permettant d'apprendre et de générer du texte de manière cohérente.

Les LLMs sont des **modèles de fondation**, pré-entraînés sur des données non étiquetées en auto-supervision, ce qui leur permet d'identifier des motifs et d'effectuer diverses tâches de traitement du langage naturel. Leur architecture repose principalement sur des **réseaux de neurones Transformers**, qui analysent le contexte des mots dans une séquence pour mieux en comprendre le sens.

L'**entraînement** d'un LLM consiste à prédire le mot suivant dans une phrase, en ajustant progressivement ses paramètres pour améliorer ses prédictions. Une fois pré-entraîné, le modèle peut être **affiné (fine-tuning)** sur un corpus spécifique pour se spécialiser dans une tâche particulière.

Architecture et fonctionnement

Les LLMs sont composés de multiples couches de réseaux neuronaux comprenant des couches récurrentes, des couches feedforward, des couches d'embedding et des couches d'attention qui travaillent ensemble pour traiter le texte d'entrée et générer des réponses cohérentes. Cette architecture complexe leur permet de produire du texte et d'interpréter le langage avec une finesse qui imite la fluidité humaine.

Modèles de Fondation

Un modèle de fondation est un modèle d'intelligence artificielle de grande taille, entraîné sur une quantité massive de données non étiquetées, généralement par apprentissage auto-supervisé. Ces modèles peuvent être adaptés à un large éventail de tâches en aval sans nécessiter une reprogrammation complète.

Les modèles de fondation représentent un paradigme pour la construction de systèmes d'IA, dans lequel un modèle pré-entraîné sur une grande quantité de données non étiquetées peut être adapté à de nombreuses applications. Les premiers modèles de fondation étaient de grands modèles de langage basés sur l'architecture des "Transformeurs", notamment BERT et GPT-3.

Évolution et caractéristiques

Les modèles de fondation sont caractérisés par deux aspects clés : l'apprentissage par transfert et la capacité à générer des connaissances de référence qui peuvent être modifiées pour des tâches spécifiques. Ce terme a été popularisé par le centre de recherche sur les modèles de fondation (CRFM) du Stanford Institute for Human-Centered Artificial Intelligence (HAI).

Tokens

Les tokens sont les unités fondamentales de traitement du texte dans les LLMs. Un token peut représenter un mot complet, une partie de mot, un caractère, ou un symbole de ponctuation, selon le système de tokenisation utilisé par le modèle. La tokenisation est le processus qui consiste à découper le texte en ces unités plus petites que le modèle peut traiter.

Les LLMs ont généralement une limite de contexte exprimée en nombre de tokens (par exemple, 2048, 4096 ou 8192 tokens). Cette limite détermine la quantité de texte que le modèle peut traiter en une seule fois, influençant sa capacité à maintenir la cohérence sur de longs passages.

Retrieval Augmented Generation (RAG)

Le RAG (Retrieval Augmented Generation) est une technique qui améliore les capacités des LLMs en les augmentant avec des connaissances externes, comme des bases de données ou des documents. Cette approche fonctionne en récupérant un ensemble de documents pertinents à partir d'une source externe, puis en concaténant ces documents avec l'invite originale avant de les envoyer au générateur de texte.

Cette méthode permet aux LLMs d'accéder aux informations les plus récentes sans nécessiter un réentraînement du modèle, ce qui est particulièrement utile puisque la connaissance paramétrique des LLMs est statique. Le RAG aide également à réduire les problèmes d'hallucination (génération de fausses informations) et améliore les performances dans des environnements en évolution rapide.

Embeddings

Les embeddings textuels, également appelés word embeddings, sont des représentations numériques de données textuelles où chaque mot est représenté sous forme d'un vecteur de nombres réels. Ces représentations vectorielles permettent aux algorithmes d'apprentissage automatique de comprendre et de traiter efficacement le langage humain.

Il existe deux techniques principales pour créer des embeddings textuels :

1. Les embeddings basés sur la fréquence, qui utilisent la fréquence des mots pour créer leurs représentations vectorielles.
2. Les embeddings basés sur la prédiction, qui capturent les relations sémantiques et les informations contextuelles, fournissant des représentations riches des concepts et des relations sémantiques.

Les embeddings ont permis aux modèles de langage comme les réseaux neuronaux récurrents (RNNs), BERT et GPT d'évoluer rapidement. Ils sont fondamentaux pour des applications telles que la classification de texte, la recherche d'informations et la détection de similarité sémantique.

