République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Abou Bakr Belkaid – Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études pour l'obtention du diplôme de Master en Informatique

Option : Système d'information et de connaissance & Modèle Intelligents et Décision (SIC &MID)

Thème

Interrogation d'une Base de Données Relationnelle en Langage Naturel à l'aide Des Grands Modèles de Langage

Réalisé par:

- Melle Rezzoug Nesrine Fatima Zohra
- Melle Sadaoui Hadjer

Présenté le 25/06/2024 devant le jury composé de :

- Mr Hadjila FethAllah *(Président)*
- Mr Belabed Amine *(Encadrant)*
- Mr Smahi Mohamed Ismail *(Examinateur)*

Année universitaire : 2023-2024

Remerciements

Au seuil de la conclusion de mon parcours d'études je ressens le besoin de témoigner ma sincère reconnaissance envers plusieurs personnes et entités qui ont contribué à mon cheminement.

Tout d'abord, je souhaite exprimer mon sincère remerciement à Dieu le Tout-Puissant. Je le remercie d'avoir illuminé mon chemin et de m'avoir donné la force et la foi nécessaires pour surmonter chaque épreuve. Je suis également reconnaissante envers lui pour toutes les bénédictions, l'inspiration et la guidance qu'il m'a accordée tout au long de mon chemin académique.

À mon encadrant, Mr BELABED Amine, je souhaite exprimer toute ma reconnaissance pour vos compétences et votre encadrement tout au long de ce projet. Ce travail n'aurait pas pu aboutir sans votre suivi attentif, vos conseils avisés et votre assistance précieuse. Je suis reconnaissant pour votre accueil chaleureux et vos conseils inestimables.

Je tiens à vous remercier sincèrement pour votre soutien constant et pour être une source d'inspiration constante. C'est un honneur pour moi d'avoir eu l'opportunité de travailler sous votre direction.

Je tiens à exprimer toute ma gratitude envers les membres du jury Mr HADJILA FethAllah et Mr SMAHI Mohamed Ismail pour leur précieuse évaluation de mon mémoire.

Je souhaite également exprimer ma profonde gratitude envers tous les enseignants du département d'informatique pour leur expertise scientifique et leurs compétences pédagogiques, Leur contribution a été essentielle à mon parcours académique, et je leur suis sincèrement reconnaissant pour leur engagement à partager leur savoir.

Enfin, je tiens à adresser des remerciements chaleureux à tous mes proches à mes parents et à toutes les personnes qui m'ont soutenue, de près ou de loin,dans la réalisation de ce travail. Leurs encouragements et leur soutien moral ont été d'une importance capitale, et je leur suis profondément reconnaissante.

Dédicace

Avec tout mon respect et l'expérience de ma reconnaissance, je dédie ma remise de diplôme et Ma joie À

Mon cher papa Hamza

Ce travail est le fruit de tes sacrifices que tu as consentis pour mon
Éducation et ma formation. Sans ton aide, tes conseils et tes encouragements ce travail
N'aurait vu le jour.

Ma chère Maman Nawel

Aucune dédicace ne saurait exprimer l'amour, l'estime, le dévouement et le
Respect que j'ai toujours eu pour vous. Rien au monde ne vaut les efforts fournis jour et
Nuit pour mon éducation et mon bien être.

Ma Sœur Adorée Chérine

Ta présence apaisante et ton humour rayonnant ont enrichi ma vie.
Cette réussite, c'est aussi la tienne, car tu as été ma lumière à travers les moments difficiles.
Merci pour tout ce que tu es pour moi.

Mes frères bien-aimés Réda et Rayane

Votre soutien constant et votre présence ont été essentiels dans ce voyage
Merci d'avoir été mes piliers et d'avoir partagé ce parcours avec moi. Cette réussite est aussi
La vôtre. Avec toute ma gratitude et mon affection, je vous dédie cette victoire.

À toute ma chère famille À mes chères grands-mères, que Dieu leur accorde une longue vie, et à mes grands-pères bien-aimés, que Dieu bénisse leurs âmes et les accueille dans ses vastes paradis.

Mes chers amis

Merci à tous mes amis qui m'ont soutenu de près ou de loin et qui ne m'ont
Jamais oublié. Vous étiez Différents mais chacun d'entre vous a su comment apporter
Le bonheur à ma vie.

Ce travail est le fruit de vos efforts collectifs et de votre soutien Je vous aime tous.

Rezzoug Nesrine Fatima Zohra

Dédicace

{وَآخِرُ دَعْوَاهُمْ أَنِ الْحَمْدُ لِلَّهِ رَبِّ الْعَالَمِينَ}

Avant tout, je remercie ALLAH, qui m'a aidé et m'a donné la patience pour accomplir ce travail.

Je dédie ce travail :

À celle qui m'a donné la vie, qui s'est sacrifié pour mon bonheur et ma réussite, **ma tendre mère**.

À celui qui a toujours été à mes côtés pour me soutenir et qui a été mon ombre durant toutes les années des études et ma vie **mon cher père**.

À mon frère **Yahya**, Pour son soutien inconditionnel et son inspiration constante.

À mes meilleurs amis **Douaa** et **Manel**, pour votre amitié sincère et votre soutien indéfectible.

À mes chères cousines **Ikram**,**Sara**,**Rabiaa** et **Amira** pour leur présence chaleureuse et leur soutien précieux.

