

Mémoire du Projet de Fin d'Études

En vue de l'obtention du diplôme

**MASTER SCIENCE ET INGÉNIERIE
DE DONNÉES**

Présenté(e) par

M. Zarouala Abdellah

Sous le thème :

**Système de recommandation de cours
éducatifs basé sur l'IA générative**

Soutenu le 12 Juillet 2025, devant le jury composé de :

Pr. SOUMIA ZITI

PES, FSR - UM5R

Présidente

Pr. NASSIM KHARMOUM

MCH, FSR - UM5R

Encadrant interne

Dr. WAFAE ABBAOUI

MC EMSI

Examinateuse

Dr. MOHAMMED AMRAOUI

Docteur, FSR - UM5R

Examinateur

Dédicace

À nos très chers parents,

En témoignage de notre amour et de notre gratitude pour les sacrifices consentis à notre égard.

À nos frères et sœurs,

Pour leur soutien et leur encouragement.

À nos familles,

Pour avoir attendu avec patience les fruits de leur bonne éducation.

À nos amis,

À tous ceux qui, de près ou de loin, nous sont chers,

Nous dédions ce rapport.

Remerciements

C'est avec un grand honneur et un plaisir énorme que je dédie ces quelques lignes afin de remercier toute personne ayant contribué de près ou de loin à l'accomplissement de ce projet.

Notre gratitude s'adresse spécialement à :

M. **Kharmoum Nassim**, pour son encadrement, son assistance et ses précieux conseils tout au long de ce projet, ainsi que pour l'aide qu'il m'a prodiguée durant la rédaction de ce rapport.

Je tiens à exprimer ma profonde gratitude à Mme **Ziti**, notre responsable du Master MSID, pour son soutien inestimable tout au long de mon parcours académique. Son engagement, sa disponibilité, son expertise ainsi que ses conseils ont été d'une grande importance.

Mes remerciements s'adressent également à tous les membres des jurys qui ont accepté d'évaluer mon humble travail.

Je tiens également à remercier tous les enseignants du Master MSID qui m'ont pleinement fait bénéficier de leurs connaissances et de leur savoir-faire.

Résumé

La plupart des plateformes d'enseignement en ligne encouragent leurs utilisateurs à explorer différentes thématiques avant de s'engager dans une spécialisation, et à développer une culture académique large en suivant des parcours diversifiés. Chaque période, les apprenants doivent choisir, parmi des milliers de cours disponibles dans de nombreux domaines, une sélection restreinte de formations à suivre. L'environnement en ligne est également très dynamique, et une communication insuffisante ainsi que des fonctions de recherche peu performantes peuvent limiter la capacité des apprenants à découvrir de nouveaux cours adaptés à leurs intérêts.

Pour aider à la fois les apprenants et les conseillers pédagogiques dans ce contexte, nous explorons un système novateur de recommandation de cours en ligne basé sur un grand modèle de langage (LLM) utilisant une méthode de *Retrieval Augmented Generation* (RAG) appliquée au corpus des descriptions de cours. Le système génère d'abord une description de cours « idéale » à partir de la requête ou du profil de l'utilisateur. Cette description est convertie en un vecteur de recherche via des embeddings, qui est ensuite utilisé pour retrouver des cours réels au contenu similaire en comparant les similarités d'embeddings.

Nous décrivons la méthode et évaluons la qualité ainsi que la pertinence de certains exemples de requêtes. Les étapes pour déployer un système pilote sur une plateforme d'apprentissage en ligne sont également discutées.

Mots-clés : systèmes de recommandation, grands modèles de langage, génération augmentée par récupération, technologie éducative, recommandation de cours en ligne

Abstract

Most online learning platforms encourage their users to explore various topics before committing to a specialization and to develop a broad academic culture by following diverse learning paths. Each term, learners must choose a limited selection of courses to take from among thousands of available options in many fields. The online environment is also highly dynamic, and insufficient communication as well as poorly performing search functions can limit learners' ability to discover new courses suited to their interests.

To support both learners and educational advisors in this context, we explore an innovative online course recommendation system based on a large language model (LLM) using a Retrieval-Augmented Generation (RAG) method applied to a corpus of course descriptions. The system first generates an "ideal" course description from the user's query or profile. This description is converted into a search vector via embeddings, which is then used to retrieve actual courses with similar content by comparing embedding similarities.

We describe the method and evaluate the quality and relevance of some example queries. The steps to deploy a pilot system on an online learning platform are also discussed.

Keywords : recommendation systems, large language models, retrieval-augmented generation, educational technology, online course recommendation

Table des matières

Dédicace	1
Remerciements	2
Résumé	3
Abstract	4
Table des matières	7
Liste des figures	8
Liste des tableaux	9
1 Intelligence artificielle générative	13
Introduction	13
1.1 Quelles techniques sont utilisées dans l'IA générative ?	13
1.2 A quoi sert l'IA générative ?	13
1.3 Quels sont les inconvénients de l'IA générative ?	14
1.4 Quels sont les avantages de l'IA générative ?	14
1.5 Comment a évolué la technologie d'IA générative ?	15
1.5.1 VAE	15
1.5.2 Transformateurs	16
1.5.3 L'avenir	16
1.6 Comment fonctionne l'IA générative ?	16
1.7 Fonctionnement des modèles d'IA générative	17
1.7.1 Modèles de diffusion	17
1.7.2 Réseaux antagonistes génératifs	18
1.7.3 Autoencodeurs variationnels	19
1.7.4 Modèles basés sur des transformateurs	20
1.8 Quelles sont les limites de l'IA générative ?	20
1.9 Quelles sont les bonnes pratiques en matière d'adoption de l'IA générative ?	21
1.10 Amélioration de la pertinence des chatbots d'IA générative	22
1.10.1 RAG (Retrieval-Augmented Generation)	22
1.10.2 Fine-Tuning ?	24
1.10.3 Le prompt engineering	26
1.10.4 Few Shot Learning ?	28
1.10.5 Low Rank Adaptation	30
1.10.6 Comparaison des Techniques	32
Conclusion	33

2 Système de recommandation	34
Introduction	34
2.1 Classification des systèmes de recommandation	35
2.1.1 Le filtrage basé sur le contenu	35
2.1.2 Le filtrage collaboratif	37
2.1.3 Filtrage item-item (Linden et al., 2003)	37
2.1.4 Filtrage utilisateur-utilisateur	38
2.1.5 Le filtrage hybride	39
2.2 Évaluation des systèmes de recommandation	39
2.2.1 Mesures de précision des recommandations	39
2.2.1.1 RMSE (Root Mean Squared Error)	39
2.2.1.2 MAE (Mean Absolute Error)	39
2.2.2 Mesures de classification des recommandations	39
2.2.2.1 Précision (Precision)	39
2.2.2.2 Rappel (Recall)	40
2.2.2.3 F1-Score	40
2.2.3 Mesures de classement des recommandations	40
2.2.3.1 NDCG (Normalized Discounted Cumulative Gain)	40
2.2.3.2 MRR (Mean Reciprocal Rank)	40
2.2.4 Autres critères d'évaluation	40
2.2.4.1 Couverture	40
2.2.4.2 Diversité	41
2.2.4.3 Nouveauté	41
2.3 Limites des systèmes de recommandation	41
2.3.1 Problème du Cold Start	41
2.3.2 Problème de Sparsité	41
2.3.3 Problème de Sur-apprentissage (Overfitting)	42
2.3.4 Problème du Biais et de la Diversité	42
2.3.5 Problème de Scalabilité	42
2.3.6 Problème d'Évaluation et de Mesure	43
3 Intégration de l'IA Générative dans les Systèmes de Recommandation Éducatifs	44
Introduction	44
Contexte de l'IAG	44
Objectifs du chapitre	45
Rôle des LLM	45
3.1 Apports de l'IA générative aux systèmes de recommandation éducatifs	46
3.1.1 Personnalisation avancée des parcours d'apprentissage	46
3.1.2 Génération de contenus pédagogiques adaptés	46
3.1.3 Réponses contextuelles et interactives aux besoins des apprenants	47
3.2 Systèmes à base de connaissances	47
3.3 Approches hybrides	47
3.4 Exigences en matière de données	47
3.5 LLM pour les systèmes de recommandation	48
3.6 Processus de Recommandation de Cours	48
3.7 Données	49
3.8 Génération du Contexte	50
3.9 Recommandation	51

4 Conception Détailée	53
Introduction	53
4.1 Analyse et Conception	54
4.1.1 Les Acteurs du Système	54
4.2 Diagramme de Cas d'utilisation	54
4.3 Diagrammes de Séquence	54
4.4 Diagrammes d'Activité	55
5 Réalisation	57
Introduction	57
5.1 Méthodologie de développement	58
5.1.1 Langage UML	58
5.1.2 CrewAI	58
5.1.2.1 Agents d'IA vs. LLMs	59
5.1.2.2 La nécessité d'un cadre pour les agents	60
5.1.2.3 Orchestration et coordination	60
5.1.2.4 Qu'est-ce que CrewAI ?	60
5.1.3 Ollama	62
5.2 Quoi de neuf avec LLaMA 3.2 ?	63
5.2.1 Architecture technique	63
5.2.2 Performances et benchmarks	64
5.2.3 Points forts de LLaMA 3.2	64
5.2.4 Cas d'usage concrets	64
5.2.5 Enjeux et limites	64
5.2.6 Perspectives futures	65
5.2.7 Streamlit	65
5.3 Base de Données : MySQL	66
5.4 Procédure de Déploiement de l'Application Streamlit	67
5.4.1 1. Pré-requis	67
5.4.2 2. Création et activation de l'environnement virtuel	67
5.4.3 3. Installation des dépendances	67
5.4.4 4. Lancement de l'application en local	67
5.4.5 5. Déploiement sur une plateforme en ligne (facultatif)	67
5.4.6 6. Mise à jour de l'application	68
5.4.7 7. Résolution des problèmes	68
5.5 Interfaces graphiques web	69
Conclusion Générale et Perspectives	76
Bibliographie et Webographie	78

Table des figures

1.1	VAE Illustration	15
1.2	Transformateurs	16
1.3	L'avenir de l'IA générative	16
1.4	Exemple de modèle de diffusion	18
1.5	GAN (réseaux antagonistes génératifs)	19
1.6	Exemple d'autoencodeur variationnel (VAE)	19
1.7	Modèle basé sur un transformateur	20
2.1	Classification des systèmes de recommandation (Isinkaye et al., 2015) . .	35
2.2	Structure en arbre d'un simple site web.PRES [Maarten et Someren, 2000]	36
2.3	Filtrage collaboratif basé sur les éléments.	38
3.1	Pipeline hybride LLM-Embedding pour les recommandations de cours . .	49
3.2	Phases de la Génération du Contexte	50
4.1	Diagramme de cas d'utilisation du système	54
4.2	Diagramme de séquence du système	55
4.3	Diagramme d'activité représentant le déroulement des actions du système	56
5.1	Logo UML	58
5.2	Agents d'IA vs. LLMs	59
5.3	Logo de CrewAI	60
5.4	Principaux avantages offerts par CrewAI	62
5.5	Logo de Streamlit	65
5.6	Interface ou composant lié à MySQL	66
5.7	Interface de la page d'inscription	69
5.8	Interface de la page d'inscription	70
5.9	Interface de la page d'accueil après connexion	71
5.10	Formulaire de saisie manuelle des informations utilisateur	71
5.11	Page d'historique des recommandations de cours	72
5.12	Interface de sélection du fichier CV	72
5.13	Indicateur de chargement pendant l'analyse du CV	73
5.14	Aperçu des données extraites du CV	73
5.15	Analyse du profil de l'utilisateur	74
5.16	Création des recommandations personnalisées	74
5.17	Affichage final des recommandations de cours	75

Liste des tableaux

1.1	Comparaison des principales techniques d'optimisation de modèles d'IA	32
2.1	Matrice Items-utilisateurs	37

Liste des abréviations et sigles

API	Application Programming Interface
CV	Curriculum Vitae
HTTP	Hypertext Transfer Protocol
IA	Intelligence Artificielle
IAG	Intelligence Artificielle Générative
JSON	JavaScript Object Notation
LLM	Large Language Model (Modèle de Langage de Grande Taille)
MAE	Mean Absolute Error
MRR	Mean Reciprocal Rank
NDCG	Normalized Discounted Cumulative Gain
NLP	Natural Language Processing (Traitement Automatique du Langage Naturel)
PDF	Portable Document Format
RAG	Retrieval-Augmented Generation
RMSE	Root Mean Squared Error
SQL	Structured Query Language
UML	Unified Modeling Language
UI	User Interface (Interface Utilisateur)
UX	User Experience (Expérience Utilisateur)

Introduction Générale

Dans un monde en constante évolution, où les compétences requises sur le marché du travail changent rapidement, l'adaptation continue des parcours de formation devient un enjeu stratégique tant pour les individus que pour les organisations. La transformation numérique a bouleversé de nombreux secteurs, et le domaine de l'éducation et de la formation n'y échappe pas. Dans ce contexte, les technologies de l'intelligence artificielle (IA) offrent aujourd'hui des solutions puissantes pour répondre aux nouveaux besoins de personnalisation, d'automatisation et d'assistance intelligente à l'apprentissage.

L'émergence des modèles de langage de grande taille (LLM - *Large Language Models*), tels que GPT, LLaMA ou Claude, a marqué un tournant dans la manière dont les machines interagissent avec le langage humain. Ces modèles sont capables de comprendre, générer et manipuler du texte de manière de plus en plus fluide, ouvrant la voie à des applications variées : agents conversationnels, assistants rédactionnels, moteurs de recherche augmentés, ou encore systèmes de recommandation intelligents. Combinés à des techniques comme le RAG (*Retrieval-Augmented Generation*), qui permettent d'enrichir les réponses générées par l'IA à partir de données fiables et ciblées, ces modèles deviennent des outils puissants pour produire des recommandations contextuelles, pertinentes et personnalisées.

C'est dans cette optique que s'inscrit le projet de stage intitulé « **Système de recommandation basé sur l'IA générative** », dont l'objectif est de développer une plateforme intelligente capable de recommander des **cours personnalisés** à un utilisateur, à partir de l'analyse automatisée de son **curriculum vitae (CV)**. Concrètement, l'utilisateur téléverse son CV (en format PDF, Word ou texte), et le système, à l'aide d'un **LLM embarqué localement via la plateforme Ollama**, analyse le contenu du document, extrait les informations clés (formations, compétences, expériences professionnelles, etc.), puis interroge une base de connaissances de cours afin de proposer une liste pertinente de formations adaptées à son profil, ses objectifs, ou ses lacunes identifiées.

Pour ce faire, le système repose sur plusieurs briques technologiques modernes :

- **LLaMA 3.2**, un modèle de langage performant, local et open-source, permettant d'assurer à la fois puissance et confidentialité des données ;
- **Ollama**, une plateforme facilitant l'intégration et le déploiement local de modèles LLM sans dépendance directe au cloud ;
- **RAG (Retrieval-Augmented Generation)**, qui permet d'interroger dynamiquement une base de cours structurée afin de renforcer la pertinence des recommandations ;
- Une interface utilisateur développée avec **Streamlit**, offrant une expérience fluide et intuitive.

Ce projet se distingue par sa capacité à **automatiser le processus de recommandation de formation**, en le rendant à la fois **intelligent, interactif et respectueux des données personnelles**, puisque l'intégralité du traitement est effectuée en local. Il constitue une avancée significative dans le domaine de la formation personnalisée, avec des perspectives d'utilisation dans des plateformes de recrutement, des centres de formation ou même dans le cadre de l'orientation professionnelle.

Le présent rapport de stage s'articule autour des axes suivants : nous commencerons par présenter le **contexte général du projet** et les motivations qui ont conduit à son développement. Ensuite, nous détaillerons les **choix technologiques**, l'**architecture du système**, ainsi que les étapes de conception et d'intégration des différents modules. Nous aborderons par la suite les **résultats obtenus**, les tests effectués, et les **limites identifiées**, avant de conclure par une ouverture sur les améliorations possibles et les perspectives futures du projet.

Chapitre 1

Intelligence artificielle générative

Introduction

En quoi consiste l'IA générative ? L'intelligence artificielle générative est une catégorie d'IA qui se concentre sur la création autonome de contenu, tels que des textes, des images, des vidéos, des sons et d'autres types de données, par des systèmes informatiques.

Ces systèmes utilisent des modèles avancés d'apprentissage automatique pour générer du contenu qui peut ressembler à ce qui est créé par des êtres humains.

1.1 Quelles techniques sont utilisées dans l'IA générative ?

Les deux types d'IA générative les plus utilisées sont :

- **Les GAN (neurones génératifs antagonistes)** : une architecture de réseau de neurones artificiels composée de deux parties, le générateur et le discriminant. Le générateur crée de nouvelles données, tandis que le discriminant essaie de distinguer les données générées de données réelles. Les GAN s'améliorent continuellement à mesure que le générateur tente de tromper le discriminant, créant ainsi des données de plus en plus réaliste. Les GAN sont couramment utilisés pour générer des images, des vidéos et des textes.
- **Les GPT (Generative Pre-trained Transformer)** : des modèles d'apprentissage automatique qui ont été formés sur de grandes quantités de données textuelles. Ces modèles peuvent générer du texte cohérent et contextuellement pertinent en fonction des données d'entrée. Ils sont utilisés pour des applications telles que la génération automatique de textes, la traduction automatique et la rédaction assistée par ordinateur.

1.2 A quoi sert l'IA générative ?

Les applications de l'IA générative sont nombreuses et très diverses. Elle peut servir pour :

- **Alimenter la création artistique** : en générant de l'art visuel, de la musique, de la littérature et d'autres formes d'expression artistique ;
- **Améliorer la création de contenu** : en aidant les rédacteurs à générer du contenu rédactionnel, tels que des articles, des rapports ou même des scripts pour la création de vidéos ;

- **Créer des mondes virtuels** : des personnages et des scénarios dans des jeux vidéo et des simulations ;
- **Personnaliser l'expérience utilisateur** : en tenant compte des préférences individuelles de chaque utilisateur ;
- **Générer des données de test** : en informatique ou en science ;
- **Coder des programmes simples** : grâce notamment au *no-code*, et remplacer le *low-code*.

1.3 Quels sont les inconvénients de l'IA générative ?

Les concepteurs, les développeurs et les utilisateurs de ces systèmes doivent également être conscients des implications éthiques et sociales et veiller à une utilisation responsable de cette technologie. Voici quelques-uns des principaux inconvénients à avoir en tête :

- **Qualité variable du contenu** : Les résultats générés par des modèles d'IA générative peuvent varier en termes de qualité et de pertinence. Il est possible d'obtenir du contenu de qualité médiocre, trompeur ou inutile. L'IA générative peut être utilisée de manière malveillante pour créer de la désinformation, des *deepfakes* et d'autres formes de manipulation de contenu. La source du contenu généré peut être remise en question, affectant la confiance du public dans les informations en ligne.
- **Problèmes de responsabilité** : Déterminer la responsabilité en cas de contenu inapproprié ou problématique généré par une IA peut être complexe, notamment lorsqu'il s'agit de modèles pré-entraînés sur de vastes ensembles de données.
- **Surcharge d'informations** : L'IA générative peut générer un volume massif de contenu, ce qui complique la recherche d'informations pertinentes.
- **Biais et discrimination** : Les modèles peuvent reproduire les biais présents dans les données d'entraînement, entraînant la création de contenu discriminatoire, offensant ou partial.
- **Menace pour l'emploi** : Dans certains secteurs, l'automatisation de la création de contenu peut mener à la suppression d'emplois (rédaction, conception graphique, etc.).
- **Problèmes de sécurité** : Les cybercriminels peuvent exploiter l'IA générative pour créer des contrefaçons, des faux documents ou des attaques de *phishing* plus sophistiquées.
- **Défis éthiques** : Des questions importantes se posent en matière de propriété intellectuelle, de création automatisée sans consentement, et de respect de la vie privée.

1.4 Quels sont les avantages de l'IA générative ?

Comme vu précédemment, l'Intelligence Artificielle (IA) générative offre des possibilités innovantes et transforme la manière dont les entreprises opèrent à travers divers secteurs.

Au cœur de cette révolution se trouve la capacité de l'IA à créer :

- **Du contenu original** : en utilisant des modèles d'apprentissage automatique. L'IA générative analyse et apprend à partir de vastes ensembles de données pour produire de nouveaux textes, images, vidéos ou musiques. Grâce à sa capacité à comprendre les motifs et les structures sous-jacents des données, elle génère un contenu authentique et créatif, souvent indiscernable de celui créé par des humains.

- **Des solutions sur mesure et des analyses prédictives** : en comprenant les tendances et les schémas sous-jacents, elle produit des solutions adaptées aux exigences spécifiques de chaque situation, augmentant ainsi l'efficacité.
- **Une automatisation efficace des tâches répétitives** : ce qui permet aux entreprises de réduire les coûts et les délais de production, tout en libérant du temps pour des activités à plus forte valeur ajoutée.
- **L'innovation** : notamment à travers la génération automatique de rapports, l'IA générative booste la productivité et recentre les efforts humains sur des tâches stratégiques.
- **La personnalisation à des niveaux inédits** : dans les médias sociaux, la publicité ou les services en ligne, elle permet de proposer des expériences uniques adaptées à chaque utilisateur.

1.5 Comment a évolué la technologie d'IA générative ?

Les modèles génératifs primitifs sont utilisés depuis plusieurs décennies dans les statistiques pour faciliter l'analyse numérique des données. Les réseaux neuronaux et le deep learning sont les précurseurs récents de l'IA générative moderne. Les autoencodeurs variationnels (VAE), développés en 2013, ont été les premiers modèles génératifs de deep learning capables de générer des images et des discours réalistes.

1.5.1 VAE

Les VAE (encodeurs automatiques variationnels) ont été les premiers à créer de nouvelles variantes de plusieurs types de données. Cela a conduit à l'émergence rapide d'autres modèles d'IA générative tels que les réseaux antagonistes génératifs et les modèles de diffusion. Ces innovations visaient à générer des données qui ressemblaient de plus en plus à des données réelles, bien qu'elles aient été créées artificiellement.



FIGURE 1.1 – VAE Illustration

1.5.2 Transformateurs

En 2017, un nouveau tournant dans la recherche sur l'IA s'est produit avec l'introduction des transformateurs. Les transformateurs ont parfaitement intégré l'architecture encodeur-décodeur grâce à un mécanisme d'attention. Ils ont rationalisé le processus d'entraînement des modèles de langage avec une efficacité et une polyvalence exceptionnelles. Des modèles remarquables tels que GPT se sont imposés comme des modèles incontournables capables de se pré-entraîner sur de vastes corpus de texte brut et de les affiner pour des tâches variées.

Les transformateurs ont repoussé les limites du traitement du langage naturel. Ils ont renforcé les capacités génératives pour des tâches allant de la traduction et de la synthèse à la réponse aux questions.

1.5.3 L'avenir

De nombreux modèles d'IA générative continuent de progresser de manière significative et ont maintenant des applications intersectorielles. Les innovations récentes visent à affiner les modèles afin qu'ils puissent utiliser des données propriétaires. Les chercheurs souhaitent également créer du texte, des images, des vidéos et des discours de plus en plus humains.