Prompt

Un **prompt** est une instruction textuelle ou vocale transmise à une IA générative pour déclencher la génération de contenu spécifique. Cette commande constitue le point de départ que le modèle analyse pour produire des réponses cohérentes.

Caractéristiques clés :

- **Format flexible** : Texte (ex. ChatGPT), voix (ex. Siri/Alexa)
- **Fonction duale** : Déclencheur d'action + guide contextuel pour orienter la réponse
- **Applications types** :
  - Génération de contenu créatif (poèmes, scénarios)
  - Résolution de problèmes techniques (codage, débogage)
  - Analyse de données complexes (médicales, financières)

Exemple de structure :
"Écris un article de blog sur l'impact du RAG dans les LLMs en adoptant un ton pédagogique, avec 3 études de cas concrètes et des statistiques récentes."

Prompt Engineering

Le **prompt engineering** est une technique visant à optimiser la formulation des prompts pour maximiser la pertinence des sorties IA. Cette pratique combine compétences linguistiques et connaissances techniques des modèles.

Méthodologies clés :

	Technique	Objectif	Exemple
Précision contextuelle	Ajout de balises sémantiques	Améliorer la compréhension du contexte	"[En tant que data scientist senior] Explique les embeddings..."
Contraintes structurelles	Définition de formats de sortie	Standardiser les réponses	"Génère un tableau comparatif avec colonnes X/Y/Z"
Optimisation itérative	Raffinement progressif	Affiner les résultats par cycles	Version 1 → Analyse → Version 2

Applications avancées :

- **Adaptation cross-modèle** : Techniques spécifiques pour GPT-4 (meilleur en synthèse) vs Bard (accès web temps réel)
- **Réduction des hallucinations** : Mécanismes de vérification intégrés ("Vérifie les faits avant de répondre")
- **Automatisation industrielle** : Intégration dans les pipelines MLOps pour le préprocessing de données

1.1 Comprendre les Agents IA

https://www.anthropic.com/engineering/building-effective-agents

Qu'est-ce qu'un Agent IA ? (Définition et concepts clés)

La définition d'un "agent" est complexe et peut varier selon les sources. Nous pouvons concevoir les agents comme des systèmes entièrement autonomes qui opèrent indépendamment sur de longues périodes, utilisant divers outils pour accomplir des tâches complexes. Nous pouvons également les définir dans le cadre d'implémentations plus prescriptives suivant des flux de travail prédéfinis. Dans le cadre de ce workshop nous categorisoerons toutes les versions sous le terme **agentic systems**.

Il est important d'etablir une distinction architecturale importante entre les workflows et les agents:

Les **workflows** sont des systèmes où les LLMs et les outils sont orchestrés à travers des chemins de code prédéfinis. L'exécution des actions et l'utilisation des outils

sont déterminées par une logique programmée en amont.

Les **agents**, en revanche, sont des systèmes où les LLMs dirigent dynamiquement leurs propres processus et l'utilisation des outils, conservant le contrôle sur la manière dont ils accomplissent les tâches. L'agent prend des décisions en temps réel sur les outils à utiliser et la séquence des actions en fonction du contexte et de son raisonnement.

Un **Agent IA** est une entité qui perçoit son environnement, raisonne et agit pour atteindre des objectifs. Ils sont conçus pour être capables de penser, d'agir et d'utiliser des fonctionnalités externes et d'accéder à des données en temps réel de manière automatisée et autonome.

La relation entre les Agents IA et les **grands modèles de langage (LLMs)** est symbiotique, le LLM agissant souvent comme le cerveau de l'agent. Les Agents IA augmentent les capacités des LLMs en leur permettant d'interagir avec le monde extérieur et de dépasser les limites de leurs données d'entraînement. Par exemple, un LLM seul ne peut pas accéder à des informations en temps réel sur l'état d'une commande, mais un Agent IA peut utiliser des outils pour le faire.

l'idée fondamentale est celle d'un système intelligent capable d'agir de manière autonome pour résoudre des problèmes, en utilisant potentiellement des outils et en se basant sur un LLM pour la prise de décision dynamique (dans le cas des "agents" selon la distinction d'Anthropic) ou en suivant des étapes prédéfinies (dans le cas des "workflows" selon Anthropic).