À la petite **Assinat** trésor de la famille, je lui souhaite une vie épanouissante, pleine de bonheur et de succès.

À mes chers collègues qui m'ont aidé, encouragé et conseillé **Mohammed**, **Abdennour** et **Ihab**.

À tous les membres de ma famille, **SADAOUI** et **DJELTI**.

À tous ceux qui n'ont jamais cessé de veiller sur mon avenir et qui ont essayé de me soutenir d'une manière ou d'une autre au cours de mon processus éducatif.

La dernière dédicace est pour ma petit sœur **Amina Ferdaous** à qui je souhaite le plus meilleur des parcours et une vie saine et joyeuse.

Sadaoui Hadjer

Table Des Matières

Liste Des Figures

Liste Des Tableaux

Liste Des Abréviations

LLM : Large Language Model

GPT: Generative pre-trained transformers

LLaMA: Large Language Model Association

PaLM: Path-Augmented Language Model

BERT: Bidirectional Encoder Representations from Transformers

NLP: Natural Language Processing

SQL: Structured Query Language

ToT: Tree of thoughts

IA : Intelligence Artificielle

API : Application Programming Interface

Introduction Générale

I. Contexte et problématique

Dans notre société actuelle, où les données jouent un rôle central dans nos décisions, l'accès aux informations significatives des bases de données est crucial pour une prise de décision éclairée.

Cependant, les méthodes traditionnelles d'interrogation des bases de données, principalement basées sur le langage SQL, posent des défis pour les utilisateurs non techniques. Parmi les défis majeurs, nous pouvons citer :

- **Complexité du Langage SQL** : Cela signifie que seuls les utilisateurs ayant des compétences techniques avancées peuvent interagir directement avec la base de données. Les utilisateurs non techniques, comme les responsables marketing ou les gestionnaires, doivent souvent dépendre des analystes de données pour obtenir les informations dont ils ont besoin. Cette dépendance crée des goulets d'étranglement, ralentit la prise de décision et limite l'autonomie des employés.
- **Temps et Efficacité :** La création de requêtes SQL, en particulier pour des requêtes complexes, peut prendre beaucoup de temps, même pour des utilisateurs expérimentés. Cela réduit l'efficacité opérationnelle et augmente le temps nécessaire pour obtenir des insights précieux. Les entreprises perdent ainsi en agilité, car le délai pour accéder à des informations critiques peut retarder les actions et les décisions stratégiques.

L'utilisation du langage naturel pour interroger les bases de données émerge comme une solution prometteuse, offrant une approche plus intuitive et accessible à tous les utilisateurs, quelle que soit leur expertise technique.

La transition vers l'interrogation en langage naturel présente cependant d'autres défis tels que **la compréhension sémantique** et **l'ambiguïté des requêtes**, nécessitant une approche Multidisciplinaire impliquant le traitement du langage naturel, la gestion de base de données et la conception d'interfaces utilisateur.

II. Contribution

Ce mémoire explore la problématique de l'interrogation d'une base de données relationnelle en langage naturel. Nous proposons une solution à l'aide des grands modèles de langage (LLMs), pour faire le mapping entre le langage naturel et les requêtes SQL. Les LLMs sont formés sur d'énormes corpus de texte, ce qui leur confère une compréhension approfondie des structures-Grammaticales et des nuances sémantiques du langage naturel. Cette capacité permet aux utilisateurs de poser des questions complexes sans avoir à se soucier de la syntaxe spécifique des requêtes SQL ou d'autres langages de requête. Nous rappelons que l'objectif de ce mémoire n'est pas de proposer une nouvelle approche, mais de réaliser une implémentation fonctionnelle et utile en s'initiant au domaine émergent des LLMs.

Du fait, nous avons implémenté une application nommé « Chat with MySQL » qui permet à un utilisateur d'interroger et tirer des informations d'une base de données en langage naturel.

III. Structure de Mémoire

Le mémoire est structuré en trois chapitres :

- Une introduction générale au départ où nous avons décrit le contexte, la problématique et la contribution de notre travail.
- Dans le premier chapitre, intitulé **«Les grands modèles de langage (LLMs)»**, nous avons donné une vue générale sur les LLMs, de leurs principales familles, leur architecture, ainsi que leurs avantages et défis.
- Dans le deuxième chapitre, intitulé **«Prompt Engineering »**, nous avons exploré l'importance du prompt engineering, les principaux composants d'un prompt ainsi que les différentes techniques du prompt engineering.
- Le troisième chapitre **«Conception et implémentation »** est dédié aux détails de conception et d'implémentation de notre approche d'interrogation d'une base de données en langage naturel. Nous avons également présenté les différents outils et technologies utilisés tout au long du développement de notre application « Chat with MySQL ».
- Nous clôturons ce mémoire par une conclusion générale qui résume notre travail et présente quelques perspectives.

Chapitre

1

Les Grands Modèles de Langage (LLMs)

I.1 Introduction

Les progrès fulgurants dans le domaine de l'intelligence artificielle ont récemment été marqués par l'émergence des grands modèles de langage, suscitant un intérêt sans précédent tant dans la communauté scientifique que dans le grand public. Ces modèles, également connus sous le nom de LLM (Large Language Models), sont des systèmes d'intelligence artificielle capables de comprendre et de générer du langage naturel de manière avancée.