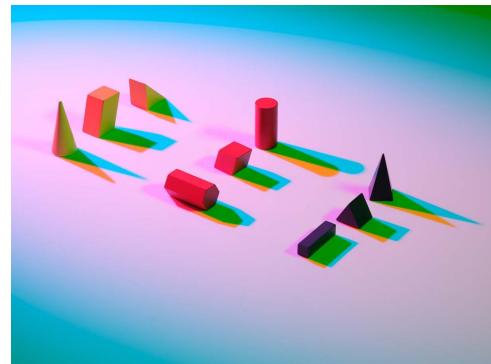


FIGURE 1.2 – Transformateurs

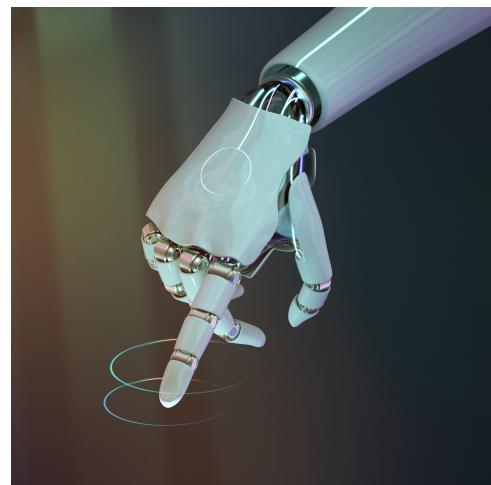


FIGURE 1.3 – L'avenir de l'IA générative

1.6 Comment fonctionne l'IA générative ?

Comme toutes les intelligences artificielles, l'IA générative fonctionne en utilisant des modèles de machine learning, de très grands modèles pré-entraînés sur de vastes quantités de données.

Modèles de fondation

Les modèles de fondation (FM) sont des modèles de ML entraînés sur un large éventail de données généralisées et non étiquetées. Ils sont capables d'effectuer une grande variété de tâches générales.

Les FM sont le résultat des dernières avancées d'une technologie qui évolue depuis des décennies. En général, un FM utilise des modèles et des relations appris pour prédire le prochain élément d'une séquence.

Par exemple, lors de la génération d'images, le modèle analyse l'image et crée une version plus nette et plus clairement définie de l'image. De même, dans le cas du texte, le modèle prédit le mot suivant dans une chaîne de texte en fonction des mots précédents et de leur contexte. Il sélectionne ensuite le mot suivant à l'aide de techniques de distribution de probabilité.

Grands modèles de langage

Les grands modèles de langage (LLM) sont une classe de FM. Par exemple, les modèles transformateurs génératifs pré-entraînés (GPT) d'OpenAI sont des LLM. Les LLM sont spécifiquement axés sur les tâches basées sur le langage, telles que le résumé, la génération de texte, la classification, la conversation ouverte et l'extraction d'informations.

Ce qui rend les LLM spéciaux, c'est leur capacité à effectuer de multiples tâches. Ils peuvent le faire car ils contiennent de nombreux paramètres qui les rendent capables d'apprendre des concepts avancés.

Un LLM comme GPT-3 peut prendre en compte des milliards de paramètres et générer du contenu à partir de très peu d'entrées. Grâce à leur exposition, avant l'entraînement, à des données à l'échelle de l'Internet sous toutes leurs formes et dans une myriade de modèles, les LLM apprennent à appliquer leurs connaissances dans un large éventail de contextes.

1.7 Fonctionnement des modèles d'IA générative

Les modèles de machine learning traditionnels étaient discriminants ou centrés sur la classification des points de données. Ils tentent de déterminer la relation entre les facteurs connus et inconnus. Par exemple, ils examinent des images (des données connues telles que la disposition des pixels, les lignes, les couleurs et les formes) et les mappent avec des mots, le facteur inconnu. Mathématiquement parlant, ces modèles identifient des équations qui peuvent mapper numériquement des facteurs inconnus et connus sous forme de variables x et y .

Les modèles génératifs vont un peu plus loin. Au lieu de prédire une étiquette en fonction de fonctionnalités, ils essaient de prédire les caractéristiques associées à une étiquette donnée. Mathématiquement parlant, la modélisation générative calcule la probabilité que x et y apparaissent ensemble. Elle apprend la distribution des différentes fonctionnalités des données et leurs relations.

Par exemple, les modèles génératifs analysent des images d'animaux pour enregistrer des variables telles que différentes formes d'oreilles, d'yeux, de queue et de peau. Ils apprennent leurs fonctionnalités et leurs relations pour comprendre à quoi ressemblent les différents animaux en général. Ils peuvent ensuite reconstituer des images d'animaux qui ne figuraient pas dans le jeu de données d'entraînement.

Nous allons maintenant nous pencher sur les principales catégories de modèles d'IA générative.

1.7.1 Modèles de diffusion

Les modèles de diffusion créent des données en apportant de manière itérative des modifications aléatoires contrôlées à un échantillon de données initial. Ils commencent par ajouter des modifications subtiles (bruit) et progressives aux données d'origine. Ce

bruit est soigneusement contrôlé pour garantir que les données générées restent cohérentes et réalistes.

Après avoir ajouté du bruit sur plusieurs itérations, le modèle de diffusion répète le processus dans le sens inverse. Le débruitage inversé supprime progressivement le bruit pour produire un nouvel échantillon de données qui ressemble à l'original.

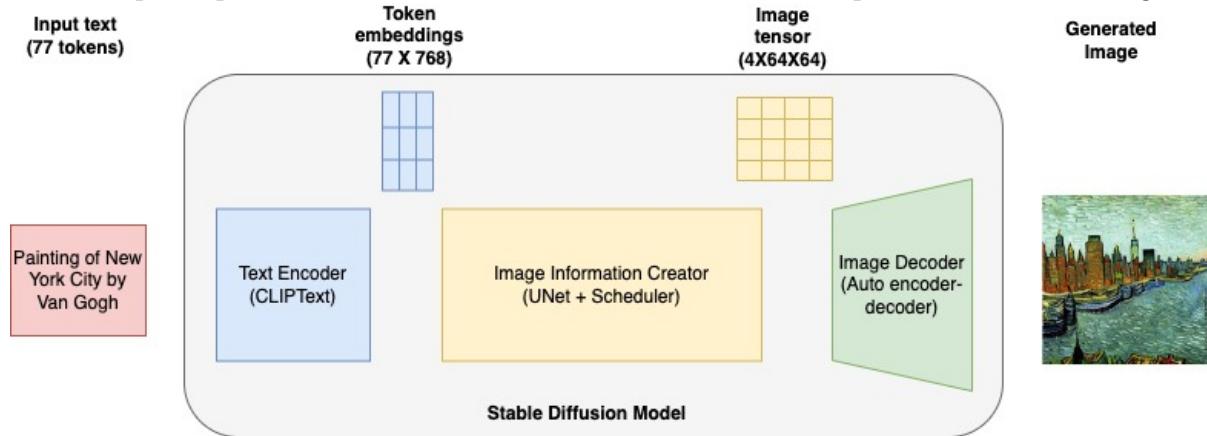


FIGURE 1.4 – Exemple de modèle de diffusion

1.7.2 Réseaux antagonistes génératifs

Le réseau antagoniste génératif (GAN) est un autre modèle d'IA génératrice qui s'appuie sur le concept du modèle de diffusion.

Les GAN fonctionnent en entraînant deux réseaux neuronaux de manière compétitive. Le premier réseau, appelé *générateur*, génère de faux échantillons de données en ajoutant du bruit aléatoire. Le second réseau, appelé *discriminateur*, essaie de faire la distinction entre les données réelles et les fausses données produites par le générateur.

Pendant l'entraînement, le générateur améliore continuellement sa capacité à créer des données réalistes, tandis que le discriminateur sait de mieux en mieux distinguer le vrai du faux. Ce processus contradictoire se poursuit jusqu'à ce que le générateur produise des données si convaincantes que le discriminateur ne peut pas les différencier des données réelles.

Les GAN sont largement utilisés pour la génération d'images réalistes, le transfert de style et les tâches d'augmentation des données.



FIGURE 1.5 – GAN (réseaux antagonistes génératifs)

1.7.3 Autoencodeurs variationnels

Les autoencodeurs variationnels (VAE) apprennent une représentation compacte des données appelée *espace latent*. L'espace latent est une représentation mathématique des données. Vous pouvez le voir comme un code unique représentant les données en fonction de tous leurs attributs. Par exemple, si vous étudiez des visages, l'espace latent contient des nombres représentant la forme des yeux, du nez, des pommettes et des oreilles.

Les VAE utilisent deux réseaux neuronaux : l'*encodeur* et le *décodeur*. L'encodeur mappe les données d'entrée selon une moyenne et une variance pour chaque dimension de l'espace latent. Il génère un échantillon aléatoire à partir d'une distribution gaussienne (normale). Cet échantillon est un point dans l'espace latent et représente une version compressée et simplifiée des données d'entrée.

Le décodeur extrait ce point échantillonné de l'espace latent et le reconstruit en données qui ressemblent à l'entrée d'origine. Des fonctions mathématiques sont utilisées pour mesurer à quel point les données reconstruites correspondent aux données d'origine.

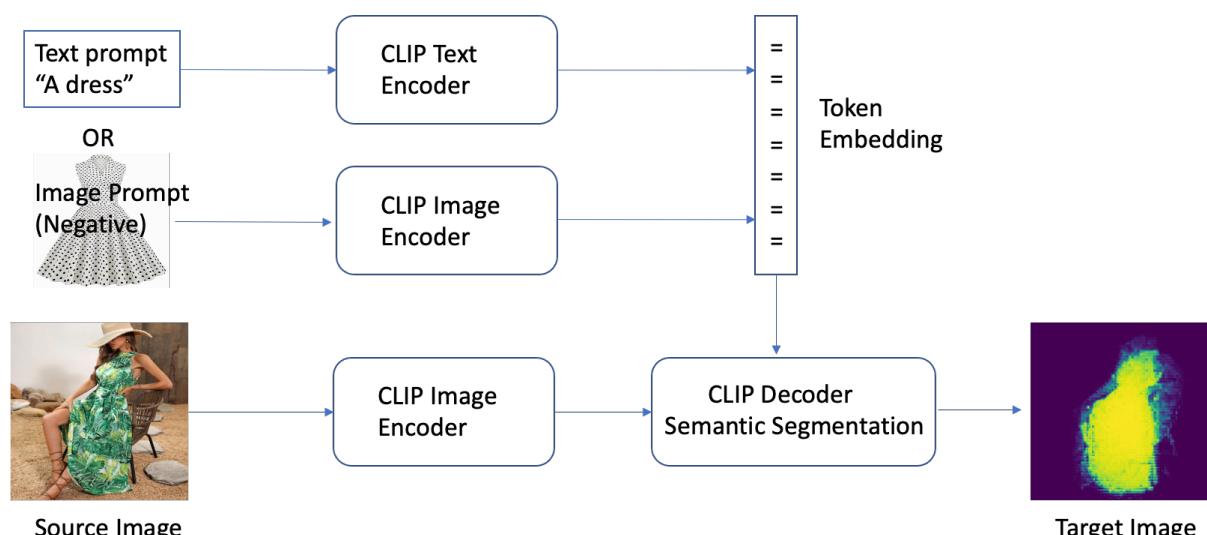


FIGURE 1.6 – Exemple d'autoencodeur variationnel (VAE)

1.7.4 Modèles basés sur des transformateurs

Les modèles d'IA générative basés sur des transformateurs s'appuient sur les concepts d'encodeur et de décodeur des VAE. Les modèles basés sur des transformateurs ajoutent des couches supplémentaires à l'encodeur afin d'améliorer les performances des tâches basées sur du texte telles que la compréhension, la traduction et l'écriture créative.

Les modèles basés sur des transformateurs utilisent un mécanisme d'*auto-attention*. Ils évaluent l'importance des différentes parties d'une séquence d'entrée lors du traitement de chaque élément de la séquence.

Une autre fonctionnalité clé est que ces modèles d'IA implémentent des *intégrations contextuelles*. Le codage d'un élément de séquence dépend non seulement de l'élément lui-même, mais également de son contexte dans la séquence.

Fonctionnement des modèles basés sur des transformateurs Pour comprendre le fonctionnement des modèles basés sur des transformateurs, imaginez une phrase sous la forme d'une séquence de mots.

L'*auto-attention* aide le modèle à se concentrer sur les mots pertinents lorsqu'il traite chaque mot. Pour capturer différents types de relations entre les mots, les modèles génératifs basés sur des transformateurs utilisent plusieurs couches d'encodeur appelées *têtes d'attention*. Chaque tête apprend à se concentrer sur différentes parties de la séquence d'entrée, ce qui permet au modèle de prendre en compte simultanément divers aspects des données.

Chaque couche affine également les intégrations contextuelles, les rendant plus informatives et capturant tout, de la syntaxe grammaticale aux significations sémantiques complexes.



FIGURE 1.7 – Modèle basé sur un transformateur

1.8 Quelles sont les limites de l'IA générative ?

Malgré leurs avancées, les systèmes d'IA générative peuvent parfois produire des informations inexactes ou trompeuses. Ils s'appuient sur des modèles et des données sur lesquels

ils ont été formés et peuvent refléter des biais ou des inexactitudes inhérents à ces données. Les autres préoccupations liées aux données d'entraînement incluent :

Sécurité

Des problèmes de confidentialité et de sécurité des données se posent si des données propriétaires sont utilisées pour personnaliser des modèles d'IA générative. Des efforts doivent être déployés pour garantir que les outils d'IA générative génèrent des réponses qui limitent l'accès non autorisé à des données propriétaires. Des problèmes de sécurité se posent également en cas de manque de responsabilité et de transparence dans la manière dont les modèles d'IA prennent des décisions.

Créativité

Bien que l'IA générative puisse produire du contenu créatif, elle manque souvent de véritable originalité. La créativité de l'IA est limitée par les données sur lesquelles elle a été entraînée, ce qui donne lieu à des résultats qui peuvent sembler répétitifs ou dérivés. La créativité humaine, qui implique une compréhension plus approfondie et une résonance émotionnelle, reste difficile à reproduire pleinement pour l'IA.

Coût

L'entraînement et l'exécution de modèles d'IA générative nécessitent des ressources de calcul importantes. Les modèles d'IA générative basés sur le cloud sont plus accessibles et plus abordables que la création de nouveaux modèles à partir de zéro.

Explicabilité

En raison de leur nature complexe et opaque, les modèles d'IA générative sont souvent considérés comme des boîtes noires. Il est difficile de comprendre comment ces modèles aboutissent à des résultats spécifiques. Il est essentiel d'améliorer l'interprétabilité et la transparence pour renforcer la confiance et l'adoption.

1.9 Quelles sont les bonnes pratiques en matière d'adoption de l'IA générative ?

Si votre entreprise souhaite implémenter des solutions d'IA générative, tenez compte des bonnes pratiques suivantes pour appuyer les efforts mis en œuvre.

Commencez par les applications internes

Il est préférable de commencer l'adoption de l'IA générative par le développement d'applications internes, en mettant l'accent sur l'optimisation des processus et la productivité des employés. Vous bénéficiez d'un environnement plus contrôlé pour tester les résultats tout en développant les compétences et la compréhension de la technologie. Vous pouvez tester les modèles de manière approfondie et même les personnaliser sur des sources de connaissances internes. Ainsi, vos clients bénéficieront d'une bien meilleure expérience lorsque vous utiliserez les modèles pour des applications externes.

Améliorez la transparence

Expliquez clairement comment fonctionnent toutes les applications et tous les résultats de l'IA générative, afin que vos utilisateurs sachent qu'ils interagissent avec l'IA et non avec des humains. Par exemple, l'IA peut se présenter comme étant une IA, ou les résultats de recherche basés sur l'IA peuvent être marqués et mis en évidence. Ainsi, vos utilisateurs peuvent prendre des décisions éclairées lorsqu'ils interagissent avec le contenu. Ils peuvent également être plus proactifs dans le traitement des inexactitudes ou des biais cachés que les modèles sous-jacents peuvent présenter en raison des limites de leurs données d'apprentissage.

Implémentez la sécurité

Mettez en place des barrières de protection permettant à vos applications d'IA générative de refuser l'accès non autorisé aux données sensibles. Impliquez les équipes de sécurité dès le départ afin que tous les aspects puissent être pris en compte immédiatement. Par exemple, vous devrez peut-être masquer des données et supprimer des données d'identification personnelle (PII) avant d'entraîner des modèles sur des données internes.

Effectuez des tests exhaustifs

Développez des processus de test automatisés et manuels pour valider les résultats et testez tous les types de scénarios auxquels le système d'IA générative peut être confronté. Formez plusieurs groupes de bêta-testeurs pour essayer les applications de différentes manières et documenter les résultats. Le modèle s'améliorera également en continu grâce aux tests, et vous aurez un meilleur contrôle sur les résultats attendus et les réponses.

1.10 Amélioration de la pertinence des chatbots d'IA générative

Qu'est-ce qu'un modèle pré-entraîné ?

Quand on parle de modèles d'IA, on parle généralement d'algorithmes utilisant des réseaux de neurones. Cette technologie est utilisée par les *Larges Modèles de Langage* (LLM) comme *ChatGPT* ou *Mistral Large*, par les algorithmes de génération d'images ou de reconnaissance visuelle. Ces modèles sont entraînés sur des jeux de données très volumineux, leur permettant d'acquérir une base solide dans le domaine de compétences qu'ils vont maîtriser.

Il est possible de régler de nombreux paramètres sur les différentes couches d'un réseau pour modifier la manière dont est traitée l'information. On arrive à plusieurs milliards de paramètres lorsque les modèles sont vraiment volumineux (de quoi laisser de la place à la personnalisation...) !

1.10.1 RAG (Retrieval-Augmented Generation)

Le *Retrieval Augmented Generation* (RAG) est une approche novatrice qui combine le meilleur de deux mondes en IA : la recherche d'informations (*retrieval*, qui ne génère pas de réponse originale) et la génération de contenu (qui ne s'appuie que sur les données de son entraînement).

Traditionnellement, les *Large Language Models* (LLM) génèrent du contenu en s'appuyant uniquement sur les informations apprises durant leur phase d'entraînement. Le RAG, en

revanche, permet au modèle de « consulter » une base de données ou un corpus de documents externes en temps réel pour enrichir sa génération de texte. Cette capacité de recherche améliore significativement la précision, la pertinence et la richesse du contenu généré.

Comment fonctionne le RAG ?

Le processus du *Retrieval Augmented Generation* (RAG) peut être divisé en deux grandes étapes :

- **La récupération** : Lorsque le modèle reçoit une requête, il effectue une recherche dans un ensemble prédéfini de documents ou de données pour trouver les informations les plus pertinentes par rapport à la requête. Cette recherche est souvent facilitée par des techniques d'indexation et de récupération d'informations sophistiquées.
- **La génération** : Une fois les informations pertinentes récupérées, le modèle les utilise, en plus de sa propre connaissance interne, pour générer une réponse ou un contenu qui non seulement répond à la requête initiale mais le fait de manière plus informée et précise.

Avantages du RAG

L'intégration du *Retrieval Augmented Generation* dans les systèmes d'IA générative offre plusieurs avantages significatifs, améliorant non seulement la qualité du contenu généré mais aussi son applicabilité dans divers contextes.

Amélioration de la pertinence et de l'exactitude

En puisant dans une base de données externe pour compléter ses connaissances, un modèle RAG peut fournir des réponses plus précises et pertinentes aux questions posées. Cela est particulièrement utile pour les requêtes nécessitant des données actualisées ou spécifiques à un domaine.

Contenu plus riche et informé

La génération de contenu ne se contente plus de refléter les connaissances préalablement acquises pendant l'entraînement du modèle. Cela conduit à la création de textes plus informatifs, détaillés et nuancés.

Flexibilité et adaptabilité

Grâce à sa capacité à consulter une vaste gamme de sources, un modèle RAG peut s'adapter à une variété de sujets et de domaines. Cette flexibilité le rend particulièrement précieux pour les applications nécessitant une expertise dans des domaines de niche ou en constante évolution.

Interactivité améliorée

Le RAG permet aux systèmes d'interagir de manière plus sophistiquée avec les utilisateurs, en répondant à des questions complexes ou en fournissant des explications détaillées qui s'appuient sur des données et des sources externes.

Applications du RAG

Les applications du RAG sont multiples et permettent d'améliorer grandement la qualité du contenu généré. Voici quelques exemples illustrant comment cette approche peut s'intégrer à différents environnements professionnels :

- **Support client** : Le RAG permet de fournir des réponses personnalisées et précises aux requêtes des clients en accédant en temps réel à une base de données exhaustive, améliorant ainsi l'expérience client. Les systèmes de support équipés de RAG peuvent générer des FAQ dynamiques, répondant aux questions courantes avec des informations à jour, réduisant le volume de requêtes nécessitant une intervention humaine.
- **Génération de contenu** : Dans le marketing, le RAG peut être utilisé pour créer des articles, des billets de blog, et des descriptions de produits personnalisés pour le public cible, en s'appuyant sur des données de recherche pertinentes.
- **Ventes** : Le RAG peut dynamiser les stratégies de vente en créant des propositions commerciales sur mesure qui résonnent avec les besoins et préférences spécifiques des prospects. Basé sur l'analyse des interactions précédentes avec un client, le modèle peut générer des scripts de vente optimisés et des points de discussion pertinents.
- **Ressources humaines** : Le RAG facilite la rédaction de descriptions de poste attractives et précises, en s'inspirant des meilleures pratiques et des exemples de succès dans l'industrie. Il permet également de créer des FAQ internes détaillées, répondant efficacement aux questions des employés en se référant à des politiques et procédures actualisées, favorisant ainsi un environnement de travail informé et harmonieux.

Défis et limites

Malgré ses nombreux avantages, le RAG n'est pas sans défis. La qualité et la pertinence des informations récupérées dépendent fortement de la qualité du corpus de documents sous-jacent. De plus, l'intégration efficace des informations récupérées dans la génération de texte reste un défi technique non négligeable. Enfin, des considérations éthiques et de confidentialité entrent en jeu lorsqu'il s'agit de déterminer quelles informations peuvent être récupérées et utilisées par ces modèles.

1.10.2 Fine-Tuning ?

Contrairement à l'entraînement initial qui nécessite des jeux de données massifs comme ImageNet, cet affinage se concentre sur des données plus restreintes et spécialisées.

Il s'agit d'un processus itératif qui vise à améliorer la performance du modèle sur une tâche particulière, sans perdre les connaissances préalables acquises lors de l'entraînement initial.

L'idée centrale réside dans la capacité du modèle à généraliser à de nouveaux domaines tout en conservant sa capacité à se spécialiser.

Cette approche trouve des applications variées à travers de nombreux domaines comme :

- **Computer Vision** : un modèle peut être fine-tuné pour la détection d'objets spécifiques dans des contextes particuliers (véhicules autonomes, caméras de surveillance).
- **Médical** : un modèle peut être affiné pour la détection d'organes spécifiques dans les images médicales.
- **NLP** : un modèle peut être adapté pour la classification de documents juridiques, la détection de tonalité émotionnelle, ou la traduction automatique spécialisée.