Composants fondamentaux d'un Agent IA

Selon les sources, les composants clés d'un Agent IA incluent :

- **Agent Core** : Le moteur de décision, souvent basé sur un LLM, qui interprète les entrées et décide des actions. Il utilise des prompts, comme le *ReAct Prompt*, pour déterminer ses pensées et ses actions.
- **Mémoire** : La capacité de l'agent à conserver et à utiliser des informations des interactions passées. Ceci est essentiel pour le suivi des conversations et la construction de workflows.
- **Outils (Actions)** : Des fonctions externes ou des API que l'agent peut invoquer pour interagir avec le monde réel ou accéder à des données. Exemples : recherche web, accès à des bases de données, calculatrices.
- **Module de Planification (si applicable)** : Pour les agents plus complexes, la capacité à décomposer des tâches en étapes et à les exécuter séquentiellement.
- **Système de Prompt** : Un ensemble d'instructions fondamentales qui définissent le comportement, le ton et l'approche décisionnelle de l'agent.

Pourquoi utiliser des agents IA ?

Les Agents IA offrent de nombreux avantages et peuvent être utilisés dans divers contextes :

- **Automatisation Avancée des Tâches** : Automatisation de flux de travail complexes, allant au-delà des simples scripts.
- **Amélioration de la Productivité** : Délégation de tâches répétitives ou nécessitant de la recherche pour libérer du temps.
- **Accès et Traitement de Données en Temps Réel** : Interrogation de données à jour pour prise de décision informée.
- **Support Client Intelligent** : Réponse aux questions fréquentes, suivi des commandes, consultation de bases de connaissances.
- **Automatisation de Flux de Travail Métier** : Analyse de documents, génération de rapports, initiation d'actions basées sur des événements.
- **Analyse de Données et Recherche** : Interrogation de bases de données, recherches web, synthèse d'informations.
- **Création d'Assistants Personnels Avancés** : Planification de rendez-vous, gestion d'informations personnelles.
- **Systèmes Multi-Agents** : Collaboration entre agents spécialisés pour résoudre des problèmes complexes.

1.2 Introduction aux Outils : OpenAI SDK et Agno

Qu'est-ce qu'OpenAI SDK ?

L'**OpenAI SDK** est une bibliothèque Python permettant aux développeurs d'interagir avec les API d'OpenAI.

Avec l'OpenAI SDK, vous pouvez :

- Configurer votre clé d'API pour authentifier vos requêtes.
- Utiliser `openai.chat.completions.create()` pour envoyer des prompts aux modèles et obtenir des réponses.
- Définir des *prompts système* pour orienter le comportement des modèles.
- Passer des *prompts utilisateur* pour poser des questions ou donner des instructions.
- Analyser les réponses de l'API pour extraire le contenu généré.
- Utiliser des outils définis selon le format d'OpenAI pour exécuter des fonctions externes.

Qu'est-ce qu'Agno ?

Agno (anciennement *Phidata*) est un framework Python open-source conçu pour construire des Agents IA multimodaux avec mémoire, connaissances et outils. Il est simple, rapide et indépendant du modèle (*model-agnostic*).

Agno permet de créer des agents qui peuvent travailler avec du texte, des images, de l'audio et de la vidéo. Il facilite la construction d'équipes d'agents spécialisés (*multi-agents*) et offre des fonctionnalités pour la gestion de la mémoire, l'utilisation de bases de connaissances (*vector databases* pour la RAG), et la production de sorties structurées.

Pourquoi utiliser Agno ?

- **Simplicité et Rapidité** : Création rapide d'agents avec seulement trois lignes de code.
- **Indépendance du Modèle** : Compatible avec plusieurs fournisseurs de modèles (ex. OpenAI, Mistral, Anthropic).
- **Capacités Multimodales** : Support natif pour texte, images, audio et vidéo.
- **Support Multi-Agents** : Facilite la collaboration entre agents spécialisés.
- **Fonctionnalités Avancées** : Mémoire intégrée, gestion des connaissances (*RAG*), sorties structurées.
- **Approche IA comme Ingénierie Logicielle** : Utilisation de constructions Python standards (*if, else, while, for*).
- **Facilité d'Intégration avec des Outils** : Recherche web, finance, bases de données.
- **Interface Utilisateur (UI)** : Agno propose une UI graphique pour interagir avec les agents.