Dans ce contexte, le présent chapitre donne une vue générale sur les grands modèles de langage. Il introduit principalement la définition des LLMs, leurs architectures et leur fonctionnement, leurs différents types ainsi que leurs principaux domaines d'application.

I.2 Définition des LLMs

Les grands modèles de langage (LLM) ou grands modèles de langage sont des algorithmes d'apprentissage profonds qui prennent la forme d'un type de modèles de traitement du langage naturel (NLP) basés sur un réseau de neuronaux , Ces Modèles sont pré-entraînés sur de grands ensembles de données textuelles. Ces modèles sont conçus pour comprendre et générer le langage humain à la manière d'un humain.

Le LLM est capable d'analyser et de traiter de grands volumes de texte et possède une riche compréhension du contexte et de la structure du langage, ce qui lui permet d'effectuer une variété de tâches de traitement du langage naturel telles que la traduction automatique, la génération de texte, la classification de texte.

I.3 Différence entre NLP et LLM

Le terme "NLP" fait référence au Traitement du Langage Naturel, un domaine de l'IA. Il implique la création d'algorithmes. Le domaine du NLP est plus vaste que celui du LLM, englobant des algorithmes et des méthodes.

NLP gouverne deux approches l'apprentissage automatique et l'analyse des données linguistiques. Les applications de NLP sont :

- Automatiser les tâches quotidiennes
- Améliorer la recherche
- Optimiser les moteurs de recherche
- Classification de grands ensembles de documents
- Étude des réseaux sociaux

Par contre, LLM est un modèle de langage vaste et se concentre davantage sur le texte de type humain, offrant ainsi une création de contenu et des conseils sur mesure. [1]

I.4 Familles de Grands Modèles de Langage

Les grands modèles de langage (LLMs) sont des modèles de traitement de texte très gros etpuissants. Parmi eux, il y a trois grandes familles : GPT, LLaMA et PaLM *(Voir Figure I.1)* :

Famille GPT (Generative Pretrained Transform-ers) : La famille GPT a été créée par Open AI. GPT-3 est le plus connu, avec 175 milliards de paramètres. Il peut apprendre de nouveaux concepts sans formation supplémentaire et excelle dans de nombreuses tâches, comme traduire et répondre à des questions. CODEX est un GPT spécialement conçu pour écrire du code à partir d'un langage naturel. WebGPT est comme un navigateur Web qui répond aux questions. [2]

La Série LLaMA : comprend des modèles de langage de base publiés par Meta et disponibles en open source sous une licence non commerciale. Ils surpassent généralement les modèles GPT sur de nombreux benchmarks. Le modèle LLaMA utilise une architecture de transformation avec des modifications mineures. La famille LLaMA comprend des modèles tels que guanaco, koala et mixtral-7b, qui offrent des performances et des fonctionnalités spécifiques pour différentes applications en langage naturel. [2]

La famille PaLM : Développé par Google, il inclut des modèles de langage basés sur Transformer, avec son premier modèle annoncé en avril 2022 et rendu public en mars 2023, avec 540 milliards de paramètres. Ces modèles sont formés sur de grands corpus

de textes et démontrent des performances de pointe en matière d'apprentissage à partir de quelques exemples et de compréhension et de génération du langage. Ils sont également personnalisés pour des domaines spécifiques, tels que Med-PaLM Médicine. [2]

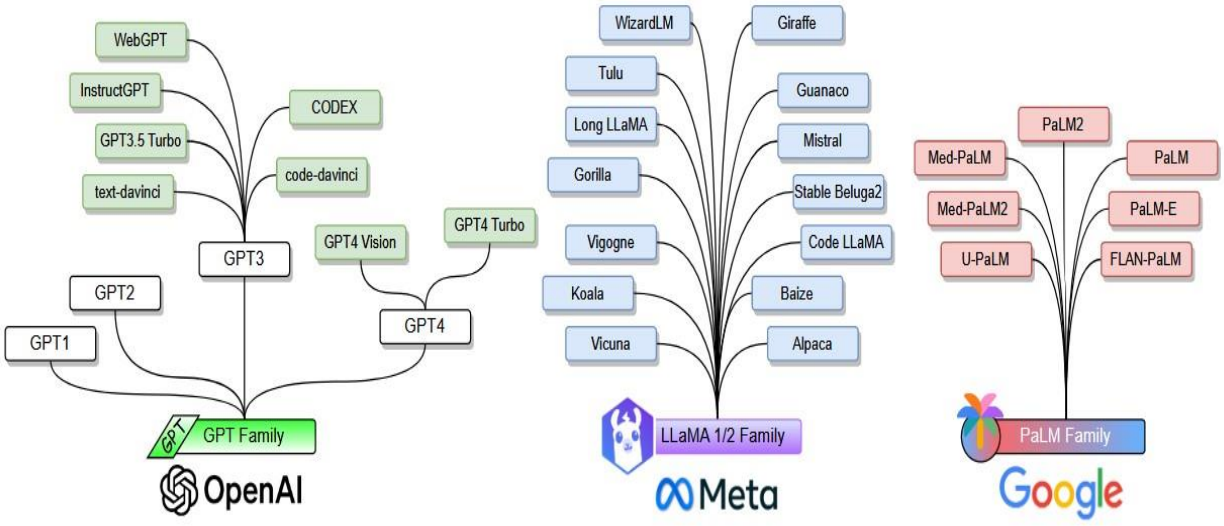

Figure I. 1: Les Familles Populaires de Grands Modèles de Langage [2]

I.5 Architecture de Base des LLMs

Les transformeurs (ou Transformateurs) se sont imposés comme l'architecture principale pour la modélisation des LLMs. Bien que les transformeurs fournissent des directives pour l'architecture du modèle, il existe encore des décisions de conception de haut niveau qui peuvent être prendre dans ce cadre. Cette section se base principalement sur l'architecture et les différents types de transformeurs.