Les étapes du processus de Fine-Tuning

Le Fine-Tuning suit une approche méthodique :

1. **Collecte et préparation des données** : données spécifiques, de haute qualité, nettoyées.
2. **Choix du modèle pré-entraîné** selon la tâche cible.
3. **Évaluation initiale** du modèle sur la tâche cible pour disposer d'une base de référence.
4. **Ajustement des hyperparamètres** (taux d'apprentissage, nombre d'itérations, taille du lot).

Pour optimiser ces hyperparamètres, on peut utiliser :

- la recherche aléatoire,
- la recherche en grille,
- l'optimisation bayésienne.

Les stratégies avancées d'affinage

Au-delà du simple ajustement, plusieurs stratégies avancées existent :

- **Transfert d'apprentissage** : utiliser la connaissance acquise sur une tâche pour une autre tâche similaire.
- **Fine-Tuning progressif** : affiner progressivement, en commençant par les couches supérieures, puis inférieures.
- **Techniques de régularisation** comme le dropout pour limiter le surajustement.

Une évaluation à chaque étape permet de détecter signes de surajustement ou de sous-ajustement.

Les meilleurs outils et bibliothèques de Fine-Tuning

Parmi les outils les plus utilisés :

- **TensorFlow** : fonctionnalités avancées de transfert d'apprentissage.
- **Keras** : interface haut niveau simplifiant le Fine-Tuning.
- **PyTorch** : flexibilité et manipulation aisée des couches du modèle.
- **Hugging Face Transformers** : bibliothèque de modèles pré-entraînés.

Pour la visualisation :

- **TensorBoard** pour TensorFlow,
- **TensorBoardX** pour PyTorch.

Les communautés GitHub et Stack Overflow offrent aussi de nombreux exemples, tutoriels et solutions aux problèmes courants.

Les défis du Fine-Tuning : difficultés à surmonter

Parmi les défis principaux :

- **Surajustement** : atténué par des techniques comme le dropout ou la normalisation par lots.
- **Déséquilibre des classes** : l'usage de poids de classe peut corriger ce biais.
- **Amplification des biais** : diversifier les sources de données et utiliser des techniques de correction de biais.

1.10.3 Le prompt engineering

Le prompt engineering, aussi appelé « ingénierie de requête », est une technique qui consiste à fournir des instructions détaillées aux modèles de traitement du langage naturel (Natural Language Processing, ou NLP) afin d'améliorer leurs performances.

Concrètement, le prompt engineering permet de guider plus précisément un modèle NLP en lui donnant des indications sur la tâche à effectuer et le contexte dans lequel elle s'inscrit. Plutôt que de laisser le modèle répondre de manière générique à une question posée, le prompt engineering permet de cadrer la réponse attendue.

Par exemple, il est possible d'utiliser le prompt engineering pour entraîner un modèle de génération de texte à rédiger des e-mails de relance client plus efficaces et personnalisés. Au lieu de lui donner comme simple instruction « Rédigez un e-mail de relance client », le prompt engineering doit fournir des détails : « Rédigez un e-mail de 200 mots pour relancer le client Jean Dupont qui n'a pas payé sa dernière facture depuis 30 jours. Utilisez un ton courtois et proposez une remise de 10% s'il paie dans les 7 prochains jours. » Ainsi, le prompt engineering permet de mieux contrôler et orienter les réponses d'un modèle NLP en fournissant un contexte, des exemples et des instructions claires.

Comment fonctionne le prompt engineering ?

L'utilisation du prompt engineering repose sur deux mécanismes complémentaires. D'une part, l'entraînement via des exemples commentés : cette technique consiste à fournir au modèle NLP des exemples concrets de prompts efficaces, accompagnés d'explications sur les raisons qui les rendent performants. Par exemple, les prompts donnés au modèle peuvent être conçus pour générer des slogans publicitaires impactants, en expliquant pourquoi ils cadrent bien la tâche attendue.

D'autre part, l'orientation pas à pas : le modèle est guidé étape par étape sur la façon de construire un prompt optimal. Il est, par exemple, possible de lui indiquer qu'un bon prompt doit définir clairement l'objectif, donner des exemples pertinents, établir le contexte, utiliser un vocabulaire simple, etc.

En combinant ces deux techniques, le modèle « apprend » à générer de meilleurs prompts de manière autonome. À force d'exemples commentés et d'orientation, le modèle finit par intégrer les bonnes pratiques du prompt engineering et les appliquer de lui-même.

L'enjeu est de fournir suffisamment d'informations dans le prompt pour que le modèle produise une réponse de qualité, mais sans non plus le surcharger d'instructions qui limitent sa créativité. Le prompt engineering cherche ainsi le bon équilibre entre guidage et autonomie.

Quels sont les différents types de prompt engineering ?

Il existe plusieurs façons d'aborder le prompt engineering en fonction du résultat recherché. Voici les trois grands types de prompt engineering :

- Le prompt engineering pour l'entraînement des modèles : il consiste à fournir au modèle NLP des prompts spécifiquement conçus pour l'entraîner sur une tâche donnée. Par exemple, lui donner des milliers de prompts illustrant la génération de textes publicitaires.
- Le prompt engineering pour l'inférence en production : une fois le modèle déployé, il est alimenté avec des prompts optimisés pour les cas d'usage visés. Cette méthode est utilisée pour les prompts de modération de contenu, de réponse aux questions clients, etc.
- Le prompt engineering adaptatif : le modèle NLP est entraîné à générer lui-même les meilleurs prompts en fonction du contexte. Cette approche a pour but de rendre

le modèle capable d'un prompt engineering autonome.

Chacune de ces variantes présente des spécificités en termes de méthodologie. Le prompt engineering pour l'entraînement met l'accent sur la diversité et la quantité de données. Celui pour l'inférence se concentre sur la qualité des prompts en contexte réel. L'approche adaptative cherche à donner au modèle la capacité d'apprendre à faire un prompt engineering efficace par lui-même.

Les chercheurs combinent souvent ces différents types de prompt engineering de manière complémentaire pour obtenir les meilleures performances. Cette diversité des techniques de prompt engineering est une force pour cette discipline encore émergente.

Quelles compétences requiert la maîtrise du prompt engineering ?

Réussir dans le prompt engineering, sans pour autant être un prompt engineer (ingénieur de requête), requiert de solides compétences pluridisciplinaires.

Tout d'abord, une connaissance approfondie des modèles de traitement du langage naturel est indispensable pour créer des prompts adaptés à leurs forces et à leurs limites.

La rédaction est également cruciale, car le prompt se doit d'être un texte concis, précis et cohérent. De bonnes capacités rédactionnelles sont donc essentielles.

La maîtrise de techniques d'apprentissage automatique comme le transfer learning est aussi un atout, celles-ci étant souvent employées conjointement au prompt engineering.

De même, une expertise en linguistique computationnelle permet de mieux appréhender les mécanismes d'interprétation du langage par les modèles.

Côté technique, la programmation et la data science sont nécessaires pour intégrer et évaluer les performances des prompts. La rigueur dans les tests et les expérimentations permet d'identifier les prompts les plus efficaces.

Enfin, la créativité et un esprit critique sont des qualités essentielles pour le prompt engineering.

Quels sont les avantages du prompt engineering ?

L'utilisation du prompt engineering présente plusieurs avantages. Tout d'abord, cette technique permet d'obtenir des réponses plus précises et pertinentes de la part des modèles NLP. Avec un prompt bien conçu, le modèle va générer une réponse correspondant exactement à la tâche demandée, et non pas quelque chose de vague ou d'approximatif. Le prompt engineering assure également une meilleure adéquation entre la réponse du modèle et les besoins spécifiques de l'utilisateur, en cadrant la génération de texte en fonction du cas d'usage visé et des instructions fournies. Par ailleurs, un prompt détaillé va limiter les contresens et mauvaises interprétations du modèle NLP, réduisant ainsi les biais et les erreurs. Le prompt engineering favorise aussi une plus grande cohérence des réponses du modèle, même face à des inputs différents.

À terme, cette technique améliore les performances globales des modèles NLP en renforçant leur capacité à comprendre des demandes complexes. Enfin, le prompt engineering offre une meilleure contrôlabilité de ces modèles, en permettant à leurs concepteurs d'orienter finement leur comportement via les consignes fournies.

Quelles sont les limites du prompt engineering ?

Bien que présentant des avantages certains, l'utilisation du prompt engineering comporte également des limites qu'il faut garder à l'esprit. Tout d'abord, il s'agit d'une technique qui nécessite un investissement temps important ainsi que de solides compétences en conception de prompts pour élaborer des instructions optimales. Sans une expertise approfondie, il est assez facile de rédiger de mauvais prompts qui vont en réalité dégrader

les performances du modèle NLP au lieu de les améliorer.

De plus, le prompt engineering est fortement dépendant du cas d'usage : des prompts spécifiques à une tâche devront être entièrement réécrits en cas de changement d'objectif ou de contexte. Par ailleurs, des prompts trop directifs ou détaillés peuvent réduire la créativité et l'autonomie des modèles NLP en les enfermant dans des schémas de réponse prédéfinis et stéréotypés. L'équilibre subtil entre guidage et liberté accordée au modèle est difficile à trouver.

Le processus de conception de prompts efficaces s'appuie également sur de nombreux essais-erreurs, des ajustements progressifs et de fins réglages : un prompt optimal ne peut être obtenu immédiatement.

Enfin, la complexité des prompts tend à croître à mesure qu'on cherche à les optimiser, jusqu'à potentiellement devenir contre-productifs s'ils sont surchargés d'instructions. Malgré son potentiel prometteur, le prompt engineering ne constitue donc pas une solution magique aux limitations des modèles NLP actuels.

Comment le prompt engineering va-t-il évoluer ?

Même s'il s'agit d'une technologie déjà mature, le prompt engineering devrait encore progresser à l'avenir, notamment grâce aux avancées dans les domaines connexes du transfer learning et du few-shot learning.

Plusieurs évolutions sont à anticiper :

- L'émergence de nouvelles méthodes pour générer des prompts plus efficaces, via l'utilisation d'algorithmes d'optimisation par exemple ;
- Le développement de bibliothèques de prompts pré-écrits, réutilisables d'un modèle NLP à l'autre ;
- La possibilité pour les modèles d'auto-générer les prompts les plus adaptés en fonction de la tâche demandée ;
- L'utilisation du prompt engineering pour entraîner des modèles NLP toujours plus puissants, atteignant des niveaux de compétence linguistique proches de l'humain ;
- L'écriture de prompts par les utilisateurs finaux eux-mêmes, permettant de guider les modèles NLP en langage naturel ;
- Une intégration accrue entre prompt engineering et techniques de search.

Grâce à ces avancées, le prompt engineering a le potentiel de devenir une technique incontournable pour exploiter au maximum les capacités des modèles de traitement du langage naturel sur une grande diversité de cas d'usage.

1.10.4 Few Shot Learning ?

Le Few Shot Learning (FSL) est un framework du domaine du Machine Learning, c'est-à-dire une structure de base pour le développement de codes de programmation. Il est utilisé pour entraîner les modèles d'IA à faire des prédictions précises à partir d'une petite quantité de données d'entraînement. Alors que les méthodes de Machine Learning traditionnelles nécessitent souvent des milliers de points de données pour fournir des résultats fiables, le Few Shot Learning vise à optimiser l'apprentissage avec un minimum de données.

L'objectif principal du Few Shot Learning est un apprentissage efficace à partir de quelques exemples seulement. En travaillant avec une quantité minimale de données, le FSL s'avère particulièrement utile dans les situations où il est difficile de collecter de grandes quantités de données étiquetées (ex : maladies rares, manuscrits uniques).

Le Few Shot Learning peut être considéré comme un sous-groupe du n-Shot-Learning (N-Way-K-Shot), où : - N est le nombre de classes, - K est le nombre d'exemples par classe.

Cela comprend également : - le One Shot Learning (un exemple par classe), - le Zero Shot Learning (aucun exemple étiqueté).

Comment fonctionne le Few Shot Learning ?

Même si des algorithmes spéciaux et des réseaux neuronaux réussissent de nombreuses tâches de FSL, il est avant tout défini par le **problème d'apprentissage** spécifique, et non par un modèle particulier.

Les principales approches sont :

Apprentissage par transfert

Il s'agit d'adapter des modèles pré-entraînés pour de nouvelles tâches, en utilisant les connaissances déjà acquises pour éviter le surajustement, surtout avec peu d'exemples. Cela fonctionne particulièrement bien si la tâche cible est similaire à la tâche d'origine.

Approche par les données

Il s'agit de **générer des données supplémentaires** pour pallier le manque d'exemples, souvent grâce à des réseaux génératifs (GANs). Cela est crucial dans des cas rares (ex : espèces nouvellement découvertes).

Meta Learning (Méta-apprentissage)

Le modèle apprend à apprendre : il généralise des schémas à travers différentes tâches. On distingue deux types :

Méta-apprentissage basé sur les métriques Plutôt que de modéliser des classes directement, on apprend des distances entre exemples : - Réseaux siamois - Réseaux de matching - Réseaux prototypiques - Réseaux de relation (Relation Networks)

Méta-apprentissage basé sur l'optimisation On optimise les modèles pour qu'ils s'adaptent rapidement à de nouvelles tâches : - MAML (Model Agnostic Meta-Learning) - Optimisation par LSTM - LEO (Latent Embedding Optimization)

Principaux domaines d'application du Few Shot Learning

Les secteurs concernés sont nombreux :

- **Computer Vision** : classification d'images, reconnaissance d'objets.
- **Robotique** : adaptation rapide à de nouveaux environnements.
- **Traitements du langage** : adaptation de grands modèles de langage.
- **Santé publique** : diagnostic de maladies rares.
- **Secteur bancaire** : détection de fraudes avec peu d'exemples.

Enjeux concrets liés à la mise en œuvre du Few Shot Learning

Le Few Shot Learning présente plusieurs défis :

- **Surajustement** : le risque est élevé avec peu de données.

- **Qualité des données** : si les données sont de mauvaise qualité, la performance chute.
- **Sélection de caractéristiques** : difficile avec peu d'exemples.
- **Ressources de calcul** : optimisation fine requiert du temps et des ressources.

1.10.5 Low Rank Adaptation

Cette approche novatrice se concentre sur la réduction de la dimensionnalité des données, dans le but de faciliter l'adaptation de domaine : le transfert d'un modèle depuis un domaine source où il est entraîné vers un domaine cible où les données peuvent être différentes.

Un concept central de cette méthode est la décomposition à faible rang des matrices. Celle-ci permet de représenter les données sous une forme plus compacte, tout en préservant leur structure sous-jacente.

Dans le contexte de l'apprentissage automatique, cette décomposition vise à extraire les caractéristiques les plus importantes des données d'entraînement. Le but est de construire un modèle plus généralisable pour le domaine cible.

Il existe différentes méthodes pour y parvenir. Les plus utilisées sont la décomposition en valeurs singulières (Singular Value Decomposition ou SVD) et la décomposition en facteurs non négatifs (Non-Negative Matrix Factorization ou NMF).

Grâce à la réduction de dimensionnalité des données, on évite les problèmes de surajustement (overfitting) pouvant survenir quand les modèles sont directement appliqués à des données de cibles différentes. C'est ce qui simplifie le transfert de connaissances.

L'un des principaux points forts de la Low Rank Adaptation est sa capacité à capturer les corrélations et dépendances entre les caractéristiques des données. Ainsi, l'information cruciale pour l'adaptation est mieux représentée.

Un autre avantage est la possibilité d'appliquer cette approche à différentes tâches de Machine Learning comme la classification, la régression ou même la génération de données synthétiques.

À quoi ça sert ? Quelles sont les applications ?

La Low Rank Adaptation est une méthode qui a fait ses preuves, et qui est déjà adoptée dans de nombreux domaines du Machine Learning.

Dans la vision par ordinateur (Computer Vision), elle est utilisée pour la reconnaissance d'objets au sein d'environnements différents de ceux utilisés pour l'entraînement initial. En utilisant des décompositions à faible rang sur les représentations des caractéristiques des images, les modèles ML peuvent mieux généraliser à de nouvelles conditions d'éclairage, d'angle de vue ou d'environnement pour améliorer considérablement leurs performances dans le monde réel.

Pour le NLP ou traitement naturel du langage, cette approche s'avère aussi très efficace pour les tâches de traduction automatique.

Alors que les données des domaines sources et cibles peuvent considérablement varier, l'adaptation à faible rang permet de créer des modèles plus flexibles en extrayant les aspects linguistiques essentiels et en les appliquant au nouveau domaine. La traduction dans des contextes différents s'en trouve améliorée.

De même, pour la reconnaissance de la parole, la Low Rank Adaptation est utile pour adapter les modèles de reconnaissance vocale à des locuteurs spécifiques ou à différents environnements acoustiques.

Les techniques de décomposition à faible rang permettent aux systèmes de mieux capturer les variations entre les interlocuteurs et donc d'obtenir une meilleure précision dans des situations variées.

Enfin, en intégrant cette méthode d'adaptation aux architectures de réseaux de neurones, l'apprentissage par transfert est facilité. Les connaissances acquises dans un domaine sont transmises à un domaine cible, augmentant fortement les performances pour une large variété de tâches.

Les défis de la Low Rank Adaptation

Tous les exemples évoqués dans le chapitre précédent démontrent l'efficacité de la Low Rank Adaptation pour surmonter les défis liés à l'adaptation de domaine dans le Machine Learning.

Malgré ses nombreux avantages, cette approche présente aussi des défis spécifiques à relever. Par exemple, la gestion du surajustement (overfitting) et du sous-ajustement (underfitting) des modèles peut être difficile.

Une décomposition à faible rang trop restrictive peut entraîner une perte d'informations cruciales, tandis qu'une décomposition trop complexe peut conduire à la suradaptation aux données d'entraînement. Trouver le bon équilibre est donc essentiel pour l'optimisation des performances sur de nouvelles données.

Autre problème potentiel : les domaines sources et cibles peuvent être extrêmement différents, ce qui entraîne une hétérogénéité des données compliquant fortement l'adaptation. La diversité des caractéristiques entre les deux domaines peut en effet affecter la capacité du modèle à transférer des connaissances de manière appropriée.

Pour éviter ce désagrément, il convient de développer des techniques de décomposition à faible rang capables de capturer les variations tout en conservant les structures essentielles.

Enfin, dans de nombreuses situations, il n'est pas toujours évident d'obtenir des données étiquetées dans le domaine cible. Elles peuvent être limitées, ou tout simplement trop coûteuses.

Or, la Low Rank Adaptation requiert généralement une large quantité de données pour construire des modèles généralisables. Ceci exige des recherches supplémentaires pour développer des méthodes exploitant les données non étiquetées ou semi-étiquetées, comme l'adaptation à faible rang transductive.

1.10.6 Comparaison des Techniques

TABLE 1.1 – Comparaison des principales techniques d'optimisation de modèles d'IA

Technique	Avantages	Limitations
RAG (Retrieval-Augmented Generation)	<ul style="list-style-type: none"> - Accès à des informations actualisées - Améliore la pertinence des réponses 	<ul style="list-style-type: none"> - Dépendance aux bases de données - Complexité de mise en œuvre
Fine-Tuning	<ul style="list-style-type: none"> - Modèle spécialisé pour une tâche précise - Meilleure performance sur des cas spécifiques 	<ul style="list-style-type: none"> - Coûteux en calculs - Nécessite beaucoup de données
Prompt Engineering	<ul style="list-style-type: none"> - Pas besoin de réentraîner le modèle - Rapide à mettre en place 	<ul style="list-style-type: none"> - Résultats instables - Fortement dépendant de la qualité du prompt
Few-Shot Learning	<ul style="list-style-type: none"> - Apprentissage avec peu d'exemples - Réduit le besoin de grandes bases de données 	<ul style="list-style-type: none"> - Moins performant qu'un entraînement complet - Peut être sensible au choix des exemples
LoRA (Low-Rank Adaptation)	<ul style="list-style-type: none"> - Fine-tuning léger - Moins de ressources nécessaires 	<ul style="list-style-type: none"> - Peut être moins performant que le fine-tuning complet
Distillation	<ul style="list-style-type: none"> - Modèle plus léger et rapide - Adapté aux environnements contraints (mobile, embarqué) 	<ul style="list-style-type: none"> - Perte de performance par rapport au modèle original

Conclusion

Dans ce chapitre, nous avons découvert l'univers fascinant de l'intelligence artificielle générative. Cette technologie, qui permet de créer du contenu original (textes, images, sons...), repose sur des modèles puissants comme les autoencodeurs variationnels (VAE), les réseaux antagonistes génératifs (GAN) ou encore les transformateurs.

Nous avons vu que l'IA générative offre énormément d'opportunités, mais qu'elle comporte aussi des risques et des limites. Elle peut révolutionner des domaines comme l'art, la santé ou l'éducation, tout en soulevant des questions importantes autour de l'éthique, de la fiabilité et de l'utilisation abusive.

Grâce aux nouvelles techniques comme le fine-tuning, le prompt engineering ou le few-shot learning, il est possible d'améliorer encore la pertinence et la qualité des résultats produits. Mais pour en tirer le meilleur parti, il est essentiel de l'utiliser avec précaution et de garder à l'esprit l'importance de l'humain dans le processus.

L'IA générative n'en est qu'à ses débuts. Elle continuera d'évoluer et de transformer notre quotidien. À nous de savoir l'accompagner de manière responsable, en restant attentifs aux défis qu'elle pose et en utilisant son potentiel pour construire des outils plus créatifs, plus justes et plus utiles.

Chapitre 2

Système de recommandation

Introduction

Les systèmes de recommandation sont une forme spécifique de filtrage d'information visant à présenter les éléments d'information qui sont susceptibles d'intéresser l'utilisateur. Ils peuvent être définis comme des programmes qui tentent de recommander les articles (produits ou services) les mieux adaptés à des utilisateurs particuliers (particuliers ou entreprises) en prévoyant l'intérêt d'un utilisateur pour un article en fonction d'informations connexes sur les articles, les utilisateurs et les interactions entre eux (Bobadilla et al., 2013).

L'élaboration de systèmes de recommandation a pour objectif de réduire la surcharge d'informations en récupérant les informations et les services les plus pertinents à partir d'une quantité énorme de données, fournissant ainsi des services personnalisés. La caractéristique la plus importante d'un système de recommandation est sa capacité à « deviner » les préférences et les intérêts d'un utilisateur en analysant le comportement de cet utilisateur et / ou celui d'autres utilisateurs pour générer des recommandations personnalisées (Resnick & Varian, 1997).

Les systèmes de recommandation ont été définis de plusieurs façons. La définition la plus populaire et la plus générale que nous citons ici est celle de Robin Burke (Burke, 2002) : Un système de recommandation est : « un système capable de fournir des recommandations personnalisées ou permettant de guider l'utilisateur vers des ressources intéressantes ou utiles au sein d'un espace de données important ».

Dans tout système de recommandation, il existe deux entités qui représentent le cœur du système. Toute technique de recommandation tourne autour de ces entités qui sont : les utilisateurs et les items.

L'utilisateur est la personne qui interagit avec le système, à qui le système recommande des items et à son tour donne son avis sur les items.

L'item est le terme général utilisé pour désigner la ressource que le système suggère aux utilisateurs.