En résumé, **Agno** est un framework puissant et flexible pour développer des Agents IA intelligents et adaptatifs.

1.3 Définition de votre Agent IA

- **Atelier interactif** : Définir un cas d'usage simple

    - Objectif de l'agent
    - Entrées et sorties attendues
    - Actions/outils nécessaires
- Exemples : chatbot de support, analyseur de texte, recherche web

---

2eme heure – Développement Pratique d'un Agent IA

2.1 Configuration de l'environnement

- Installation des dépendances
- Gestion des clés API avec variables d'environnement
- Exécution d'un premier test simple avec OpenAI SDK

2.2 Interaction avec OpenAI en Python

- Création d'un premier agent conversationnel simple
- Structuration des prompts et rôle du system message

2.3 Ajout d'outils à l'Agent avec Agno

- Définition et enregistrement d'actions avec Agno
- Intégration d'une action simple (exemple : récupération de la météo, recherche web)
- Exécution et test de l'agent avec plusieurs entrées

3eme heure – Amélioration, Scalabilité et Déploiement

3.1 Optimisation et Cas d'Utilisation Avancés

- Agents multi-outils
- RAG (Retrieval Augmented Generation)
- Discussion sur les limitations et meilleures pratiques

3.2 Projet Pratique : Création de votre Agent IA

- Développement d'un agent en binôme ou individuel
- Exemples de tâches :
    - Un chatbot assistant
    - Un analyseur de texte intelligent
    - Un agent de veille automatique
- Assistance et debugging en direct

3.3 Prochaines Étapes et Ressources

- Exploration des sujets avancés :
    - Ajout de mémoire pour la gestion du contexte
    - Utilisation d'outils plus complexes (connexion à une base de données, génération de texte avancée)
    - Déploiement dans un environnement cloud
- Ressources pour approfondir : Documentation OpenAI et Agno, cours en ligne, GitHub de projets

Citations et sources utilisés pour le workshop

AI Tools:

- perplexity.ai
- NotebookLM
- Claude.ai
- OpenAI

Ressources

[1] https://www.data-bird.co/blog/llm-definition

[2] https://fr.wikipedia.org/wiki/Mod%C3%A8le_de_fondation

[3] https://www.promptingguide.ai/research/rag

[4] https://bigblue.academy/en/text-embeddings

[5] https://aws.amazon.com/fr/what-is/ai-agents/

[6] https://www.dataleon.ai/blog/quest-ce-quun-large-language-model-llm

[7] https://www.redhat.com/fr/topics/ai/what-are-foundation-models

[8] https://cloud.google.com/use-cases/retrieval-augmented-generation

[9] https://datascientest.com/le-word-embedding

[10] https://www.ibm.com/fr-fr/think/topics/ai-agents

[11] https://www.cloudflare.com/fr-fr/learning/ai/what-is-large-language-model/

[12] https://www.lebigdata.fr/modele-de-fondation-une-notion-fondamentale-en-intelligence-artificielle

[13] https://aws.amazon.com/what-is/retrieval-augmented-generation/

[14] https://infoscience.epfl.ch/record/221430/?v=%5B%27pdf%27%5D

[15] https://fr.wikipedia.org/wiki/Agent_intelligent

[16] https://www.hpe.com/ch/fr/what-is/large-language-model.html

[17] https://aws.amazon.com/fr/what-is/foundation-models/

[18] https://www.redhat.com/fr/topics/ai/what-is-retrieval-augmented-generation

[19] https://www.datacamp.com/fr/blog/what-is-text-embedding-ai

[20] https://botpress.com/fr/blog/what-is-an-ai-agent

[21] https://datascientest.com/prompt-tout-savoir

[22] https://www.ibm.com/fr-fr/think/topics/prompt-engineering

[23] https://www.lebigdata.fr/prompt-definition

[24] https://datascientest.com/prompt-engineer-tout-savoir

[25] https://www.intelligence-artificielle-school.com/ecole/technologies/quest-ce-quun-prompt-en-ia/

[26] https://platform.openai.com/docs/guides/prompt-engineering

[27] https://www.lacreme.ai/post/quest-ce-quun-prompt-definition-intelligence-artificielle

[28] https://en.wikipedia.org/wiki/Prompt_engineering

[29] https://www.orientaction-groupe.com/vocabulaire-ia-prompt/