I.5.1 Définition d'un modèle de transformateur

Un transformeur (ou Transformeurs) est une architecture de réseau neurones utilisée pour modéliser le LLMs. Il se base sur des mécanismes d'attention pour comprendre les relations entreles différentes parties d'un ensemble de mots en fonction du contenu et de la position du mot.

Ces mécanismes d'attention aident le modèle à saisir le contexte et l'ordre des mots dans une phrase, ce qui est crucial pour une compréhension précise du langage.

Contrairement aux réseaux neuronaux récurrents, les calculs dans les transformateurs peuvent être effectués en parallèle, accélérant considérablement le processus d'apprentissage et de traitement des données textuelles. [3]

I.5.2 Types de transformateur dans Les Modèles LLMs

Les transformateurs se composent de deux modules clés : un encodeur et un décodeur *(Voir Figure I.2).*

Ces modules peuvent être autonomes ou combinés, ce qui permet trois types de transformateurs :

Encodeur seul - un encodeur traduit les jetons en une représentation numérique sémantiquement significative (c'est-à-dire des plongements) en utilisant l'auto-attention. Les plongements prennent en compte le contexte. Ainsi, le même mot/jeton aura différentes représentations en fonction des mots/jetons qui l'entourent. Ces transformateurs fonctionnent bien pour les tâches nécessitant une compréhension de l'entrée, telles que la classification de texte ou l'analyse de sentiment. Un modèle d'encodeur seul populaire est le BERT de Google. [3]

Décodeur seul - un décodeur, comme un encodeur, traduit les jetons en une représentation numérique sémantiquement significative. La différence clé, cependant, est qu'un décodeur ne permet pas l'auto-attention avec les éléments futurs d'une séquence (appelée auto-attention masquée). Un autre terme pour cela est la modélisation du langage causale, ce qui implique l'asymétrie entre les jetons futurs et passés. Cela fonctionne bien pour les tâches de génération de texte et est la conception sous-jacente de la plupart des LLMs (par exemple, GPT-3, Llama, Falcon, et bien d'autres). [3]

Encodeur-Décodeur - nous pouvons combiner les modules d'encodeur et de décodeur pour créer un transformateur encodeur-décodeur. C'était l'architecture proposée dans le document original "Attention is all you need" [4] La caractéristique clé de ce type de transformateur (non possible avec les autres types) est l'attention croisée. En d'autres

termes, au lieu de restreindre le mécanisme d'attention à apprendre des dépendances entre les jetons dans la même séquence, l'attention croisée apprend des dépendances entre les jetons dans différentes séquences (c'est-à-dire des séquences des modules encodeur et décodeur). Cela est utile pour les tâches génératives nécessitant une entrée, telles que la traduction, le résumé, ou les questions-réponses. Les noms alternatifs pour ce type de modèle sont le modèle de langue masquée ou l'auto encodeur de débruitage. Un LLM populaire utilisant cette conception est le BART de Meta. [3]