Avec l'utilisateur et l'item, le domaine d'information d'un système de recommandation contient aussi une préférence exprimée par un utilisateur pour un item qui est appelée note, et est souvent représentée par un triplet (utilisateur ; item ; note). Ces notes peuvent prendre différentes formes. Cependant, la majorité des systèmes utilisent des notes sous forme d'une échelle de 1 à 5, ou bien des notes binaires (j'aime/je n'aime pas). L'ensemble des triplets (utilisateur ; item ; note) forme ce que l'on appelle la matrice des notes. Les paires (utilisateur ; item) où l'utilisateur n'a pas donné de note pour l'item sont des valeurs non connues dans la matrice.

Dans ce chapitre, nous utiliserons les notations suivantes concernant les différents éléments du modèle d'un système de recommandation :

$$r_{\bar{u}} = \frac{\sum_{i \in I_u} r_{u,i}}{|I_u|} \quad (1.1)$$

$$r_{\bar{i}} = \frac{\sum_{u \in U_i} r_{u,i}}{|U_i|} \quad (1.2)$$

2.1 Classification des systèmes de recommandation

Les techniques de recommandation peuvent être classées de différentes manières. Parfois plusieurs termes sont utilisés pour désigner une même méthode ou approche. La classification la plus utilisée repose sur trois types : filtrage basé sur le contenu, filtrage collaboratif et filtrage hybride (Adomavicius & Tuzhilin, 2005). En plus de ces deux approches, Robin Burke (Burke, 2002) propose de considérer trois autres approches : la recommandation basée sur les données démographiques, la recommandation basée sur la connaissance (knowledge-based) et la recommandation basée sur l'utilité (utility-based). Mais il note que ces trois approches sont des cas particuliers des approches classiques. L'objectif ici est de s'appuyer sur la classification la plus connue sur laquelle nous basons notre étude. Nous présentons dans la suite les approches basées sur le contenu et le filtrage collaboratif, puis les approches hybrides (figure 1.1). Nous allons détailler la technique de filtrage collaboratif que nous allons utiliser dans notre travail.

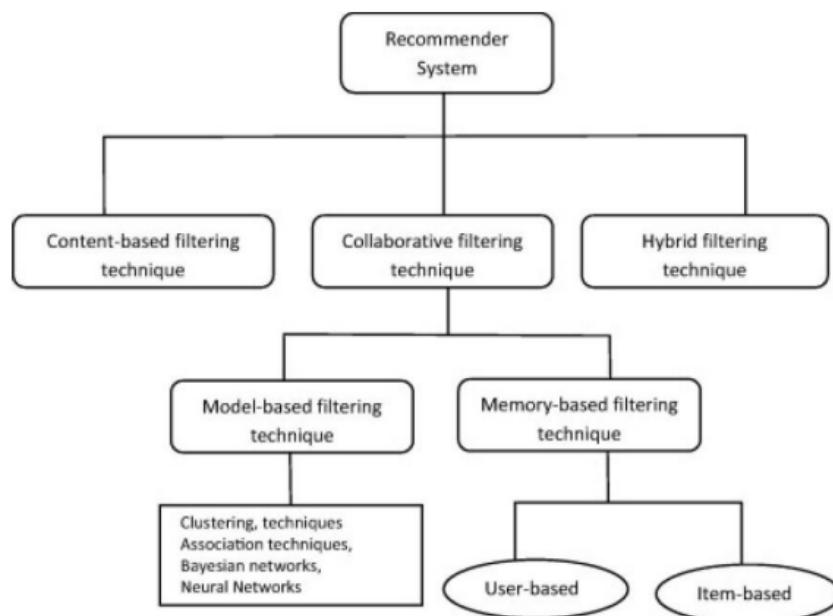


FIGURE 2.1 – Classification des systèmes de recommandation (Isinkaye et al., 2015)

2.1.1 Le filtrage basé sur le contenu

Un système qui utilise le filtrage basé contenu exploite seulement les représentations des documents et les informations qui peuvent être dérivées de ces documents. Un tel type de filtrage pourrait par exemple utiliser la similarité des documents dans une matrice termes-documents pour déterminer la pertinence d'un document. Si un utilisateur exprime un intérêt pour un document, les documents similaires seront jugés potentiellement pertinents aussi. C'est d'ailleurs la technique que nous allons adopter dans notre propre système, FCRC.

Pour mieux comprendre cette approche, nous allons étudier l'exemple du système PRES (acronym for Personalized Recommender System) PRES [Maarten et Someren, 2000] qui est un système de recommandation basé-contenu. PRES crée des hyperliens dynamiques pour un site web qui contient une collection de conseils pour l'amélioration de la personnalité. Le but de ces hyperliens dynamiques est d'aider les utilisateurs à trouver des items pertinents.

La figure 2.2 présente une structure simple du site web où le contenu des pages représente les feuilles de l'arbre tandis que les pages de navigation se trouvent en haut. PRES affiche ses recommandations par la création dynamique des liens hypertextes vers des contenus de pages qui contiennent les items pertinents.

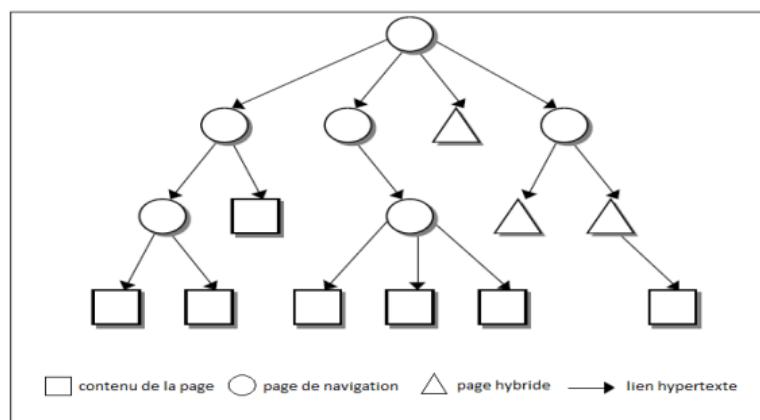


FIGURE 2.2 – Structure en arbre d'un simple site web.PRES [Maarten et Someren, 2000]

Un site Web peut être structuré en divisant ses pages web en des pages de contenu et des pages de navigation (voir figure 2.2). Une page de contenu fournit à l'utilisateur les items pertinents tandis qu'une page de navigation aide l'utilisateur à les rechercher. Les pages peuvent également être hybrides, elles fournissent à la fois le contenu ainsi que des fonctionnalités de navigation. Une page de navigation pour un utilisateur peut être une page de contenu pour un autre et vice versa.

PRES traite les données et classifie les items dans une classe positive nommée C (items pertinents). Par la suite il utilise les corrélations entre les contenus des items et les préférences des utilisateurs pour déterminer le degré de similarité.

PRES affiche ses recommandations dynamiquement par la création des liens hypertextes vers des pages de contenu qui contiennent les items pertinents. Il fait des recommandations en comparant chaque profil d'utilisateur avec le contenu de chaque document de la collection. La similarité entre le vecteur de profil et un document est déterminée par l'utilisation de la mesure du cosinus dans un espace vectoriel <terme, contenu du site web>.

Quelques limites des systèmes de filtrage basé contenu sont présentées par Berrut et Denos dans [Berrut et Denos, 2003]. La première difficulté pour ce type des systèmes c'est qu'il ne traite que des documents textes, nous ne pouvons pas indexer des documents multimédia. De plus les systèmes basés contenu ne considèrent que le facteur thématique comme critère de pertinence malgré qu'il existe d'autres facteurs tels que : la fiabilité de

la source d'information, la qualité scientifique et le degré de précision des faits présentés. Les auteurs évoquent aussi l'effet « entonnoir » qui restreint le champ de vision de l'utilisateur : à force de ne recommander que des items reliés à une seule et même thématique, et du fait que l'intérêt de l'utilisateur est déterminé à partir des items choisis, on crée un effet d'auto-entraînement qui restreint toujours plus la définition des intérêts de l'utilisateur et empêche l'exploration de thématiques différentes.

2.1.2 Le filtrage collaboratif

Le filtrage collaboratif est un parmi les technologies les plus populaires dans le domaine des systèmes de recommandation [?]. Le filtrage collaboratif se base sur l'idée que les personnes à la recherche d'information devraient se servir de ce que d'autres ont déjà trouvé et évalué. Dans la vie quotidienne, si quelqu'un a besoin d'une information, il essaye de s'informer généralement auprès de ses amis, ses collègues, qui vont à leurs tours lui recommander des articles, des films, des livres, etc. Cette collaboration entre les gens permet d'améliorer l'échange de connaissances, mais cela prend beaucoup de temps vu que cette ressource d'information ne peut pas toujours être à notre disposition. C'est à partir d'ici que l'idée de filtrage collaboratif est née, le besoin étant d'automatiser et de rendre l'échange des expériences et des avis personnels de certaines personnes utilisables par d'autres. Selon Golberg [?], le filtrage collaboratif est l'automatisation des processus sociaux.

À la base du filtrage collaboratif, on utilise les choix explicites ou implicites des utilisateurs pour des items. Donc plutôt que d'utiliser une matrice termes-documents comme la recherche d'information le fait et comme l'approche basée sur le contenu, le filtrage collaboratif utilise la matrice des votes utilisateurs-items, où un vote peut être soit explicite (ex. fournir une cote à un film), soit implicite (ex. acheter un DVD du film).

Le tableau 2.1 illustre un exemple de matrice utilisateurs-items. Les valeurs peuvent représenter des votes ou des comportements. Dans cet exemple, l'item I3 n'a pas de vote pour l'utilisateur U1. On peut tenter de l'estimer soit avec une approche item-item ou une approche utilisateur-utilisateur comme expliqué dans les sections qui suivent.

TABLE 2.1 – Matrice Items-utilisateurs

Utilisateur	Item I1	Item I2	Item I3	Item I4
U1	5	1	?	2
U2	4	1	0	3
U3	4	2	1	2
U4	1	4	3	2

Pour mieux définir ce type de filtrage, on se réfère aux travaux de Breese, Heckerman et Kadie dans [Breese, Heckerman et Kadie, 1998].. Ils décrivent plusieurs algorithmes conçus pour cette tâche, comprenant des techniques basées sur les coefficients de corrélation, le calcul de similarité basé sur les vecteurs et des méthodes bayésiennes statistiques. Le filtrage collaboratif peut prendre plusieurs formes : Item-Item et Utilisateur-Utilisateur.

2.1.3 Filtrage item-item (Linden et al., 2003)

Le filtrage item-item calcule la similarité des items dans la matrice utilisateur-items. Afin de déterminer les items les plus similaires à un item donné, cet algorithme construit un tableau des items similaires en cherchant les items populaires (que les clients ont tendance à acheter). Une matrice des produits entre les items est construite, ces produits représentant le degré de similarité entre les items (paire par paire). On peut trouver des paires qui n'ont aucun client en commun.

Dans le but de conserver le temps d'exécution et l'utilisation de la mémoire, il sera beaucoup mieux de calculer le produit d'un item par rapport aux autres items, au lieu de calculer les produits de tous les items à la fois.

La similarité entre les items peut être calculée de différentes méthodes, mais la méthode la plus commune est la mesure de cosinus des items.

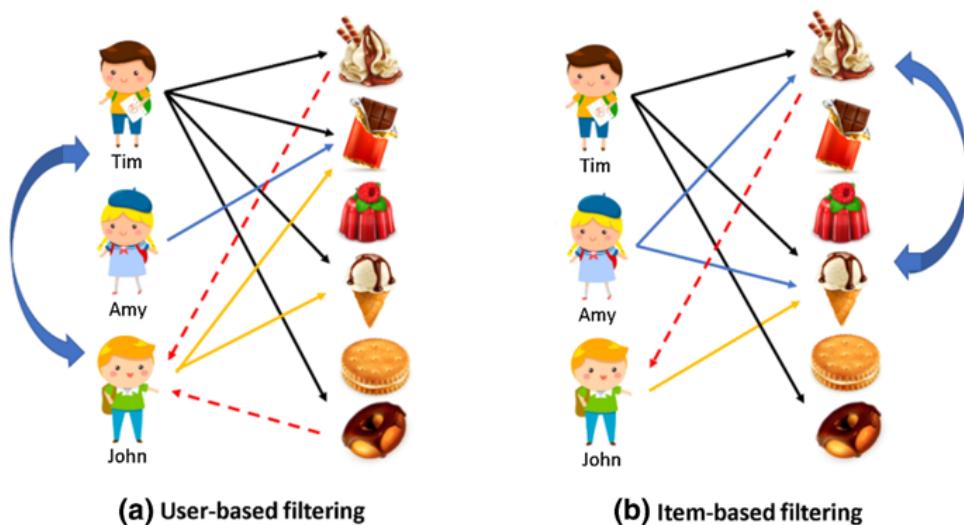


FIGURE 2.3 – Filtrage collaboratif basé sur les éléments.

2.1.4 Filtrage utilisateur-utilisateur

Les filtres collaboratifs utilisent des bases de données sur les préférences des utilisateurs. Généralement, un utilisateur est caractérisé par ses goûts, ses préférences et ses choix, à travers ses traces (des achats, des consultations de pages, des votes, etc.).

L'algorithme utilisateur-utilisateur cherche la similarité entre les profils des utilisateurs et recommande des items à un nouvel utilisateur. Ces recommandations sont basées sur le groupe le plus proche et le plus représentatif d'un individu.

Le principe de cette approche est de chercher la similarité entre les profils utilisateurs en se basant sur leurs préférences et leurs appréciations.

Le tableau 1.2 représente les valeurs des votes donnés par quatre utilisateurs : U1, U2, U3 et U4, pour quatre items. L'utilisateur U1 n'a pas encore voté pour l'item I3. L'algorithme utilisateur-utilisateur permet d'estimer le vote de cet utilisateur U1 pour l'item I3 en fonction des votes des autres utilisateurs.

La valeur estimée E de l'utilisateur U1 pour un item I3 est la somme pondérée des votes des autres utilisateurs V_i , qui ont des votes communs, où i est l'index de chaque vote.

Ainsi, pour estimer le vote pour l'item I3, on utilise la formule suivante :

$$v_{ij} = \text{moy}(v_i) + k \sum w_i v_i$$

où :

- v_{ij} : vote de l'utilisateur i pour l'item j ,
- $\text{moy}(v_i)$: vote moyen de l'utilisateur i ,
- k : constante pour normaliser la somme des w_i à 1 (par exemple, si le poids est un cosinus, ce sera la somme des cosinus),
- w_i : poids de l'utilisateur i (ce poids peut être le cosinus entre le vecteur ligne de l'utilisateur pour lequel on désire estimer le vote et l'utilisateur i ; cela peut aussi être la corrélation ou une autre mesure),
- m : nombre de votes.

2.1.5 Le filtrage hybride

Le filtrage hybride combine les deux types de filtrage : basé sur le contenu et le filtrage collaboratif.

2.2 Évaluation des systèmes de recommandation

L'évaluation des systèmes de recommandation est cruciale pour mesurer leur efficacité et leur pertinence. Plusieurs méthodes existent pour évaluer les performances des algorithmes de recommandation. Ces méthodes peuvent être classées en différentes catégories selon le type de données, les objectifs de la recommandation, et la manière dont on mesure la qualité des recommandations. Nous détaillerons les principales méthodes utilisées pour évaluer ces systèmes.

2.2.1 Mesures de précision des recommandations

Les **mesures de précision** sont souvent utilisées pour évaluer la qualité des prédictions de score ou de note (rating) dans un système de recommandation.

2.2.1.1 RMSE (Root Mean Squared Error)

Le RMSE mesure l'erreur quadratique moyenne entre les valeurs prédites par le système et les valeurs réelles. La formule du RMSE est la suivante :

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2}$$

où y_i représente la valeur réelle et \hat{y}_i la valeur prédite pour l'élément i .

Avantages : Le RMSE pénalise fortement les grandes erreurs, ce qui est utile pour les systèmes où la précision des prédictions est essentielle.

2.2.1.2 MAE (Mean Absolute Error)

Le MAE est une autre mesure d'erreur, mais qui prend la moyenne des écarts absolus. Cette métrique est plus robuste que le RMSE lorsqu'il y a des erreurs très grandes.

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

Avantages : MAE donne une idée générale de la moyenne des erreurs de prédiction.

2.2.2 Mesures de classification des recommandations

Les **mesures de classification** sont utilisées pour évaluer dans quelle mesure le système recommande des éléments pertinents.

2.2.2.1 Précision (Precision)

La précision mesure la proportion d'éléments correctement recommandés parmi tous ceux qui ont été recommandés par le système.

$$Precision = \frac{\text{Nombre de recommandations correctes}}{\text{Nombre total de recommandations}}$$

Avantages : La précision est utile pour les systèmes où il est important de recommander uniquement des éléments pertinents.

2.2.2.2 Rappel (Recall)

Le rappel mesure la proportion des éléments pertinents qui ont été recommandés parmi tous les éléments pertinents disponibles.

$$Rappel = \frac{\text{Nombre de recommandations correctes}}{\text{Nombre total d'éléments pertinents}}$$

Avantages : Le rappel est essentiel lorsque l'on veut minimiser les risques de ne pas recommander des éléments pertinents.

2.2.2.3 F1-Score

Le F1-score est la moyenne harmonique entre la précision et le rappel.

$$F1 = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

Avantages : Cette mesure combine à la fois la précision et le rappel, et est utilisée lorsqu'on veut un compromis entre les deux.

2.2.3 Mesures de classement des recommandations

Les **mesures de classement** évaluent la position des éléments pertinents dans la liste de recommandations.

2.2.3.1 NDCG (Normalized Discounted Cumulative Gain)

L'NDCG mesure l'efficacité d'un système en tenant compte de la position dans laquelle les éléments pertinents apparaissent. Un élément pertinent placé plus haut dans la liste est considéré comme plus utile.

Avantages : Permet d'évaluer les systèmes de recommandation en prenant en compte l'ordre des résultats.

2.2.3.2 MRR (Mean Reciprocal Rank)

Le MRR est une autre mesure de classement, qui se concentre sur la position du premier élément pertinent dans la liste. La formule est :

$$MRR = \frac{1}{N} \sum_{i=1}^N \frac{1}{r_i}$$

où r_i est la position du premier élément pertinent pour l'élément i .

Avantages : Utilisé lorsque la priorité est donnée aux premiers éléments pertinents dans les recommandations.

2.2.4 Autres critères d'évaluation

Outre les métriques de précision et de classement, d'autres critères peuvent être pris en compte pour évaluer un système de recommandation.

2.2.4.1 Couverture

La couverture mesure la proportion d'éléments dans l'ensemble total qui peuvent être recommandés par le système.

Avantages : Permet de voir si le système couvre bien une large gamme de produits ou services.

2.2.4.2 Diversité

La diversité évalue si le système recommande des éléments variés ou s'il se limite à recommander des éléments similaires.

Avantages : Une bonne diversité évite la monotonie et augmente l'expérience utilisateur.

2.2.4.3 Nouveauté

La nouveauté mesure dans quelle mesure les éléments recommandés sont nouveaux pour l'utilisateur, c'est-à-dire qu'ils ne sont pas encore connus.

Avantages : Permet d'éviter de simplement recommander les éléments populaires, mais plutôt de surprendre l'utilisateur avec des nouveautés.

2.3 Limites des systèmes de recommandation

Malgré les progrès réalisés dans le domaine des systèmes de recommandation, plusieurs défis et limitations subsistent. Ces limitations peuvent affecter la performance des systèmes, leur capacité à personnaliser les recommandations et l'expérience utilisateur. Parmi les principales limites, on trouve les problèmes du *cold start*, de la *sparsité*, et du *surapprentissage*. Nous détaillons ces limitations ci-dessous.

2.3.1 Problème du Cold Start

Le *cold start* est un problème majeur qui survient lorsque le système de recommandation doit générer des recommandations pour un utilisateur ou un produit nouvellement ajouté au système, pour lesquels il n'existe pas encore suffisamment de données historiques. Ce problème peut être divisé en trois catégories principales :

- **Cold start utilisateur** : Lorsque de nouveaux utilisateurs s'inscrivent dans le système, il n'y a pas de données sur leurs préférences ou comportements passés, ce qui rend difficile la génération de recommandations pertinentes. Cela peut entraîner une expérience utilisateur moins satisfaisante.
- **Cold start produit** : De nouveaux produits ou articles ajoutés au catalogue du système ne bénéficient pas encore d'évaluations ou de comportements d'utilisateur, ce qui les rend difficiles à recommander.
- **Cold start général** : Lorsque les deux (utilisateur et produit) sont nouveaux, le système fait face à un problème de manque de données pour établir des liens ou des similarités entre utilisateurs et produits.

Solutions possibles : Certaines approches pour atténuer le problème du cold start incluent l'utilisation de techniques de recommandation hybride, où les données externes (comme les informations démographiques des utilisateurs ou les caractéristiques des produits) sont utilisées pour combler le manque d'information.

2.3.2 Problème de Sparsité

La *sparsité* fait référence à la situation dans laquelle la matrice de notation utilisateur-produit contient un grand nombre de valeurs manquantes. Autrement dit, la majorité des utilisateurs n'ont pas évalué tous les produits disponibles. Cela rend difficile l'identification de similarités fiables entre utilisateurs ou produits, ce qui affecte la qualité des recommandations.

- **Exemple de sparsité** : Dans un grand catalogue de films, un utilisateur peut ne noter que quelques films, tandis qu'il y en a des milliers disponibles. Si la majorité

des utilisateurs évaluent seulement une petite fraction des films, la matrice de notation devient extrêmement clairsemée, rendant les algorithmes de recommandation traditionnels moins efficaces.

Solutions possibles : Pour résoudre ce problème, certaines techniques comme la *factorisation matricielle* ou l'utilisation d'approches basées sur *les voisins les plus proches* peuvent être employées pour estimer les évaluations manquantes et rendre les systèmes plus efficaces.

2.3.3 Problème de Sur-apprentissage (Overfitting)

Le *sur-apprentissage* survient lorsqu'un modèle de recommandation apprend trop bien les spécificités des données d'entraînement, au point qu'il ne généralise pas bien aux nouvelles données. Cela peut entraîner une sur-optimisation des recommandations pour les utilisateurs ou les produits déjà présents dans le jeu de données, mais une mauvaise performance sur les nouveaux utilisateurs ou produits.

— **Exemple de sur-apprentissage** : Si un système de recommandation devient trop spécifique en se basant sur des préférences utilisateurs très précises, il risque de ne pas être capable de recommander des éléments pour des utilisateurs avec des comportements différents.

Solutions possibles : Pour éviter le sur-apprentissage, il est essentiel d'utiliser des techniques de régularisation, de validation croisée, et de prendre en compte la diversité des recommandations. L'utilisation de modèles plus simples ou de méthodes hybrides peut également aider à limiter ce problème.

2.3.4 Problème du Biais et de la Diversité

Le biais dans les systèmes de recommandation survient lorsque le modèle favorise certaines préférences ou types de produits au détriment d'autres, souvent en raison d'un manque de diversité dans les données d'entraînement ou des algorithmes de recommandation. Cela peut conduire à des recommandations trop homogènes ou centrées sur des articles populaires.