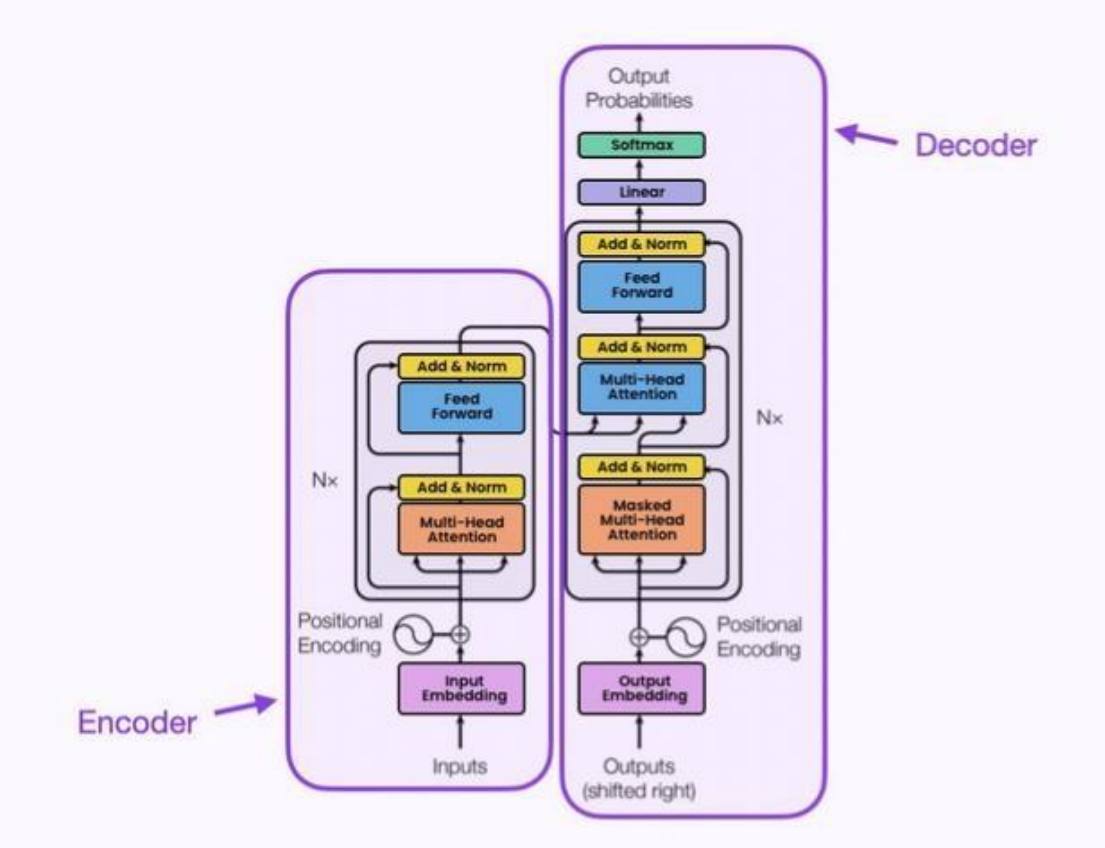

Figure I. 2: Modèle Transformeur : Architecture [5]

I.6 Principaux composants des grands modèles de langage

Les modèles basés sur des transformateurs qui ont révolutionné le traitement du langage naturel suivent généralement une architecture commune qui comprend les composants suivants :

Intégration des entrées : Le texte saisi est symbolisé en petites unités telles que : un mot ou un sous mot. Chaque jeton est intégré dans une représentation vectorielle continue. Cette étape d'intégration capture les informations sémantiques et syntaxiques de l'entrée. [6]

Codage de position : Cette fonction aide le modèle à savoir où chaque mot se trouve dans une phrase en leur attribuant des valeurs numériques selon leur position. [5]

Encodeur : L'encodeur est basé sur la technologie de réseau neuronal, Il segmente le texte d'entrée en jetons, tels que des mots, puis utilise des couches de réseaux neuronaux avec l'auto- attention pour générer des états cachés qui saisissent le sens et le contexte du texte. Plusieurs couches de l'encodeur sont impliquées dans ce processus. [5]

1. **Mécanisme d'attention:** Ces composants sophistiqués aident le modèle d'IA à prioriser certains éléments du texte d'entrée par rapport à d'autres lors de la génération d'une sortie. Pour illustrer, dans une phrase comportant une gamme de sentiments différents, un système d'attention pourrait accorder une importance accrue aux mots qui expriment des émotions. Cette stratégie permet à l'IA de générer des réponses contextuelles plus précises et nuancées. [7]
2. **Réseau neuronal d'alimentation en avant:** Après l'étape d'attention, un réseau neuronal d'avance d'alimentation est appliqué à chaque jeton indépendamment. Ce réseau comprend des couches entièrement connectées avec des fonctions d'activation non linéaires, permettant au modèle de capturer des interactions complexes entre les jetons. [6]

Couche décodeur : dans certaines conceptions basées sur un transformateur, un composant décodeur est inclus en plus de l'encodeur. Qui utilise les encodages d'entrée et de sortie pour produire du texte en langage naturel. Il anticipe le mot suivant en fonction du contexte appris.[7]

Attention multi-têtes : Les transformateurs utilisent souvent l'attention multi-têtes, où l'auto-attention est effectuée simultanément avec différents poids d'attention appris. Cela permet au modèle de capturer différents types de relations et de traiter

simultanément différentes parties de la séquence d'entrée**.** [6]

Normalisation des couches : la normalisation des couches est appliquée après chaque sous-composant ou couche de l'architecture du transformateur. Cela aide à stabiliser le processus d'apprentissage et améliore la capacité du modèle à se généraliser à différentes entrées. [6]

Couche de sortie : la couche de sortie du modèle de transformateur peut varier en fonction de la tâche spécifique. Par exemple, dans la modélisation du langage, la projection linéaire et l'activation SoftMax sont souvent utilisées pour générer une distribution de probabilité sur le jeton suivant. [6]

I.7 La construction d'un LLM

La construction d'un LLM passe par plusieurs étapes et utilise plusieurs techniques, comme : la Nettoyage des données, Tokenisation, l'encodage, l'entraînement et l'ajustement du modèle

(Voir Figure I.3), La section suivante résume ces principales étapes et techniques :

- **Nettoyage des données** : Le nettoyage des données, y compris la suppression du bruit et la déduplication, est crucial pour garantir la qualité des données d'entraînement des modèles de langage. Des techniques avancées sont utilisées pour filtrer les données bruitées et éliminer les duplicatas. [8]
- **Tokenisation** : La tokenisation consiste à diviser le texte en unités plus petites appelées tokens. Les algorithmes comme BytePairEncoding et WordPieceEncoding sont utilisés pour créer des vocabulaires de tokens couvrant un large éventail de mots. [9]
- **Encodage de position :** L'encodage de position est crucial pour que les modèles de langage comprennent l'ordre des mots dans une séquence. Différentes approches, telles que les encodages absolus, relatifs et rotatifs, sont utilisées pour incorporer des informations de position dans les modèles.
- **Pré-entraînement du modèle :** Le pré-entraînement est la première étape dans la construction des LLM, où les modèles sont formés sur de grandes quantités de données non étiquetées. Des approches comme la prédiction du jeton suivant et la modélisation de langage masquée sont utilisées pour pré-entraîner les modèles.