— **Exemple de biais** : Si un système de recommandation se concentre uniquement sur les produits populaires ou les articles déjà évalués, il pourrait ne pas offrir des recommandations suffisamment variées, limitant ainsi l'expérience de l'utilisateur.

Solutions possibles : Pour surmonter ce problème, des techniques comme l'introduction de *diversité* dans les algorithmes ou l'utilisation de modèles de recommandation qui intègrent des critères de nouveauté peuvent être appliquées. L'injection de la diversité dans les résultats peut ainsi améliorer l'expérience utilisateur et éviter les biais.

2.3.5 Problème de Scalabilité

Les systèmes de recommandation doivent traiter de grandes quantités de données, en particulier dans des environnements de grande échelle tels que les plateformes de commerce en ligne. Le problème de scalabilité se pose lorsque le système de recommandation peine à maintenir sa performance à mesure que la taille de l'ensemble des utilisateurs et des produits augmente.

— **Exemple de scalabilité** : Un système de recommandation peut devenir lent ou inefficace à mesure que le nombre d'utilisateurs ou de produits augmente. Le temps de calcul des recommandations peut devenir prohibitif.

Solutions possibles : Les solutions à ce problème incluent l'utilisation de techniques d'optimisation, telles que la *parallelisation* des calculs ou l'utilisation de *l'informatique distribuée*, ainsi que des modèles approximatifs qui permettent de réduire la charge computationnelle.

2.3.6 Problème d'Évaluation et de Mesure

L'évaluation des systèmes de recommandation est également un défi. Les mesures traditionnelles, telles que la précision ou le rappel, ne tiennent pas toujours compte des nuances des préférences des utilisateurs, comme la diversité ou la nouveauté. De plus, la mesure de la satisfaction de l'utilisateur peut être subjective et difficile à quantifier de manière objective.

Solutions possibles : L'utilisation de métriques combinées telles que le F1-score ou de mesures spécifiques aux applications, comme le NDCG (Discounted Cumulative Gain), permet d'évaluer plus précisément la pertinence des recommandations.

Chapitre 3

Intégration de l'IA Générative dans les Systèmes de Recommandation Éducatifs

Introduction

La sélection des cours constitue un élément essentiel du parcours universitaire de tout étudiant, influençant profondément son expérience académique ainsi que ses perspectives professionnelles futures [?]. Au sein des facultés marocaines, notamment dans les grandes universités publiques telles que celles de Rabat, Casablanca ou Marrakech, où des milliers de modules sont proposés chaque année, ce processus peut devenir complexe et long, surtout pour les étudiants nouvellement inscrits.

Traditionnellement, les étudiants marocains s'orientent à l'aide des conseils des enseignants, des responsables pédagogiques, ou en se basant sur les expériences partagées par leurs camarades. Cependant, cette méthode peut générer des inégalités dans l'accès à une information fiable et complète, puisque tous les étudiants ne bénéficient pas du même accompagnement ni du même réseau de contacts informés [?].

Dans d'autres domaines, les systèmes de recommandation comme le filtrage collaboratif ont été utilisés avec succès pour offrir des suggestions personnalisées. Néanmoins, lorsqu'il s'agit de recommander des cours universitaires dans le contexte de l'enseignement supérieur marocain, ces systèmes se heurtent à plusieurs limitations, notamment l'absence de données structurées, la diversité des filières, et le manque de retour systématique sur les performances des modules.

Contexte de l'utilisation de l'intelligence artificielle générative dans l'éducation

L'intelligence artificielle générative (IAG), représentée par des modèles comme *GPT* (Generative Pre-trained Transformer), *BERT* ou encore *LLaMA*, marque une nouvelle ère dans le domaine de l'intelligence artificielle. Contrairement aux systèmes traditionnels de traitement de l'information, ces modèles sont capables de générer de manière autonome des textes cohérents, de répondre à des questions, de résumer des documents, ou encore de produire du contenu personnalisé. Ces capacités ont rapidement attiré l'attention du secteur éducatif, en quête de solutions innovantes pour améliorer l'apprentissage et l'accompagnement pédagogique.

Dans un contexte où les apprenants sont de plus en plus nombreux, diversifiés et connectés, les établissements éducatifs sont confrontés à des défis majeurs : comment personnaliser l'enseignement, soutenir les élèves en difficulté, fournir un accès équitable au savoir et répondre aux attentes d'une génération numérique ? L'IAG apparaît comme une réponse prometteuse à ces problématiques.

Par exemple, les outils basés sur l'IAG peuvent générer des exercices adaptés au niveau d'un élève, résumer un cours long en quelques phrases clés, ou encore simuler un échange interactif avec un tuteur virtuel. Plus encore, l'IA générative permet de créer des parcours d'apprentissage personnalisés, d'analyser les productions des élèves pour fournir un retour immédiat et intelligent, ou de recommander des ressources en fonction du style d'apprentissage de chacun.

Cependant, cette technologie suscite aussi des interrogations : peut-elle vraiment remplacer l'accompagnement humain ? Est-elle capable de respecter la diversité des contextes pédagogiques ? Quels sont les risques d'un usage non encadré (biais, erreurs, plagiat) ?

Dans ce contexte, l'intégration de l'intelligence artificielle générative dans les systèmes éducatifs doit être pensée avec rigueur, en tenant compte à la fois de ses potentialités et de ses limites. Ce chapitre se propose d'explorer les contributions spécifiques de l'IAG dans le cadre des systèmes de recommandation éducatifs, en mettant en lumière ses applications concrètes, ses mécanismes techniques, ainsi que les défis qu'elle soulève.

Objectifs du chapitre

Ce chapitre vise à :

- Présenter le contexte général de l'émergence de l'intelligence artificielle générative (IAG) et son évolution récente ;
- Expliquer les principes de fonctionnement des modèles génératifs comme *GPT*, *BERT*, ou *LLaMA* ;
- Identifier les opportunités offertes par l'IAG dans le domaine de l'éducation, en particulier en matière de personnalisation de l'apprentissage ;
- Illustrer les cas d'usage concrets de l'IAG dans les systèmes éducatifs, notamment dans les systèmes de recommandation pédagogique ;
- Mettre en évidence les limites, risques et défis éthiques associés à l'usage de l'IAG dans l'enseignement ;
- Fournir une base de réflexion pour une intégration responsable et efficace de l'IAG dans les environnements d'apprentissage.

Rôle des modèles de type LLM (Large Language Models)

Les modèles de langage de grande taille (*Large Language Models*, LLM) sont au cœur des avancées récentes en intelligence artificielle générative. Ces modèles, entraînés sur des corpus massifs de textes, sont capables de comprendre, générer et transformer du langage naturel avec un niveau de fluidité et de pertinence sans précédent.

Dans le contexte éducatif, les LLM jouent plusieurs rôles essentiels :

- **Génération de contenu pédagogique** : production automatique de supports de cours, d'exercices, d'explications détaillées ou de résumés adaptés ;

- **Personnalisation de l'apprentissage** : recommandation de ressources spécifiques, adaptation de la difficulté, propositions de parcours individualisés ;
- **Assistance interactive** : réponses aux questions des étudiants via des chatbots éducatifs ;
- **Analyse des productions** : évaluation automatique des rédactions, retour pédagogique immédiat.

Toutefois, l'efficacité des LLM dépend fortement de la qualité des données d'entraînement et de leur contextualisation. Leur rôle est de compléter l'enseignement, non de le remplacer.

3.1 Apports de l'IA générative aux systèmes de recommandation éducatifs

L'intelligence artificielle générative (IAG) transforme en profondeur les systèmes de recommandation éducatifs, en dépassant les approches classiques fondées uniquement sur des filtrages collaboratifs ou des modèles statistiques. Grâce à sa capacité à comprendre, générer et adapter des contenus pédagogiques en langage naturel, l'IAG permet de concevoir des systèmes plus intelligents, plus adaptatifs et véritablement centrés sur les besoins individuels des apprenants.

3.1.1 Personnalisation avancée des parcours d'apprentissage

L'un des principaux apports de l'IA générative réside dans sa capacité à analyser des données complexes liées aux apprenants : préférences personnelles, style cognitif, historique des performances, interactions avec la plateforme, etc. Ces informations permettent de construire des profils d'apprentissage détaillés et dynamiques.

Ainsi, le système peut recommander des séquences pédagogiques personnalisées. Par exemple, un étudiant en difficulté sur un chapitre donné se verra proposer des modules de renforcement ciblés, tandis qu'un autre, plus avancé, sera orienté vers des contenus plus exigeants. Cette personnalisation évolue en temps réel selon la progression et l'engagement de l'apprenant, ce qui améliore l'efficacité de l'apprentissage.

3.1.2 Génération de contenus pédagogiques adaptés

Contrairement aux approches traditionnelles, l'IAG est capable de générer automatiquement du contenu éducatif pertinent et personnalisé. Ces contenus incluent notamment :

- des exercices adaptés au niveau de l'élève ;
- des quiz d'auto-évaluation dynamiques ;
- des résumés intelligents de leçons ou de chapitres ;
- des explications alternatives ou simplifiées en fonction du style d'apprentissage.

Cette production automatisée représente un gain de temps considérable pour les enseignants, tout en offrant aux élèves une diversité de supports et une meilleure accessibilité aux connaissances. Le contenu généré est aussi actualisable en fonction du niveau et du rythme de chaque apprenant.

3.1.3 Réponses contextuelles et interactives aux besoins des apprenants

Les modèles de langage avancés (comme GPT) permettent aux systèmes éducatifs d'interagir avec les utilisateurs en langage naturel. Ils peuvent :

- répondre à des questions de manière contextuelle et instantanée ;
- fournir des éclaircissements supplémentaires sur une notion ;
- guider l'élève dans la résolution d'un problème ou d'un exercice, étape par étape.

Ces interactions enrichissent considérablement l'expérience d'apprentissage en la rendant plus fluide, plus engageante et mieux adaptée aux besoins ponctuels des apprenants. Le système joue ainsi un rôle d'accompagnateur pédagogique intelligent, disponible à tout moment.

3.2 Systèmes à base de connaissances

Pour surmonter les limitations du démarrage à froid, certains chercheurs ont développé des systèmes de recommandation à base de connaissances qui intègrent une expertise du domaine et des règles académiques. Par exemple, CrsRecs (Ng et Linn, 2017) utilise l'analyse de sujets, l'analyse des sentiments et des données d'enquêtes sur les priorités des étudiants pour classer les cours potentiels. Ces systèmes peuvent formuler des recommandations même sans disposer de nombreuses données historiques, mais nécessitent un effort considérable pour encoder les connaissances du domaine et les règles.

3.3 Approches hybrides

Des travaux récents se sont concentrés sur des approches hybrides combinant plusieurs techniques de recommandation. Pathways (Chen et al., 2022) visualise les tendances d'inscription aux cours dans le passé tout en prenant en compte les intérêts des étudiants et les exigences académiques. Le système aide les étudiants à explorer les différentes options de cours via une interface interactive, en équilibrant la personnalisation et la découverte fortuite. De même, Salehudin et al. (2019) proposent un modèle de filtrage collaboratif qui s'intègre aux tendances historiques d'inscription ainsi qu'aux données de performance des étudiants afin de fournir des recommandations personnalisées.

3.4 Exigences en matière de données

La plupart des approches de recommandation avancées nécessitent une collecte de données étendue sur plusieurs semestres ou années académiques pour construire des modèles efficaces (Salehudin et al., 2019). Cette exigence représente un défi pour les établissements souhaitant mettre en place de tels systèmes, car ils doivent consacrer beaucoup de temps à identifier les sources de données et à organiser l'accès aux informations réparties sur plusieurs serveurs, souvent gérés par différentes unités, avant que le système ne devienne réellement utile.

Les universités sont des environnements où les données s'échappent facilement, et les acteurs privés peuvent bénéficier d'un avantage s'ils parviennent à agréger des données fournies par les étudiants eux-mêmes, issues de plusieurs établissements. Les dirigeants universitaires doivent se préparer en interne avant que des solutions externes de recommandation de cours ne fassent leur apparition sur leur campus. Le travail présenté ici

réflète une philosophie de prise en main et de développement innovant de l'environnement informationnel et décisionnel de l'institution.

3.5 LLM pour les systèmes de recommandation

Les avancées récentes dans les modèles de langage de grande taille (LLM) ont ouvert de nouvelles perspectives pour les systèmes de recommandation. Plutôt que de considérer la recommandation comme un simple problème d'appariement ou de classement, les LLM permettent une approche plus souple en transformant les tâches de recommandation en problèmes de compréhension et de génération du langage (Geng et al., 2023). Cela permet aux systèmes de recommandation d'exploiter la compréhension sémantique riche et les capacités de génération des modèles linguistiques, tout en fournissant un cadre uniifié pour diverses tâches de recommandation.

Les LLM peuvent remplir plusieurs rôles dans les systèmes de recommandation (Xu et al., 2024). Ils peuvent effectuer de l'ingénierie de caractéristiques en générant des descripteurs textuels riches, extraire des représentations sémantiques (*embeddings*) pour les éléments ou les utilisateurs afin de résoudre les scénarios de démarrage à froid, agir comme systèmes de recommandation directs, ou encore servir de contrôleurs pour permettre des recommandations plus interactives et explicables. Ces capacités peuvent être exploitées via des approches « zero-shot » ou « few-shot » à l'aide d'invites soigneusement conçues, ou encore par un ajustement (*fine-tuning*) sur des données de recommandation spécifiques au domaine.

L'un des avantages majeurs des systèmes de recommandation basés sur des modèles linguistiques est leur capacité à unifier divers types d'informations et de tâches dans un seul et même cadre. Comme le montre le système M6-Rec (Cui et al., 2022), un modèle fondamental unique peut prendre en charge plusieurs tâches de recommandation en les transformant en problèmes de compréhension et de génération du langage. Cela inclut des tâches classiques comme la recherche et le classement, ainsi que des tâches plus complexes comme la génération d'explications ou les recommandations conversationnelles. L'approche basée sur des représentations textuelles permet au système de traiter les données comportementales des utilisateurs, les métadonnées des éléments et les interactions utilisateur-élément dans un format cohérent, tout en contournant les limites des systèmes basés sur les identifiants lorsqu'il s'agit d'éléments jamais vus auparavant.

3.6 Processus de Recommandation de Cours

Le système de recommandation de cours utilise une approche hybride combinant la recherche de similarité basée sur des *embeddings* et le raisonnement d'un grand modèle de langage (LLM) afin de fournir des suggestions de cours personnalisées. Les utilisateurs interagissent avec le système de recommandation en utilisant un langage naturel pour décrire leur parcours, leurs centres d'intérêt et leurs objectifs. Le système fonctionne sur une base de données contenant des descriptions de cours et permet aux utilisateurs de restreindre l'espace de recherche grâce à un filtrage simple basé sur les niveaux des cours, ce qui permet d'obtenir des recommandations adaptées aux différents niveaux académiques.

Le processus de recommandation comprend cinq opérations principales illustrées dans la Figure 1 et organisées en deux étapes :

1. **Génération de Contexte** : Création d'un contexte de recherche pertinent à partir de la requête de l'utilisateur
2. **Recommandation** : Identification et classement des cours les plus adaptés

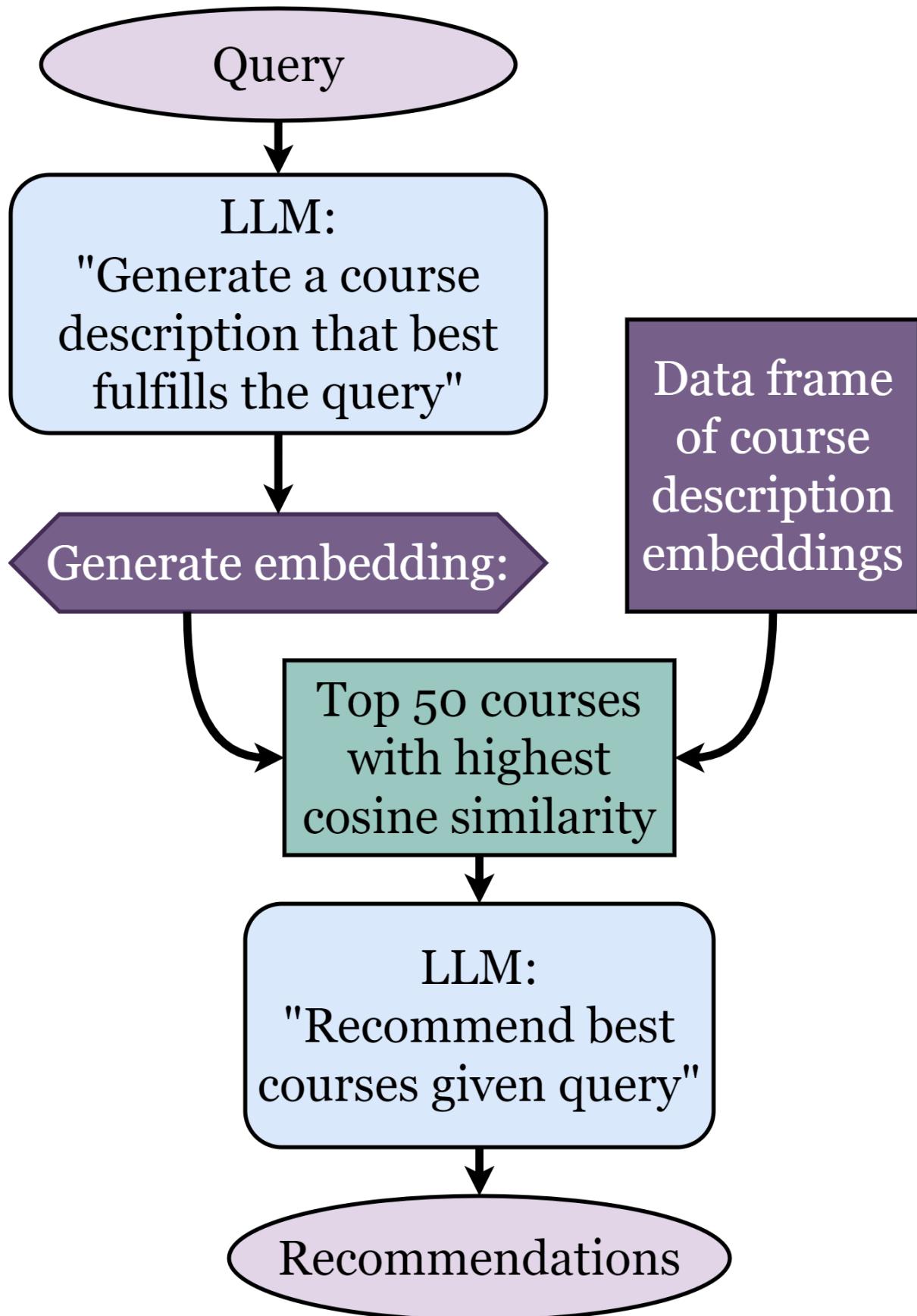


FIGURE 3.1 – Pipeline hybride LLM-Embedding pour les recommandations de cours

3.7 Données

Le système repose sur un **fichier CSV** contenant les informations descriptives des cours. Ces données sont chargées dans un **dataframe Pandas** pour traitement. Chaque enre-

gistrement représente un cours et contient les champs suivants :

- **Objectifs académiques visés** (Academic Goals)
- **Filière ou spécialisation concernée** (Major)
- **Loisirs liés au contenu du cours** (Hobbies)
- **Compétences informatiques requises ou développées** (Computer Skills)
- **Intérêt pour les langues** (Interest in Languages)
- **Moyenne générale minimale recommandée** (GPA)

Pour chaque cours, un vecteur d'*embedding* est pré-calculé à partir des champs textuels, en utilisant le modèle `mxbai-embed-large`. Ce modèle projette les données dans un espace vectoriel de grande dimension, ce qui permet de mesurer la similarité sémantique entre les cours et de les comparer aux profils des utilisateurs.

Les embeddings sont stockés dans une base vectorielle **ChromaDB**, afin de permettre une recherche rapide par similarité lors de la phase de recommandation.

3.8 Génération du Contexte

Les systèmes RAG traditionnels reposent sur la similarité sémantique directe entre les embeddings (représentations vectorielles) des requêtes et des documents pour effectuer la recherche, ce qui peut s'avérer sous-optimal lorsqu'il existe un écart lexical et sémantique important entre la manière dont les utilisateurs expriment leurs besoins et la façon dont les documents sont rédigés. Ce problème est particulièrement marqué dans les contextes éducatifs, où les requêtes des étudiants (par exemple : « Je veux apprendre comment les ordinateurs pensent ») manquent souvent du vocabulaire technique et de la structure formelle des descriptions de cours (par exemple : « Introduction aux architectures neuronales », « Méthodes mathématiques des systèmes d'apprentissage profond »). Même les systèmes de recherche dense modernes comme DPR (Dense Passage Retrieval, Karpukhin et al., 2020), entraînés pour associer les requêtes aux documents pertinents, peuvent rencontrer des difficultés lorsque les textes de requête et les documents cibles présentent des caractéristiques linguistiques aussi différentes.

Naïve RAG

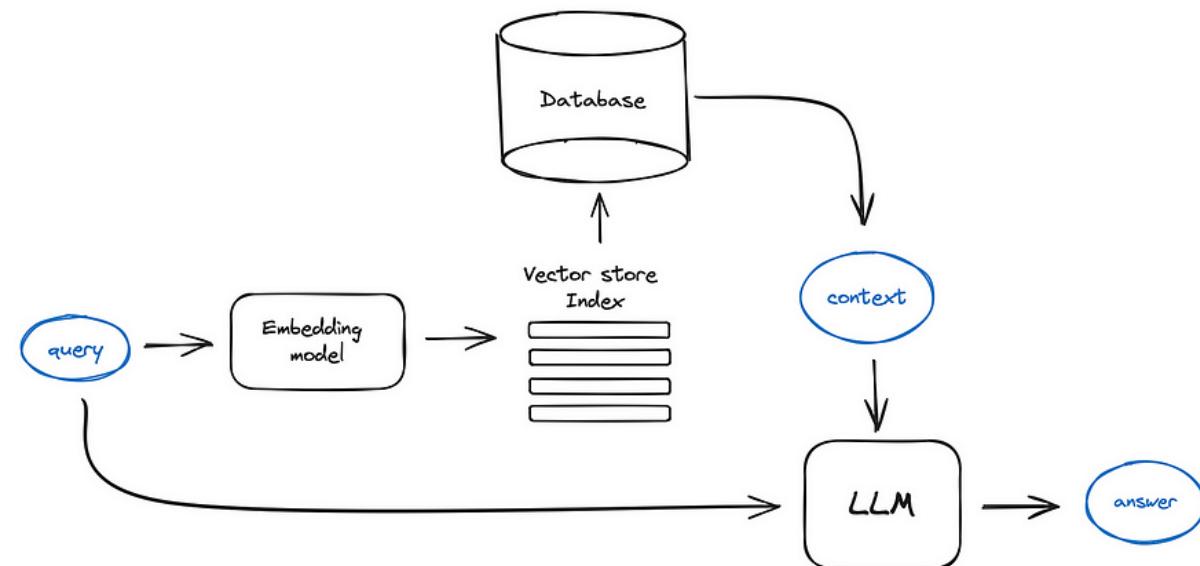


FIGURE 3.2 – Phases de la Génération du Contexte

Pour combler cet écart sémantique, nous utilisons un processus de recherche en deux étapes qui exploite la capacité des modèles de langage de grande taille (LLM) à faire le lien entre ces différents domaines. Nous commençons par solliciter GPT-3.5-turbo (Brown et al., 2020) afin de générer une description de cours « idéale », qui correspondrait parfaitement aux intérêts et au niveau de l'étudiant. Cette description idéalisée sert ensuite de vecteur de requête plus efficace pour notre système de recherche basé sur les embeddings, car elle traduit l'expression en langage naturel des intérêts de l'étudiant en un langage académique structuré, typique des descriptions de cours.