- **Ajustement fin (fine tunning) et ajustement des instructions :** L'ajustement fin consiste à adapter un modèle pré-entraîné à une tâche spécifique avec des données étiquetées. L'ajustement des instructions vise à aligner les réponses du modèle sur les attentes humaines en fournissant des instructions spécifiques.
- **Alignement** : L'alignement garantit que les modèles de langage sont conformes aux objectifs humains et aux normes éthiques et sociales. Cela comprend l'évaluation de l'alignement des versions, des langues, des domaines et des normes éthiques et sociales.

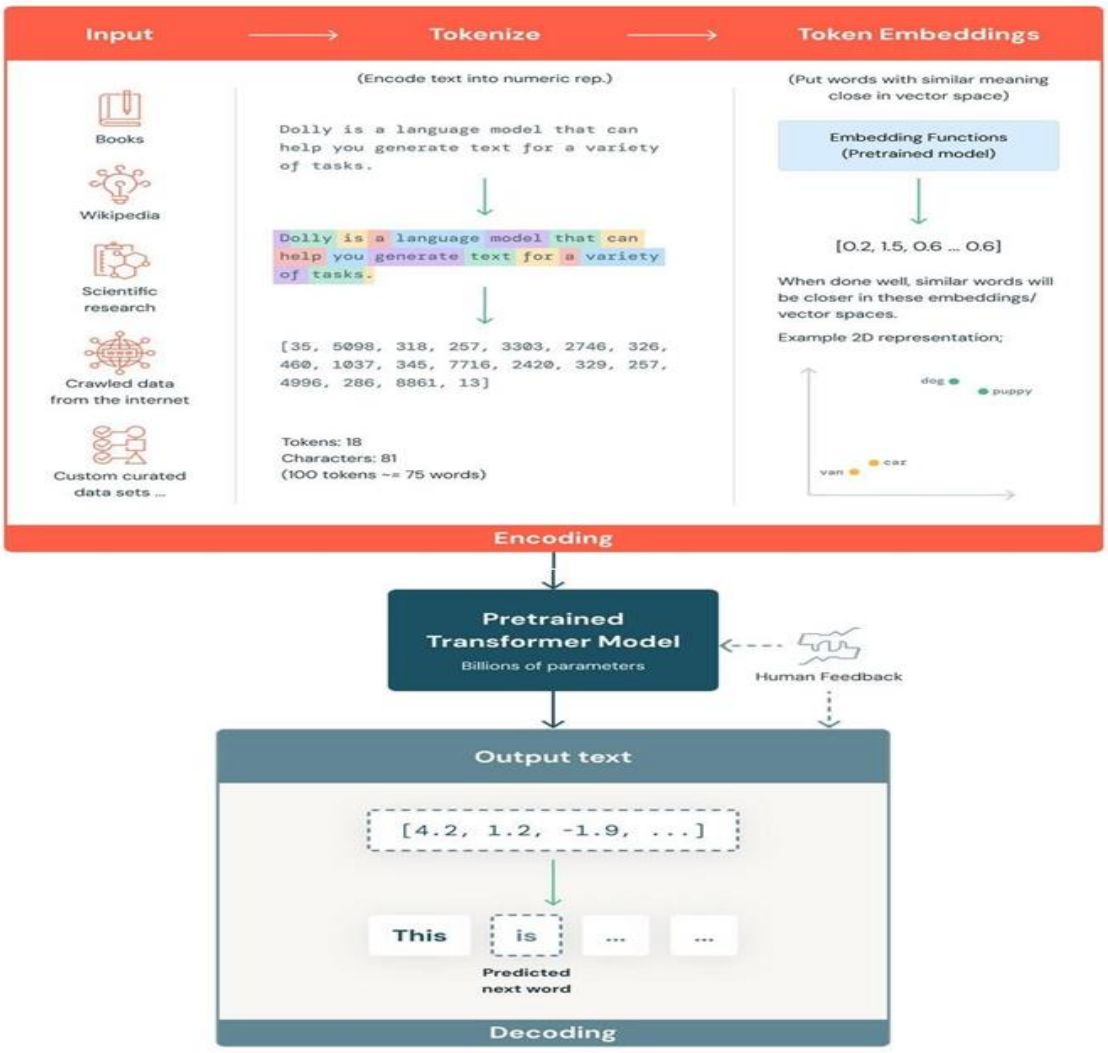

Figure I. 3: Une version simplifiée du processus d'entraînement des LLM [10]

I.8 Les applications des LLM

Un LLM peut avoir un impact dans de nombreux domaines et dans tous les domaines. Voici quelques cas d'utilisation typiques :

- **Bots conversationnels et assistants virtuels :** LLM est utilisé pour fournir des chatbots aux clients et aux employés des entreprises. Ces robots peuvent fournir une assistance, assurer le suivi des contacts via le site Web ou agir comme assistants personnels.
- **Génération de code et débogage :** La capacité à générer du texte sur n'importe quel sujet sur lequel le LLM a été entraîné est un cas d'utilisation principal.
- **Analyse des sentiments :** La plupart des LLM peuvent être utilisés pour l'analyse de sentiment pour comprendre automatiquement la satisfaction du client.
- **Classification et regroupement de textes :** Un LLM est capable de classifier et de catégoriser le contenu.
- **Traduction :** Les LLM peut traduire des documents et des pages Web dans différentes langues.
- **Synthèse et paraphrase :** Les LLM peuvent résumer des articles, des publications, des appels clients ou des réunions en mettant en évidence les points les plus importants, ce qui est une fonction utile.
- **Génération de contenu :** Le LLM a la capacité de produire du contenu sur n'importe quel sujet sur lequel il a été formé. [11]