En alignant les caractéristiques linguistiques de la requête avec celles des documents cibles avant de calculer les embeddings, nous veillons à ce que la recherche vectorielle capte une pertinence sémantique réelle, au lieu d'être perturbée par des différences de style ou de structure linguistique. Nous utilisons GPT-3.5-turbo en raison de sa rapidité, car les capacités plus avancées des modèles plus grands ne sont pas nécessaires pour cette tâche de génération.

Concrètement, le processus de recherche adopte une approche en deux étapes, décrite dans les Algorithmes 1 et 2, pour identifier les 50 cours les plus pertinents sur le plan sémantique. Ces cours sélectionnés sont ensuite regroupés dans une chaîne de contexte structurée, composée des identifiants des cours et de leurs descriptions respectives. Cette chaîne de contexte constitue la base de l'étape de recommandation.

3.9 Recommandation

Après avoir généré un contexte filtré des cours les plus pertinents sur le plan sémantique, le système utilise le modèle `ollama/llama3.2` pour formuler la recommandation finale. Nous tirons parti des capacités de raisonnement avancées de LLaMA 3.2 pour évaluer plus efficacement l'interaction complexe entre les intérêts des étudiants, le contenu des cours et les trajectoires éducatives.

L'ingénierie des invites (*prompt engineering*) pour la génération des recommandations repose sur trois objectifs principaux :

- maintenir la qualité des recommandations,
- assurer la fiabilité du système,
- fournir des informations exploitables.

Afin de garantir une cohérence maximale entre les réponses, nous fixons le paramètre `temperature` à 0, bien que des variations notables puissent subsister dans les cas de requêtes ouvertes — un comportement que nous discutons dans la section des résultats. Le système retourne un ensemble de dix recommandations de cours, présentées en format Markdown pour une lisibilité optimale. Chaque recommandation comprend :

- l'identifiant du cours,
- une justification courte et ciblée expliquant la pertinence du cours par rapport au profil de l'étudiant,
- une évaluation du niveau de confiance (*élévé, moyen ou faible*).

Ce format structuré permet de concilier exhaustivité et clarté — dix options offrent une diversité suffisante tout en restant faciles à analyser. Les justifications apportent un contexte décisionnel concret pour aider l'étudiant à faire un choix éclairé. Des liens vers des informations supplémentaires sur chaque cours seront ajoutés avant le déploiement officiel.

Pour garantir l'intégrité des recommandations et prévenir toute forme de désinformation, plusieurs contraintes sont imposées au système :

- Il ne fournit pas de conseils académiques généraux,

- Il ne discute pas des prérequis absents des descriptions de cours,
- Il ne recommande pas de cours hors contexte.

Ces règles assurent que le système reste strictement un moteur de recommandation de cours basé sur le contexte fourni, sans se substituer à un conseiller académique humain.

Chapitre 4

Conception Détailée

Introduction

Dans ce chapitre, nous allons entreprendre la conception des différents diagrammes et flux qui constituent cette application, fournissant ainsi une représentation claire et détaillée des interactions, des processus et des composants du système.

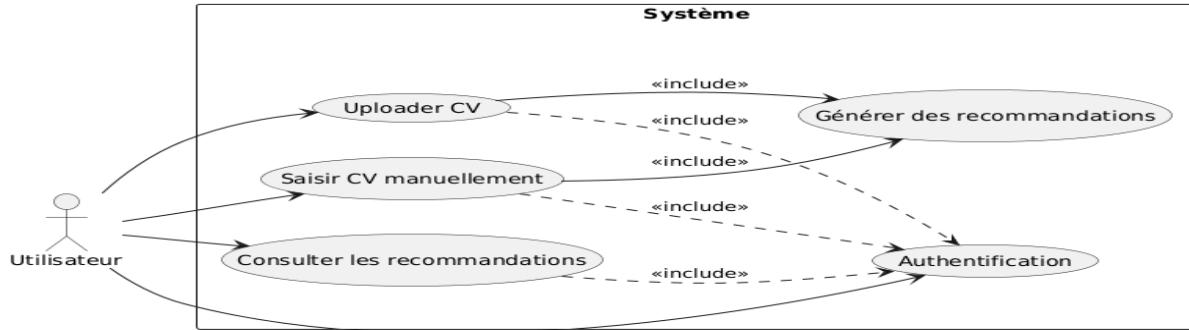


FIGURE 4.1 – Diagramme de cas d'utilisation du système

4.1 Analyse et Conception

4.1.1 Les Acteurs du Système

Apprenant (Utilisateur / Étudiant)

L'apprenant est l'acteur principal qui utilise la plateforme pour consulter, suivre et évaluer les cours.

Rôles :

- S'inscrire et créer un profil
- Choisir ses centres d'intérêt
- Recevoir des recommandations personnalisées

Système de Recommandation

Moteur intelligent qui analyse les données de l'utilisateur (centres d'intérêt, historique, niveau, etc.) pour générer des suggestions personnalisées de cours.

4.2 Diagramme de Cas d'utilisation

Le diagramme de cas d'utilisation décrit le comportement du système du point de vue utilisateur sous forme d'actions et de réactions.

Il existe deux concepts fondamentaux dans la modélisation par les cas d'utilisation :

- **Les acteurs** qui agissent sur le système.
- **Les cas d'utilisations** qui représentent les façons dont le système est manipulé par les acteurs.

Chaque cas d'utilisation indique une fonctionnalité du système déclenchée par un acteur externe au système.

Ce genre de diagramme permet de mettre en place et de comprendre les besoins des utilisateurs.

4.3 Diagrammes de Séquence

Parmi les diagrammes intéressants d'UML, on trouve le **diagramme de séquence** qui illustre une représentation graphique des interactions entre l'acteur et le système selon un ordre chronologique dans la formulation Unified Modeling Language.

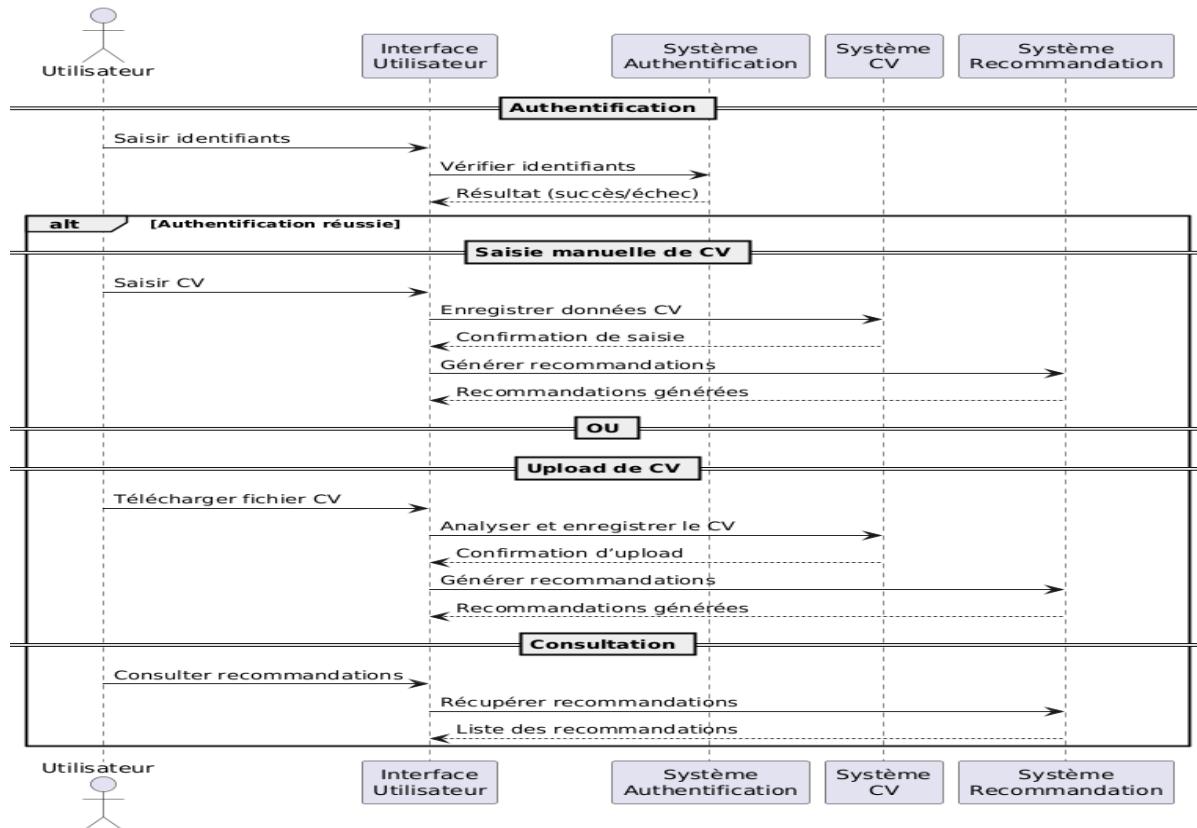


FIGURE 4.2 – Diagramme de séquence du système

L'utilité du diagramme de séquence est de montrer les interactions d'objet dans le cadre d'un scénario de cas d'utilisation. La dimension verticale du diagramme représente le **temps**, permettant de visualiser l'enchaînement des actions dans le temps, et de spécifier la naissance et la mort d'objets.

Les périodes d'activité des objets sont symbolisées par des **rectangles**, et ces objets dialoguent à l'aide de **messages**.

Donc, vu le service offert par les diagrammes de séquence, cette partie est consacrée à citer ceux des cas d'utilisation les plus importants dans l'application, ce qui permet de mieux voir et tracer l'enchaînement du projet.

4.4 Diagrammes d'Activité

Le **diagramme d'activité** est l'un des diagrammes comportementaux d'UML les plus utilisés. Il permet de modéliser les différents flux de contrôle ou de données au sein d'un processus métier ou d'un scénario spécifique du système.

Il illustre la séquence des activités ou des actions réalisées par les acteurs ou les composants du système. Chaque activité représente une étape ou une opération, et les flèches indiquent la transition d'une activité à une autre.

Ce diagramme met en évidence les **conditions**, les **boucles**, les **décisions**, et permet aussi de représenter des traitements parallèles. Il est particulièrement utile pour analyser les processus métiers, décrire les algorithmes et documenter les comportements dynamiques d'un système.

Ainsi, dans cette partie, nous allons présenter les diagrammes d'activité associés aux cas d'utilisation les plus significatifs de l'application, afin de mieux visualiser les enchaînements logiques et opérationnels des différentes fonctionnalités du système.

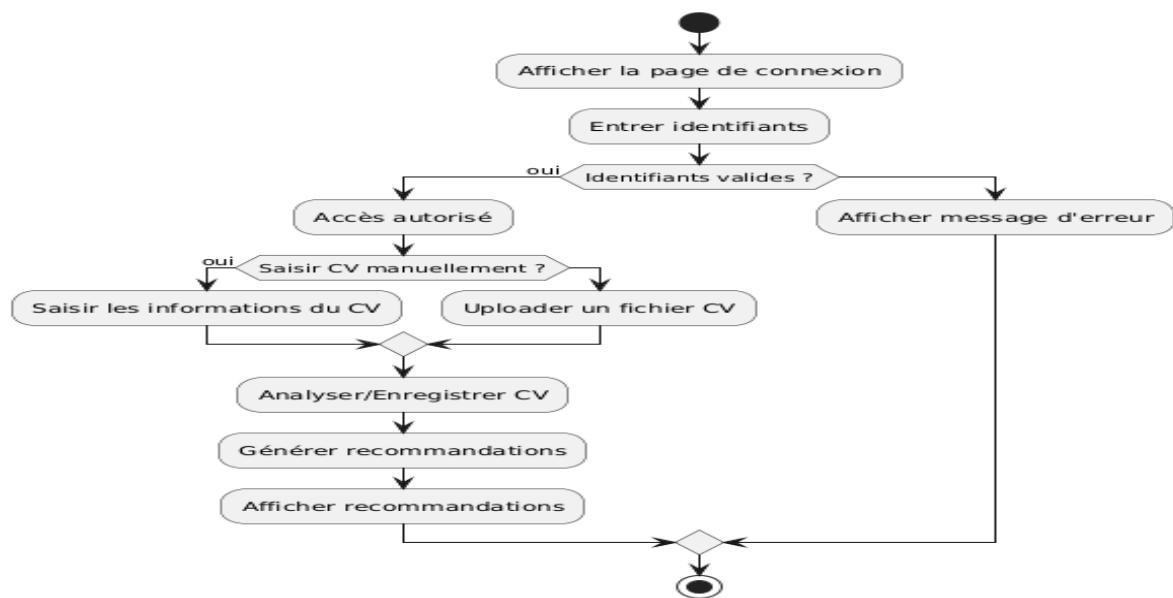


FIGURE 4.3 – Diagramme d’activité représentant le déroulement des actions du système

Chapitre 5

Réalisation

Introduction

Dans ce chapitre, nous allons survoler les différents choix techniques pour lesquels nous avons opté, et présenter l'application dans sa globalité.

5.1 Méthodologie de développement

5.1.1 Langage UML

De nos jours, pour pouvoir spécifier et exprimer les besoins et les exigences des acteurs, du système et de l'architecture globale, plusieurs outils de modélisation de processus métier se sont présentés.



FIGURE 5.1 – Logo UML

La modélisation **UML** permet de vulgariser les aspects liés à la conception et à l'architecture, propres au logiciel, au client. Aussi, elle apporte une compréhension rapide du programme à d'autres développeurs externes en cas de reprise du logiciel et facilite sa maintenance.

L'utilisation d'UML varie en fonction du type d'application à réaliser, et de sa taille. Par exemple, sur les douze diagrammes que contient UML, deux ou trois suffisent à réaliser un petit projet. Il faut savoir qu'un modèle doit avoir un objectif précis. Sinon, il risque de ne pas être adapté à son usage.

Exemples de diagrammes :

- Le diagramme de cas d'utilisation,
- Les diagrammes de séquence,
- Le diagramme d'activité.

Justification du choix d'UML

UML est avant tout un support de communication performant, qui facilite la représentation et la compréhension de solutions orientées objet.

L'aspect formel de sa notation limite les ambiguïtés et les incompréhensions. Son indépendance par rapport aux langages de programmation, aux domaines d'application et aux processus, en fait un langage universel.

UML, contrairement à son prédecesseur **MERISE** qui est une méthode systémique (orientée données), donne un sens intéressant à l'approche objet et couvre de plus tout le cycle de réalisation du logiciel.

5.1.2 CrewAI

CrewAI est une plateforme qui permet aux développeurs de construire et de déployer des flux de travail automatisés en utilisant plusieurs agents d'intelligence artificielle qui collaborent pour effectuer des tâches complexes.

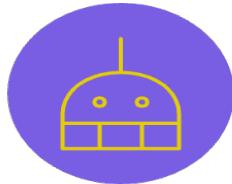
Les agents d'IA sont des assistants capables d'effectuer des tâches et d'interagir avec le monde. Contrairement aux systèmes traditionnels qui suivent des règles fixes, ils peuvent apprendre et s'adapter à de nouvelles situations. Considérez-les comme des robots intelligents qui vous aident à accomplir toute une série de tâches.

Mais qu'est-ce qui différencie les agents d'intelligence artificielle des autres entités d'intelligence artificielle, comme les populaires modèles de langage dont nous avons tous entendu parler ?

Dans cet article, je vais répondre à cette question et vous présenter CrewAI, un framework Python gratuit et open-source conçu pour simplifier le développement de systèmes d'IA multi-agents. Nous étudierons la distinction entre les agents et les modèles de langage, nous discuterons des raisons pour lesquelles les cadres d'agents sont importants pour la construction d'applications d'IA. construire des applications d'IA et nous montrerons comment CrewAI permet aux agents de collaborer et d'obtenir d'excellents résultats.

5.1.2.1 Agents d'IA vs. LLMs

Démystifions un malentendu courant concernant la différence entre les agents et les grands modèles linguistiques (LLM). Tous deux appartiennent à la famille de l'intelligence artificielle, mais ils possèdent des capacités distinctes.



Agent

Performs specific tasks and makes decisions based on its environment.



Large Language Model

Focuses on understanding and generating human-like text.

FIGURE 5.2 – Agents d'IA vs. LLMs

Les modèles linguistiques, tels que ChatGPT et Geminisont très habiles dans l'utilisation du langage. Ils ont suivi une formation approfondie sur de grandes quantités de textes et de codes, ce qui les a dotés de la capacité de comprendre et de produire un langage qui ressemble beaucoup à la communication humaine.

Les LLM sont d'habiles faiseurs de mots de l'intelligence artificielle, produisant un large éventail de contenus, y compris des traductions, des résumés, des récits créatifs ou même de la poésie. Leur champ d'application est généralement limité aux tâches liées à la langue .

Les agents, quant à eux, se concentrent principalement sur l'action. Ils sont capables de naviguer, d'interagir avec des objets et de prendre des décisions sur la base de leurs perceptions.

En bref, les modèles linguistiques sont des cerveaux et les agents sont des mains. Ensemble, ils forment un duo puissant.

Les agents jouant un rôle essentiel dans les applications d'IA, comment gérer leur complexité lorsque plusieurs agents doivent travailler ensemble ? C'est là qu'interviennent les cadres d'agents.

5.1.2.2 La nécessité d'un cadre pour les agents

Le besoin de cadres d'agents découle de la complexité croissante des applications de l'IA, en particulier celles qui impliquent plusieurs agents travaillant en collaboration pour atteindre un objectif commun. Voyons pourquoi les cadres d'agents sont essentiels.

5.1.2.3 Orchestration et coordination

À mesure que les systèmes d'IA prennent de l'ampleur, ils intègrent souvent de nombreux agents dotés de capacités diverses. Il devient de plus en plus difficile de gérer ces interactions et de s'assurer qu'elles fonctionnent harmonieusement.

Les cadres d'agents offrent un environnement structuré qui permet d'orchestrer les activités des agents, de définir leurs rôles et responsabilités et d'améliorer la communication.

Dans les systèmes multi-agents, un aspect important est l'attribution efficace des tâches aux agents les plus appropriés et la gestion efficace des ressources partagées. Les cadres d'agents fournissent des mécanismes dynamiques pour la répartition des tâches, la négociation des ressources et la résolution des conflits.

5.1.2.4 Qu'est-ce que CrewAI ?

CrewAI est un framework Python open-source conçu pour soutenir le développement et la gestion de systèmes d'IA multi-agents.

CrewAI améliore ces systèmes d'IA en attribuant des rôles spécifiques, en permettant une prise de décision autonome et en facilitant la communication entre les agents. Cette approche leur permet de s'attaquer à des problèmes complexes plus efficacement que des agents travaillant seuls.



FIGURE 5.3 – Logo de CrewAI

Ce cadre se compose d'une série d'outils, notamment des moteurs de recherche sur le web et des modèles linguistiques, qui permettent aux agents d'entrer en contact avec le monde extérieur, de collecter des informations et d'agir pour atteindre leurs objectifs.

La conception et l'évolutivité de CrewAI en font un outil idéal pour le développement d'applications multi-agents simples et complexes, encourageant une méthode collaborative pour relever les défis et prendre des décisions au sein des systèmes d'IA.

Examinons quelques-unes des principales caractéristiques qui font de CrewAI un outil puissant pour la construction de systèmes multi-agents.

Orchestration d'agents : CrewAI veille à ce que chaque agent connaisse son rôle dans la performance. Il fournit les outils nécessaires pour définir et coordonner les comportements des agents, afin de s'assurer que tout le monde joue en synchronisation.

Architecture basée sur les rôles : Comme pour l'attribution de différents instruments aux musiciens, CrewAI vous permet d'attribuer des rôles spécifiques aux agents, en définissant leurs capacités et leurs autorisations. Cela permet de créer un système modulaire et bien structuré, même lorsque les choses deviennent complexes.

Une communication souple : CrewAI prend en charge différents canaux de communication, ce qui permet aux agents d'échanger des informations en toute transparence. Pensez-y comme si vous disposiez d'un chat privé, d'une discussion de groupe et d'un mégaphone en un seul et même endroit.

Intégration des outils : CrewAI permet aux agents d'interagir avec le monde grâce à différents outils. Ces outils permettent aux agents d'effectuer des recherches sur le web, de comprendre le langage, d'analyser des données et d'effectuer des tâches personnalisées.

Évolutivité : CrewAI est conçu pour s'adapter sans effort, garantissant que votre système multi-agents reste réactif et efficace au fur et à mesure de sa croissance.

Mais quels sont les avantages que CrewAI apporte au tableau ? Voyons quels sont ses avantages.

Avantages de l'utilisation de CrewAI

Crew.ai permet à plusieurs agents d'intelligence artificielle de collaborer, de partager leurs connaissances et de coordonner leurs actions en vue d'atteindre un objectif commun.

En automatisant la distribution des tâches et la gestion des ressources, Crew.ai permet aux agents de se concentrer sur leurs rôles spécifiques avec un minimum de frais généraux.

Le cadre prend également en charge l'adaptabilité, ce qui permet aux agents d'ajuster leur comportement en fonction de l'évolution des conditions ou des objectifs.

En outre, Crew.ai simplifie le processus de développement grâce à une plateforme conviviale pour la création et la gestion de systèmes multi-agents.

Un autre point fort de CrewAI est son intégration avec une large gamme d'outils. Cela étend les capacités des agents, leur permettant d'interagir avec le monde extérieur et de recueillir des informations.

CrewAI prend en charge des outils tels que des moteurs de recherche web, des modèles de langage, des outils d'analyse de données et même des fonctionnalités personnalisées. Cela permet aux agents d'effectuer des tâches qui dépassent leurs capacités de base, telles que la recherche d'informations sur le web ou l'analyse de données complexes.

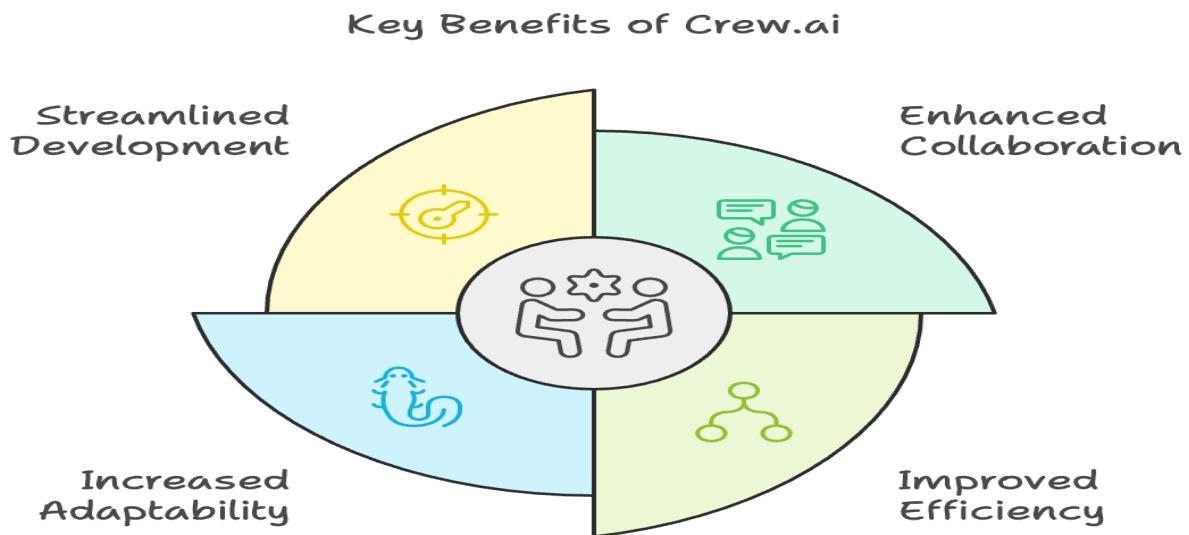


FIGURE 5.4 – Principaux avantages offerts par CrewAI

5.1.3 Ollama

Ollama est un outil open-source qui exécute de grands modèles de langage (LLM) directement sur une machine locale . Il est donc particulièrement intéressant pour les développeurs d'IA, les chercheurs et les entreprises soucieuses du contrôle des données et de la protection de la vie privée.

En exécutant les modèles localement, vous conservez l'entièvre propriété des données et évitez les risques de sécurité potentiels associés au stockage dans le cloud. Les outils d'IA hors ligne comme Ollama contribuent également à réduire la latence et la dépendance à l'égard des serveurs externes, ce qui les rend plus rapides et plus fiables.

Cet article explore les principales caractéristiques d'Ollama, les modèles pris en charge et les cas d'utilisation pratiques. À la fin, vous serez en mesure de déterminer si cet outil LLM convient à vos projets et à vos besoins en matière d'IA.

Comment fonctionne Ollama

Ollama crée un environnement isolé pour exécuter les LLM localement sur votre système, ce qui évite tout conflit potentiel avec d'autres logiciels installés. Cet environnement comprend déjà tous les composants nécessaires au déploiement de modèles d'IA, tels que :

- **Poids du modèle.** Les données pré-entraînées que le modèle utilise pour fonctionner.
- **Fichiers de configuration.** Paramètres qui définissent le comportement du modèle.
- **Dépendances nécessaires.** Bibliothèques et outils qui soutiennent l'exécution du modèle.

Pour simplifier, vous tirez d'abord des modèles de la bibliothèque d'Ollama. Ensuite, vous exécutez ces modèles tels quels ou vous ajustez les paramètres pour les adapter à des tâches spécifiques. Après la configuration, vous pouvez interagir avec les modèles en entrant des invites, et ils génèrent les réponses.

Cet outil d'IA avancé fonctionne mieux sur les systèmes à unité de traitement graphique (GPU) dédiée. Bien que vous puissiez l'exécuter sur des GPU intégrés au CPU, l'utilisation de GPU compatibles dédiés, comme ceux de NVIDIA ou d'AMD, réduira les temps de traitement et garantira des interactions plus fluides avec l'IA.

Nous recommandons de consulter la page GitHub officielle d'Ollama pour vérifier la compatibilité avec les GPU.

Principales caractéristiques d'Ollama

Ollama offre plusieurs fonctionnalités clés qui facilitent la gestion des modèles hors ligne et améliorent les performances.

Gestion locale des modèles IA

Ollama vous permet de télécharger, de mettre à jour et de supprimer facilement des modèles sur votre système. Cette fonctionnalité est précieuse pour les développeurs et les chercheurs qui accordent une grande importance à la sécurité des données.

Outre la gestion de base, Ollama vous permet de suivre et de contrôler les différentes versions du modèle. Cette fonction est essentielle dans les environnements de recherche et de production, où il peut être nécessaire de revenir à plusieurs versions de modèles ou de les tester pour déterminer celle qui produit les résultats souhaités.

Options de la ligne de commande et de l'interface graphique

Ollama fonctionne principalement via une interface de ligne de commande (CLI), ce qui vous donne un contrôle précis sur les modèles. L'interface de ligne de commande permet des commandes rapides pour extraire, exécuter et gérer les modèles, ce qui est idéal si vous êtes à l'aise pour travailler dans une fenêtre de terminal.

Si vous êtes intéressé par une approche en ligne de commande, n'hésitez pas à consulter notre tutoriel Ollama CLI.

Ollama prend également en charge des outils d'interface utilisateur graphique (GUI) tiers, tels que Open WebUI, pour ceux qui préfèrent une approche plus visuelle.

5.2 Quoi de neuf avec LLaMA 3.2 ?

- LLaMA 3.2 est une mise à jour importante qui améliore la version 3.
- Optimisations dans l'architecture et les algorithmes d'entraînement permettant :
 - Une meilleure compréhension contextuelle.
 - Une génération de texte plus naturelle et précise.
 - Une meilleure gestion des longues séquences textuelles.
- Amélioration des performances sur des benchmarks standards de traitement automatique du langage naturel (NLP).
- Support amélioré pour plusieurs langues avec une meilleure compréhension multilingue.

5.2.1 Architecture technique

- Basé sur l'architecture **Transformer**, norme dans les modèles de langage modernes.

- Plusieurs tailles de modèles disponibles, allant de quelques milliards à plusieurs dizaines de milliards de paramètres.
- Techniques avancées d'entraînement :
 - Apprentissage auto-supervisé sur de vastes corpus textuels.
 - Algorithmes d'optimisation modernes accélérant la convergence.
- Utilisation de régularisation pour éviter le surapprentissage et améliorer la généralisation.

5.2.2 Performances et benchmarks

- LLaMA 3.2 se positionne parmi les meilleurs modèles open source ou semi-open source.
- Rapprochement des performances avec des modèles commerciaux comme GPT-4 ou PaLM sur plusieurs tâches :
 - Réponse aux questions.
 - Traduction.
 - Résumé automatique.
 - Raisonnement logique.
- Bon équilibre entre taille du modèle, coût de calcul et performance.

5.2.3 Points forts de LLaMA 3.2

- **Efficacité** : Moins gourmand en ressources matérielles (GPU/TPU) comparé à certains concurrents.
- **Modularité** : Plusieurs tailles disponibles, permettant un déploiement flexible (serveurs, cloud, local).
- **Accessibilité** : Partiellement open source ou accessible sous licence, favorisant la recherche.
- **Robustesse linguistique** : Meilleure compréhension des nuances, contextes longs et langues multiples.
- **Adaptabilité** : Facilité de fine-tuning pour des domaines spécifiques (santé, finance, droit...).

5.2.4 Cas d'usage concrets

- Assistants virtuels intelligents (service client, support technique).
- Création de contenu (rédaction, génération créative).
- Traduction multilingue pour plateformes internationales.
- Analyse de sentiments et modération de contenu.
- Synthèse et résumé automatique de documents.
- Recherche et extraction d'informations dans des bases textuelles.

5.2.5 Enjeux et limites

- **Éthique et biais** : Le modèle peut refléter des biais présents dans les données d'entraînement.
- **Sécurité** : Risques d'usage malveillant (désinformation, deepfakes textuels).

- **Complexité du déploiement** : Nécessite des ressources matérielles importantes pour les grands modèles.
- **Dépendance aux données** : La qualité des données impacte directement la performance.

5.2.6 Perspectives futures

- Améliorations continues via :
 - Meilleurs algorithmes d'apprentissage.
 - Données d'entraînement plus diverses et de meilleure qualité.
 - Intégration avec d'autres technologies d'IA (vision, audio).
- Déploiements élargis dans des applications industrielles et grand public.
- Outils facilitant la personnalisation et l'interaction utilisateur.

5.2.7 Streamlit

Streamlit est un framework gratuit et open-source qui permet de créer et de partager rapidement de superbes applications web d'apprentissage automatique et de science des données.

Il s'agit d'une bibliothèque basée sur Python spécifiquement conçue pour les ingénieurs en apprentissage automatique. Les scientifiques des données ou les ingénieurs en apprentissage automatique ne sont pas des développeurs web et ils n'ont pas envie de passer des semaines à apprendre à utiliser ces frameworks pour créer des applications web. Ils souhaitent plutôt un outil plus facile à apprendre et à utiliser, pour autant qu'il puisse afficher des données et collecter les paramètres nécessaires à la modélisation.

Streamlit vous permet de créer une application à l'apparence étonnante avec seulement quelques lignes de code.



FIGURE 5.5 – Logo de Streamlit

L'avantage de Streamlit est que vous n'avez même pas besoin de connaître les bases du développement web pour commencer ou pour créer votre première application web. Si vous êtes un adepte de la science des données et que vous souhaitez déployer vos modèles facilement, rapidement et avec seulement quelques lignes de code, Streamlit est un bon choix.

L'un des aspects importants de la réussite d'une application consiste à la doter d'une interface utilisateur efficace et intuitive. La plupart des applications modernes, gourmandes en données, sont confrontées au défi de créer rapidement une interface utilisateur efficace, sans passer par des étapes compliquées. Streamlit est une bibliothèque Python open-source prometteuse, qui permet aux développeurs de créer des interfaces utilisateur attrayantes en un rien de temps.

Streamlit est le moyen le plus simple, en particulier pour les personnes qui n'ont pas de connaissances en matière d'interface, d'intégrer leur code dans une application web :

- Aucune expérience ou connaissance en matière d'interface (html, js, css) n'est requise.
- Vous n'avez pas besoin de passer des jours ou des mois à créer une application web, vous pouvez créer une très belle application d'apprentissage automatique ou de science des données en seulement quelques heures ou même quelques minutes.
- Il est compatible avec la majorité des bibliothèques Python (par exemple pandas, matplotlib, seaborn, plotly, Keras, PyTorch, SymPy (latex)).
- Moins de code est nécessaire pour créer des applications web étonnantes.
- La mise en cache des données simplifie et accélère les pipelines de calcul.

5.3 Base de Données : MySQL

MySQL est un système de gestion de base de données relationnelle open source. Il est utilisé pour stocker et gérer les données collectées par les capteurs, les informations des utilisateurs et les historiques des données. MySQL est connu pour sa performance, sa fiabilité et sa facilité d'utilisation.

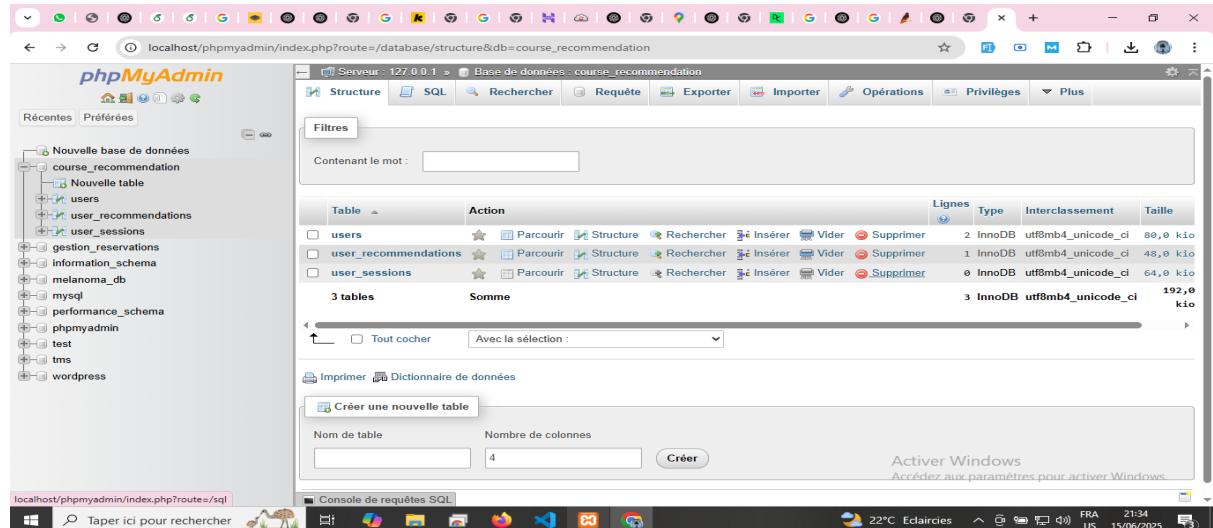


FIGURE 5.6 – Interface ou composant lié à MySQL

5.4 Procédure de Déploiement de l'Application Streamlit

Cette section décrit les étapes nécessaires pour déployer l'application développée avec le framework **Streamlit**.

5.4.1 1. Pré-requis

Avant de procéder au déploiement, assurez-vous que les éléments suivants sont installés sur la machine :

- Python 3.8 ou version supérieure
- pip (gestionnaire de paquets Python)
- Git (facultatif si vous clonez un dépôt)
- Un environnement virtuel (recommandé)

5.4.2 2. Crédation et activation de l'environnement virtuel

```
python -m venv env
source env/bin/activate          % Sous Linux ou macOS
env\Scripts\activate             % Sous Windows
```

5.4.3 3. Installation des dépendances

Installez les bibliothèques nécessaires à partir du fichier `requirements.txt` :

```
pip install -r requirements.txt
```

5.4.4 4. Lancement de l'application en local

Utilisez la commande suivante pour démarrer l'application Streamlit en local :

```
streamlit run app.py
```

où `app.py` est le fichier principal de votre application.

5.4.5 5. Déploiement sur une plateforme en ligne (facultatif)

Il est possible de déployer l'application sur des plateformes telles que :

- **Streamlit Cloud** (<https://streamlit.io/cloud>)
- **Render** (<https://render.com>)
- **Heroku** (<https://www.heroku.com>)

Exemple de déploiement sur Streamlit Cloud :

1. Créer un dépôt sur GitHub et y pousser votre code.
2. Aller sur <https://streamlit.io/cloud> et se connecter.
3. Cliquer sur `New app`, sélectionner le dépôt GitHub.
4. Indiquer le nom du fichier principal (`app.py`).
5. Lancer le déploiement.

5.4.6 6. Mise à jour de l'application

Pour mettre à jour l'application en ligne :

- Effectuer les modifications localement.
- Pousser les changements sur GitHub.
- La plateforme de déploiement redémarre automatiquement l'application avec les nouvelles modifications.

5.4.7 7. Résolution des problèmes

Quelques commandes utiles pour déboguer :

- `streamlit logs` : pour consulter les journaux
- `pip list` : pour vérifier les packages installés

5.5 Interfaces graphiques web

Page d'authentification

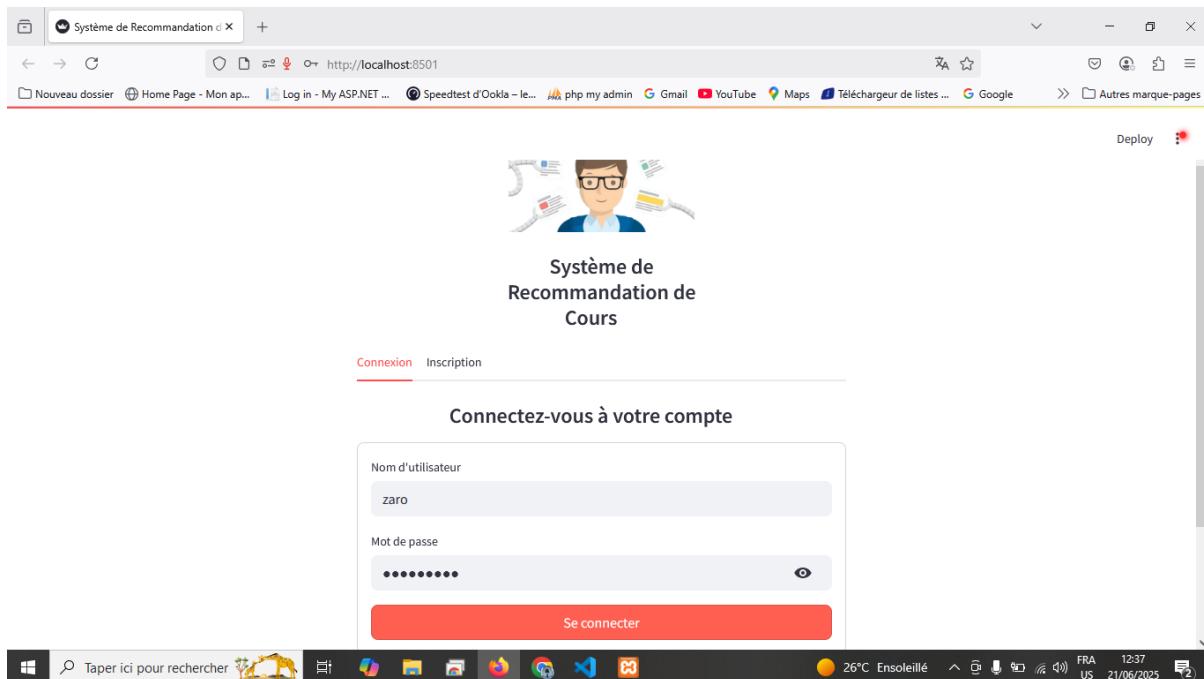


FIGURE 5.7 – Interface de la page d'inscription

Page d'inscription

La page d'inscription permet aux nouveaux utilisateurs de créer un compte sur la plateforme. Elle comprend plusieurs champs à remplir afin de recueillir les informations nécessaires à l'identification de l'utilisateur.

- **Nom d'utilisateur** : Identifiant unique choisi par l'utilisateur.
- **Nom complet** : Prénom et nom de famille de l'utilisateur.
- **Adresse e-mail** : Utilisée pour l'identification et les communications.
- **Mot de passe** : Doit être sécurisé et confidentiel.
- **Confirmation du mot de passe** : Pour s'assurer que le mot de passe a été correctement saisi.

Après avoir complété tous les champs, l'utilisateur clique sur le bouton « **S'inscrire** ».

Message de confirmation : Si l'inscription est réussie, un message s'affiche à l'écran : « *Votre compte a été créé avec succès.* »

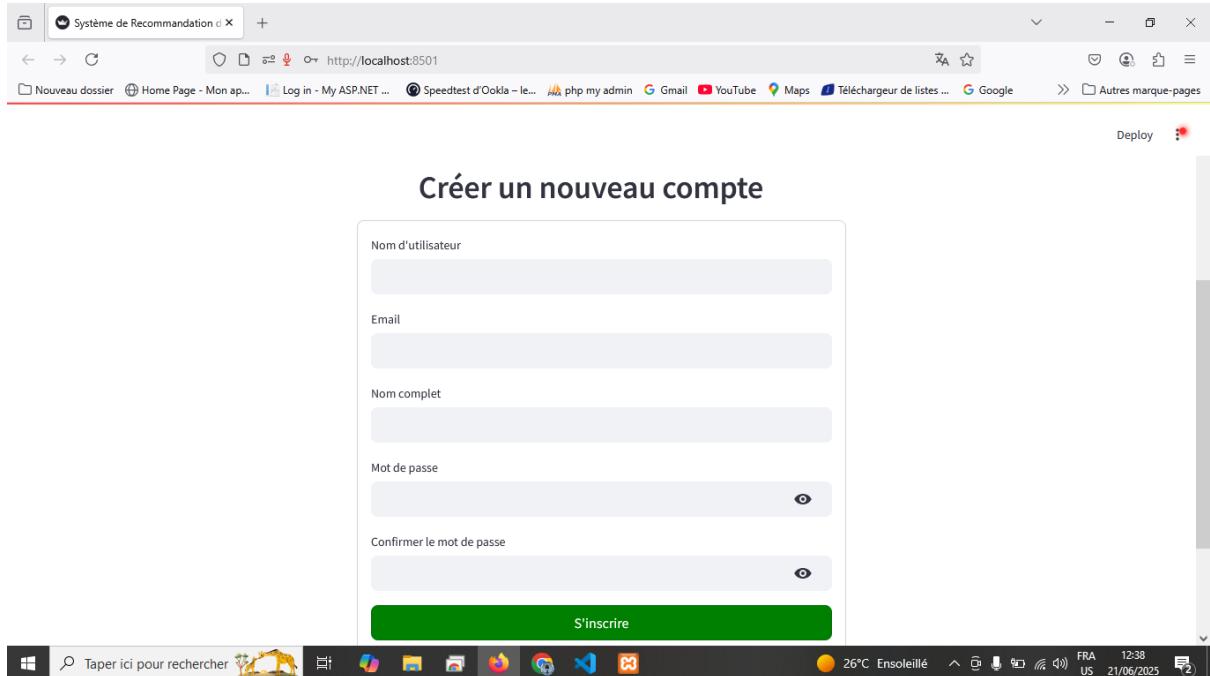


FIGURE 5.8 – Interface de la page d’inscription

Page d'accueil après authentification

Après une authentification réussie, l'utilisateur est redirigé vers la page d'accueil de la plateforme. Cette interface permet d'afficher les informations personnelles de l'utilisateur et de lui proposer diverses fonctionnalités.

- **Nom d'utilisateur et adresse e-mail** : Ces informations sont affichées en haut de la page pour confirmer l'identité de l'utilisateur connecté.
- **Bouton « Se déconnecter »** : Permet à l'utilisateur de se déconnecter de la session en cours.
- **Menu de navigation** : Permet d'accéder aux différentes pages de la plateforme :
 - *Liste des recommandations déjà effectuées* : Affiche les recommandations de cours générées précédemment pour l'utilisateur.
 - *Téléverser un CV* : L'utilisateur peut uploader son CV au format PDF ou Word pour permettre une analyse automatique de ses compétences et centres d'intérêt.
 - *Remplir manuellement un formulaire* : Permet à l'utilisateur de saisir manuellement ses informations (formation, compétences, intérêts) afin d'obtenir une recommandation personnalisée.

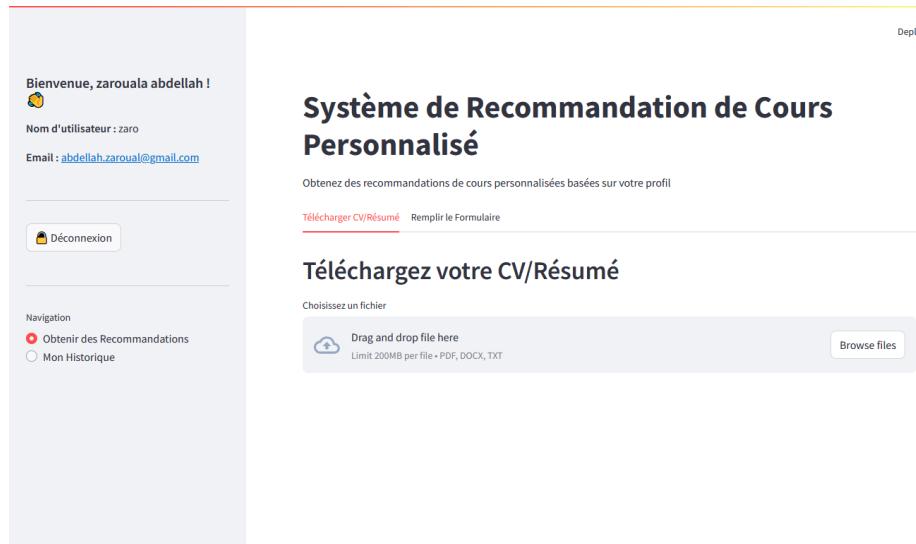


FIGURE 5.9 – Interface de la page d'accueil après connexion

FIGURE 5.10 – Formulaire de saisie manuelle des informations utilisateur

Page historique des recommandations

Cette page permet à l'utilisateur de consulter l'historique de ses recommandations de cours déjà générées par le système. Elle est utile pour garder une trace des suggestions passées et pour revenir sur les propositions faites à différents moments.