I.9 Limites et défis de la performance des LLM dans des contextes spécifiques

Hallucinations: Les hallucinations se produisent lorsqu'il y a des sorties erronées ou non intentionnelles des Grands Modèles de Langage. Ils pourraient, par exemple, déclarer ressentir des émotions, Ils ne peuvent pas toujours, en raison de leur prédiction syntaxelle, interpréter correctement le sens de ce que dit l'utilisateur, ce qui peut entraîner des erreurs d'interprétation, connues sous le nom d'hallucinations. [12]

Sécurité: Les Grands Modèles de Langage représentent des dangers de sécurité importants si leur gestion ou leur surveillance ne sont pas adéquates. Ces dangers comprennent la divulgation d'informations personnelles, les fraudes par phishing et la génération de courriels indésirables(Spam). En outre, les individus malveillants ont la capacité de reprogrammer ces modèles afin de diffuser des informations erronées, ce qui peut entraîner des conséquences graves à l'échelle mondiale. [12]

Biais: L'utilisation des données pour entraîner les modèles de langage peut avoir un impact sur les résultats qu'ils produisent. Les sorties des Grands Modèles de Langage seront influencées par les biais ou la diversité des données, ce qui peut conduire à une représentation erronée ou partielle de la réalité. [12]

Consentement: Les Grands Modèles de Langage sont fréquemment élaborés à partir d'énormes ensembles de données, dont certains peuvent avoir été obtenus sans l'accord des personnes. Cela pose des difficultés en ce qui concerne le respect des droits de copyright, le plagiat et l'utilisation non autorisée de contenus privés. En outre, il est possible de compromettre l'anonymat des créateurs, ce qui expose les utilisateurs à des risques de violation de la vie privée. [12]

Scaling et Déploiement: Le développement et la mise en place des Grands Modèles de Langage peuvent être difficiles, chronophages et nécessiter des ressources considérables en matière de matériel et d'expertise technique. Souvent, il est essentiel d'avoir des compétences en Deep Learning, des infrastructures distribuées et une compréhension approfondie du fonctionnement deces modèles pour les mettre en pratique. [12]

I.10 Conclusion

Dans Ce chapitre nous avons donné une vue générale sur les grands modèles de langage (LLMs), De ce fait, nous avons présenté les aspects couvrant les éléments clés de leur conception, de leur fonctionnement et de leurs performances. Nous avons examiné les principales familles de LLMs, les composants fondamentaux de ces modèles, ainsi que leurs architectures. Nous avons mis en évidence leur diversité et leur impact sur le domaine du traitement du langage naturel. En comprenant le fonctionnement des LLMs, nous avons clarifié la manière dont ces modèles traitent et génèrent du langage. Enfin, en abordant les limites et les défis de la performance des LLMs dans des contextes spécifiques.

Chapitre

2

Prompt Engineering

II.1 Introduction

L'ingénierie de prompt émerge comme une discipline clé visant à développer et optimiser des instructions pour améliorer l'efficacité des modèles de langage dans une variété d'applications et de champs de recherche. Cette expertise permet une meilleure compréhension des capacités et des limites des grands modèles de langage, permettant aux chercheurs d'améliorer leurs performances dans divers domaines, allant de la résolution de questions simples à des raisonnements arithmétiques complexes. Les développeurs, quant à eux, utilisent cette ingénierie de prompt pour concevoir des techniques interactives et efficaces, favorisant ainsi une interaction optimale avec les modèles de langage et autres outils.

Ce chapitre met en évidence les fondements du prompt engineering, en présentant les principaux composants d'un prompt, les divers types de prompts, ainsi que les processus et techniques d'ingénierie des prompts. Nous avons également examiné les avantages et les défis associés au domaine du prompt engineering.

II.2 Importance de Prompt engineering pour LLMs

L'ingénierie de prompts est un processus crucial dans la conception et l'amélioration des requêtes d'entrée pour obtenir des réponses souhaitées des grands modèles de langage. Elle permet d'optimiser les performances des modèles, d'explorer de nouvelles applications et de gagner du temps et des ressources. En comprenant comment formuler des prompts efficaces, les chercheurs et les développeurs peuvent débloquer le plein potentiel des modèles de langage. Cette pratique offre également un cadre pour structurer les prompts, les adapter à différents domaines et améliorer les sorties des modèles. Elle soutient ainsi le développement d'applications plus sophistiquées, facilite la compréhension des capacités des modèles et contribue à orienter les modèles vers des réponses plus véridiques et informatives. L'ingénierie de prompts devrait jouer un rôle croissant dans l'exploitation des capacités des grands modèles de langage, offrant des perspectives prometteuses pour l'efficacité commerciale et le développement technologique.

II.3 Définition d'un Prompt Engineering

Un prompt est une instruction claire et contextualisée utilisée pour guider la sortie d'un modèle de langage, tel que ChatGPT *(Voir Figure II.1)*. Ces instructions transmettent l'intention de l'utilisateur et spécifient le contexte dans lequel le modèle doit opérer, ainsi que le type de réponse attendue. L'ingénierie de prompt consiste à concevoir et optimiser ces instructions pour obtenir des résultats pertinents et adaptés aux besoins spécifiques de l'utilisateur, en tenant compte des données d'entraînement du modèle et de sa structure architecturale.