Les informations affichées pour chaque recommandation incluent :

- **Date et heure** de la recommandation : Permet à l'utilisateur de savoir quand la suggestion a été générée.
- **Liste des cours recommandés** : Cours proposés par le système en fonction du profil et des données de l'utilisateur (CV ou formulaire).
- **Méthode utilisée** : Indique si la recommandation a été faite à partir d'un CV téléchargé ou d'un formulaire rempli manuellement.

L'historique est affiché sous forme de tableau ou de liste, triée du plus récent au plus ancien.

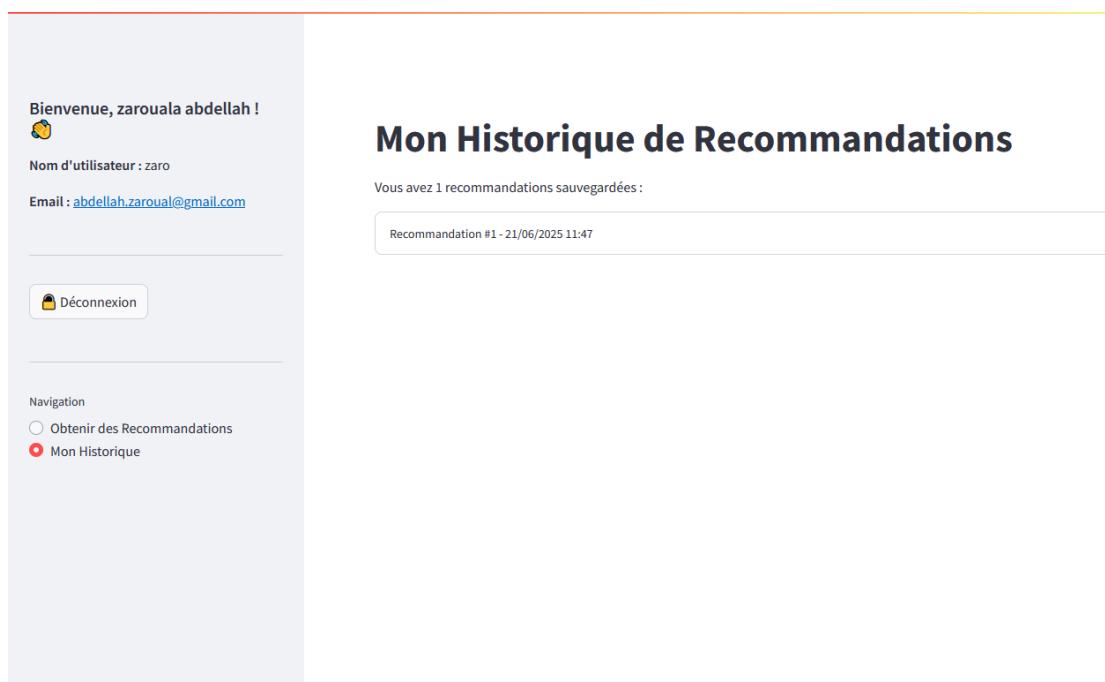


FIGURE 5.11 – Page d'historique des recommandations de cours

Démarche détaillée pour la génération des recommandations

Le système de recommandation repose sur une série d'étapes automatisées, simples à suivre, permettant à l'utilisateur de recevoir des suggestions de cours adaptées à son profil. Voici les étapes détaillées de ce processus :

Étape 1 : Sélection du fichier CV

L'utilisateur commence par cliquer sur le bouton « **Parcourir** » afin d'accéder aux fichiers présents sur son ordinateur. Il peut alors sélectionner un fichier contenant son CV.

- **Formats autorisés** : .pdf, .docx, .txt
- **Action utilisateur** : Sélectionner le fichier à importer.

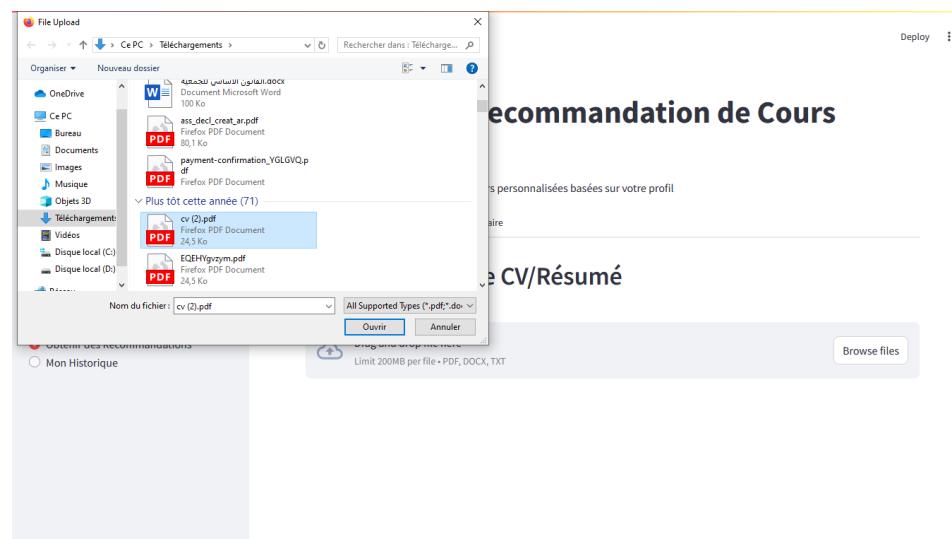


FIGURE 5.12 – Interface de sélection du fichier CV

Étape 2 : Lancement de l'analyse

Une fois le fichier sélectionné, le système démarre automatiquement l'analyse du CV. Un indicateur de chargement (spinner) apparaît avec le message suivant :

« Analyse de votre CV... »

- **Action utilisateur** : Attendre la fin de l'analyse.

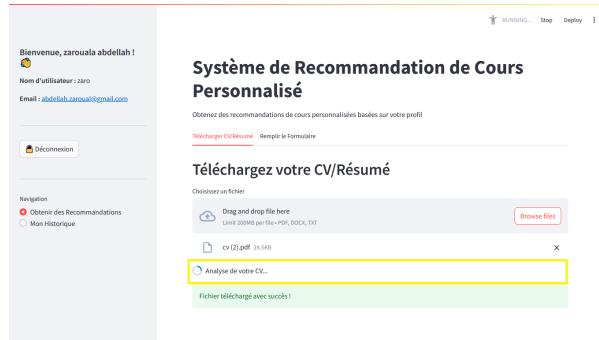


FIGURE 5.13 – Indicateur de chargement pendant l'analyse du CV

Étape 3 : Affichage des informations extraites

Après l'analyse, les données extraites du CV sont affichées dans un formulaire prérempli. L'utilisateur peut les corriger ou les compléter.

- **Informations extraites** : nom, diplôme, compétences, centres d'intérêt, expériences, etc.
- **Action utilisateur** : Vérifier les données et les modifier si nécessaire.
- **Action finale** : Cliquer sur le bouton « Obtenir des recommandations ».

FIGURE 5.14 – Aperçu des données extraites du CV

Étape 4 : Analyse du profil

Une fois le formulaire validé, le système entame la première phase de traitement. Un message et un indicateur de progression sont affichés :

« Étape 1/2 : Analyse de votre profil et sélection des cours... »

- **Action utilisateur** : Attendre pendant le traitement.

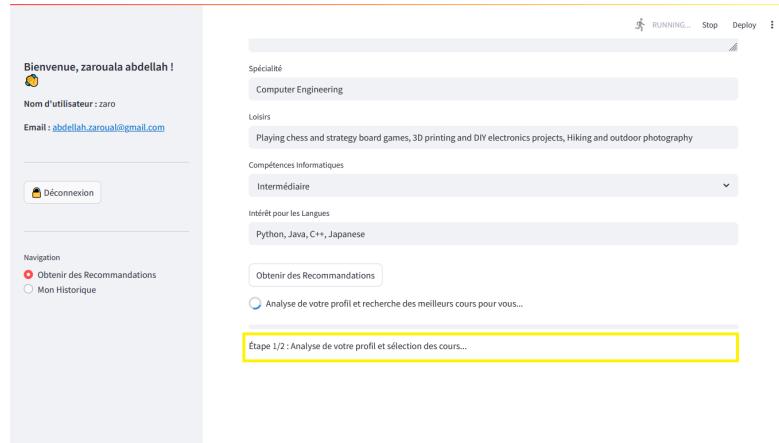


FIGURE 5.15 – Analyse du profil de l'utilisateur

Étape 5 : Crédation des recommandations

Le système passe ensuite à la deuxième phase, consistant à générer les recommandations personnalisées en fonction du profil analysé.

« Étape 2/2 : Crédation de recommandations personnalisées... »

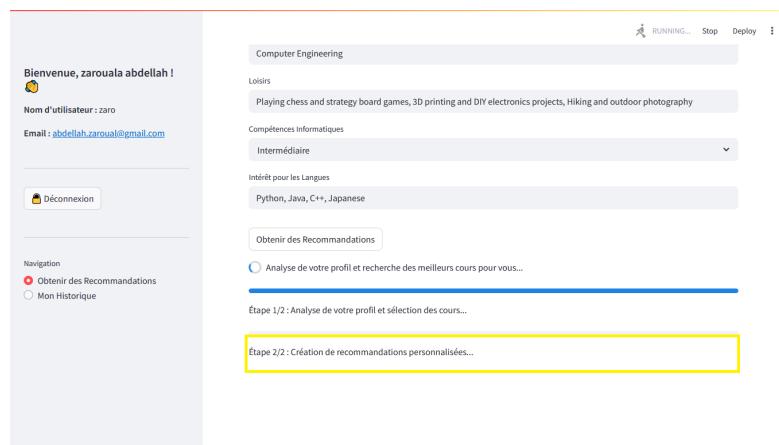


FIGURE 5.16 – Crédation des recommandations personnalisées

Étape 6 : Résultat final

À la fin du processus, le système affiche un message en vert confirmant que les recommandations sont prêtes :

« Recommandations terminées ! »

La liste des cours suggérés est ensuite affichée sur la page avec la possibilité de les consulter ou les enregistrer.

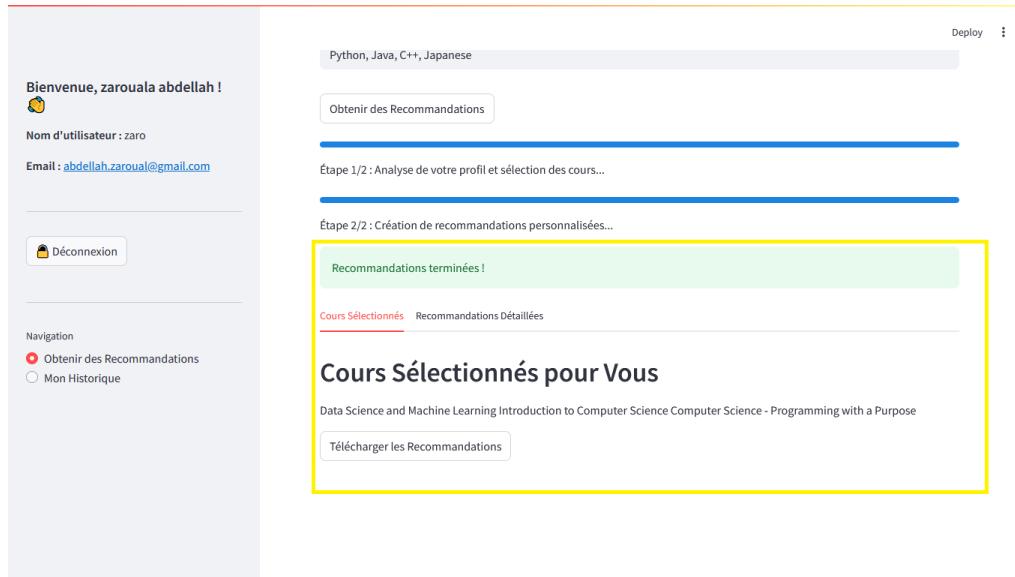


FIGURE 5.17 – Affichage final des recommandations de cours

Conclusion Générale et Perspectives

Ce mémoire a présenté le développement d'un **système de recommandation de cours personnalisé basé sur l'intelligence artificielle générative**, dont le principe repose sur l'analyse automatique d'un CV afin de proposer des formations adaptées au profil de l'utilisateur. Grâce à l'utilisation de technologies récentes et puissantes telles que les modèles de langage de grande taille (*Large Language Models*) et l'approche *Retrieval-Augmented Generation* (RAG), il a été possible de concevoir une solution intelligente, fonctionnelle et évolutive.

Le modèle **LLaMA 3.2**, intégré localement à l'aide de la plateforme **Ollama**, a permis d'effectuer le traitement linguistique de manière fluide tout en assurant la confidentialité des données. Le système peut extraire les informations clés du CV (formations, compétences, expériences, etc.), interroger une base de cours préalablement construite, et proposer des recommandations pertinentes via une interface simple et accessible. L'approche mise en œuvre permet non seulement d'automatiser le processus de recommandation, mais aussi de le personnaliser en fonction du parcours de l'utilisateur.

Ce travail nous a permis de mettre en pratique plusieurs compétences acquises durant la formation, notamment en traitement du langage naturel, en intelligence artificielle, en développement d'interfaces web et en structuration de bases de données. Il a également ouvert la voie à une réflexion sur l'impact de l'IA dans l'éducation, l'orientation professionnelle et la formation tout au long de la vie.

Cependant, bien que les résultats obtenus soient prometteurs, plusieurs **perspectives d'amélioration** peuvent être envisagées pour enrichir le système et en accroître l'efficacité :

- **Enrichissement de la base de cours** : Actuellement limitée, la base de données pourrait être étendue à des catalogues réels de plateformes d'e-learning telles que Coursera, edX, ou Udemy, afin de diversifier les recommandations.
- **Personnalisation avancée** : L'ajout d'un système de profil utilisateur, permettant de renseigner les préférences, objectifs ou contraintes, permettrait d'affiner les recommandations de manière plus précise.
- **Évaluation de la pertinence des suggestions** : Intégrer un système de feedback utilisateur pour mesurer la satisfaction vis-à-vis des cours proposés permettrait une amélioration continue des performances du modèle, notamment à l'aide de l'apprentissage par renforcement.
- **Optimisation de l'extraction d'informations** : Utiliser des techniques plus avancées de traitement du langage naturel (comme la reconnaissance d'entités nommées ou la classification automatique des compétences) pourrait rendre l'analyse du CV plus fine et plus fiable.
- **Déploiement dans un environnement réel** : Une expérimentation au sein d'un établissement universitaire ou d'un centre de formation permettrait d'évaluer le système à grande échelle et de mieux comprendre les besoins des utilisateurs finaux.
- **Sécurité et respect de la vie privée** : Bien que le traitement soit local, des

mesures supplémentaires de sécurité et de conformité (RGPD, chiffrement, anonymisation) peuvent être envisagées pour renforcer la confiance des utilisateurs.

En conclusion, ce projet constitue une contribution concrète à l'intégration de l'intelligence artificielle dans le domaine de la formation personnalisée. Il ouvre des perspectives intéressantes pour le développement de systèmes intelligents capables d'accompagner les individus dans leur parcours professionnel, en tenant compte de leurs compétences, de leurs ambitions et des évolutions du marché du travail.

Bibliographie et Webographie

Ce mémoire s'appuie sur plusieurs ouvrages, articles et ressources en ligne majeurs dans le domaine des systèmes de recommandation et des modèles de langage tels que LLaMA.

Ouvrages et Livres

- [1] présente une vue d'ensemble complète des systèmes de recommandation, incluant méthodes et applications.
- [2] est une référence incontournable en apprentissage profond.

Articles de Recherche

- [3] propose une synthèse approfondie sur les avancées et limites des systèmes de recommandation classiques.
- Le modèle Transformer, base des grands modèles de langage, est introduit dans [4].
- La publication officielle de LLaMA est disponible dans [5].
- [6] offre une vue d'ensemble des systèmes de recommandation basés sur les modèles de fondation.
- [7] présente un modèle de langage léger pour la recommandation générative.
- [8] explore l'utilisation des grands modèles de langage pour la recommandation séquentielle.
- [9] propose une revue systématique des systèmes de recommandation utilisant l'IA générative.
- [10] présente une enquête sur les grands modèles de langage pour la recommandation générative.
- [11] examine les opportunités et défis de GPT-4 pour la recommandation.
- [12] explore l'amélioration des systèmes de recommandation avec les grands modèles de langage.
- [18] introduit les modèles de langage few-shot learners.
- Les autres articles de recherche récents incluent [13], [14], [15], [16], [17], et [19].

Sites Web et Ressources en Ligne

- Un guide pédagogique sur les systèmes de recommandation disponible sur Medium [20].
- La documentation et blog officiels de LLaMA par Hugging Face [21] et Meta AI [22].
- Un article de synthèse sur l'impact des grands modèles de langage dans la recommandation sur Towards Data Science [23].
- Les ressources officielles d'Ollama [24] et les publications de Meta AI [25], [26].
- Les ressources d'OpenAI sur GPT pour la recommandation [27].

Bibliographie

- [1] Ricci, Francesco, Rokach, Lior et Shapira, Bracha. *Recommender Systems Handbook*. Springer, 2^e édition, 2015. <https://link.springer.com/book/10.1007/978-1-4899-7637-6>
- [2] Goodfellow, Ian, Bengio, Yoshua et Courville, Aaron. *Deep Learning*. MIT Press, 2016. <https://www.deeplearningbook.org/>
- [3] Adomavicius, Gediminas et Tuzhilin, Alexander. “Toward the next generation of recommender systems : A survey of the state-of-the-art and possible extensions,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.
- [4] Vaswani, Ashish et al. “Attention is All You Need,” *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [5] Touvron, Hugo et al. “LLaMA : Open and Efficient Foundation Language Models,” *arXiv preprint arXiv:2302.13971*, 2023. <https://arxiv.org/abs/2302.13971>
- [6] Huang, Chengkai et al. “A Survey of Foundation Model-Powered Recommender Systems : From Feature-Based, Generative to Agentic Paradigms,” *arXiv preprint arXiv:2504.16420*, 2025. <https://arxiv.org/abs/2504.16420>
- [7] Mei, Kai et Zhang, Yongfeng. “LightLM : A Lightweight Deep and Narrow Language Model for Generative Recommendation,” *arXiv preprint arXiv:2310.17488*, 2023. <https://arxiv.org/abs/2310.17488>
- [8] Harte, Jesse, Zorgdrager, Wouter et al. “Leveraging Large Language Models for Sequential Recommendation,” *arXiv preprint arXiv:2309.09261*, 2023. <https://arxiv.org/abs/2309.09261>
- [9] Ayemowa, M. O. et al. “Analysis of Recommender System Using Generative Artificial Intelligence : A Systematic Literature Review,” *IEEE Transactions*, 2024. https://www.researchgate.net/publication/381603751_Analysis_of_Recommender_System_Using_Generative_Artificial_Intelligence_A_Systematic_Literature_Review
- [10] Chen, Lichen et al. “Large Language Models for Generative Recommendation : A Survey,” *COLING 2024*. <https://www.comp.hkbu.edu.hk/~lichen/download/COLING24-LLM4RS-paper.pdf>
- [11] Tang, Jian et al. “GPT-4 for Recommendation : Opportunities and Challenges,” *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023. <https://dl.acm.org/doi/10.1145/3583780.3614889>
- [12] Zhang, Jing et al. “RecsysLLM : Enhancing Recommendation with Large Language Models,” *AAAI Conference on Artificial Intelligence*, 2023. <https://arxiv.org/abs/2308.04139>
- [13] Liu, Peng et al. “Prompting Large Language Models for Interactive Recommender Systems,” *WWW Conference*, 2023. <https://dl.acm.org/doi/10.1145/3539597.3570458>

- [14] Shen, Xiaoyang et al. “LLMRec : A Large Language Model Based Recommender System,” *SIGIR*, 2023. <https://arxiv.org/abs/2307.01489>
- [15] Xu, Huan et al. “Generative AI for Personalized Recommendation,” *IJCAI*, 2024. <https://www.ijcai.org/Proceedings/2024/>
- [16] Li, Jing et al. “ChatRec : Conversational Recommender System based on ChatGPT,” *arXiv preprint arXiv :2310.12042*, 2023. <https://arxiv.org/abs/2310.12042>
- [17] Wei, Feng et al. “GPT-4 for Context-Aware Recommender Systems,” *RecSys*, 2023. <https://dl.acm.org/doi/10.1145/3607451.3607824>
- [18] Brown, Tom et al. “Language Models are Few-Shot Learners,” *NeurIPS 2020*. <https://arxiv.org/abs/2005.14165>
- [19] Park, Juho et al. “Towards Generative Recommendation Models with Pretrained Language Models,” *CIKM*, 2024. <https://dl.acm.org/doi/10.1145/3459637.3482277>
- [20] Doe, John. *Understanding Recommender Systems : A Beginner’s Guide*. Medium, 2023. Consulté le 23 juin 2025. <https://medium.com/@john doe/understanding-recommender-systems-a-beginners-guide-123456789abc>
- [21] Hugging Face. *Introduction to LLaMA Models*. 2023. Consulté le 23 juin 2025. <https://huggingface.co/blog/llama>
- [22] Meta AI. *LLaMA : Open and Efficient Foundation Language Models*. 2023. Consulté le 23 juin 2025. <https://ai.facebook.com/blog/large-language-model-llama-meta-ai/>
- [23] Smith, Jane. *How Large Language Models Are Transforming Recommendation Systems*. Towards Data Science, 2023. Consulté le 23 juin 2025. <https://towardsdatascience.com/how-large-language-models-are-transforming-recommendation-systems-abcdef12345>
- [24] Liu, Wei et al. *Ollama : Open Large Language Model for Multi-task Applications*. Ollama Documentation, 2023. Consulté le 10 mai 2025. <https://ollama.ai/paper.pdf>
- [25] Tang, Zhe et al. *LLaMA : Open and Efficient Foundation Language Models*. Meta AI Research, 2023. Consulté le 30 juin 2025. <https://arxiv.org/abs/2302.13971>
- [26] Gao, T., et al. *LLaMA : Open and Efficient Foundation Language Models*. Meta AI, 2023. Consulté le 20 mai 2025. <https://ai.meta.com/llama/>
- [27] Radford, Alec et al. *GPT Models for Recommendation Systems : A New Paradigm*. OpenAI Blog, 2023. Consulté le 5 juin 2025. <https://openai.com/research/gpt-for-recommendation>