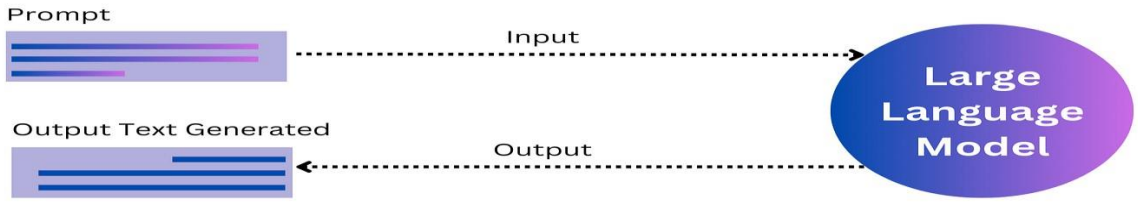

Figure II. 1: Prompt Engineering [13]

II.4 Éléments d'un prompt

Les Prompts peuvent contenir l'un des éléments suivants :

- **Instructions :** une tâche ou instructions spécifiques que vous souhaitez que le modèle exécute.
- **Contexte :** peut impliquer des informations externes ou tout autre contexte pouvant guider lemodèle vers de meilleures réponses.
- **Données d'entrée :** sont les entrées ou les questions auxquelles nous voulons trouver desréponses.
- **Indicateur de sortie :** Indique le type ou le format de la sortie. [14]

Tous les composants ne sont pas requis pour les invites, le format dépend de la tâche àaccomplir. Nous illustrerons des exemples du tableau suivant :

Prompt	Instruction	Input data	Contexte	Output indicator
Peindre un grand homme africain dansant sur de chouettes chansons folkloriques ougandaises	Peindre	NA (Non Applicable)	Homme africain de grande taille dansant sur des chansons folkloriques ougandaises cool	NA
Écrire une nouvelle de science-fiction se déroulant dans un futur où les humains, en particulier les bébés, ont colonisé Mars, en explorant les implications éthiques de la terraformation et les conflits potentiels entre les sociétés terriennes et martiennes	Écrire une nouvelle de science-fiction	Bébés	Se déroulant dans un futur où les humains ont colonisé Mars	Explorer les implications éthiques de la terraformation et les conflits potentiels entre les sociétés terriennes et martiennes
Écrivez un court paragraphe expliquant le théorème de Pythagore et fournissant un exemple pour démontrer son utilisation	Écrivez un court paragraphe	NA (Non Applicable)	Expliquer un concept mathématique	Fournir un exemple
Générer un rapport technique sur la faisabilité d'utiliser la fusion nucléaire comme source d'énergie durable, incluant une analyse des limitations technologiques actuelles et des solutions potentielles.	Générer un rapport technique	NA (Non Applicable)	Faisabilité d'utiliser la fusion nucléaire comme source d'énergie durable	incluant une analyse des limitations technologiques actuelles et des solutions potentielles.

Tableau II. 1: Exemples de prompts pour les grands modèles de langage [15]

II.5 Le Processus d'ingénierie de Prompt

Étapes de création de promptes efficaces :

II.5.1 Définition de l'objectif

Avant de commencer l'ingénierie des prompts, vous devez comprendre clairement votre objectif final. Quel problème essayez-vous de résoudre ? Quel type d'information avez- vous besoin ? Définir votre objectif vous aidera à concentrer vos questions et à générer des réponses plus précises et utiles. [16]

II.5.2 Compréhension des capacités du modèle

Pour chaque modèle de langage sélectionné pour le prompt, il est très important de comprendre ses limitations et ses capacités. Par exemple, il est important de savoir quel type de réponses le modèle génère (texte, audio, images, etc.). Les forces et les faiblesses d'un modèle donné guident la synthèse de prompts appropriés qui sont compatibles avec les capacités du modèle et qui contournent ces limitations. [15]

II.5.3 Choix du bon format de prompt

Le format du prompt a un impact énorme sur la qualité des réponses générées par les LLM. Les formats de prompts clairs, précis et concis fournissent au modèle tous les détails nécessaires pour générer des réponses cohérentes. La sélection de formats de prompts appropriés améliore la compréhension du langage naturel pour les LLM, ce qui améliore la qualité de la réponse donnée. [15]

II.5.4 Fournir du contexte

Le contexte est crucial pour générer des réponses précises. Assurez-vous de fournir suffisamment d'informations dans vos questions pour donner au modèle une compréhension claire de la situation. Par exemple, si vous demandez des informations sur la météo, fournissez l'emplacement et l'heure de la journée pour donner au modèle une meilleure compréhension des conditions. [16]

II.5.5 Test et Raffinement

L'ingénierie de la requête est un processus itératif, il est donc essentiel de tester vos questions et de les affiner en fonction des résultats que vous recevez.

Essayez différentes formulations et variations de vos questions pour voir lesquelles génèrent les réponses les plus précises et utiles. [16]

II.6 Les différents types de prompt engineering

L'ingénierie de prompts peut être abordée de différentes manières selon l'objectif visé. Voici lesgrands types de prompt engineering *(Voir Figure II.2)* :

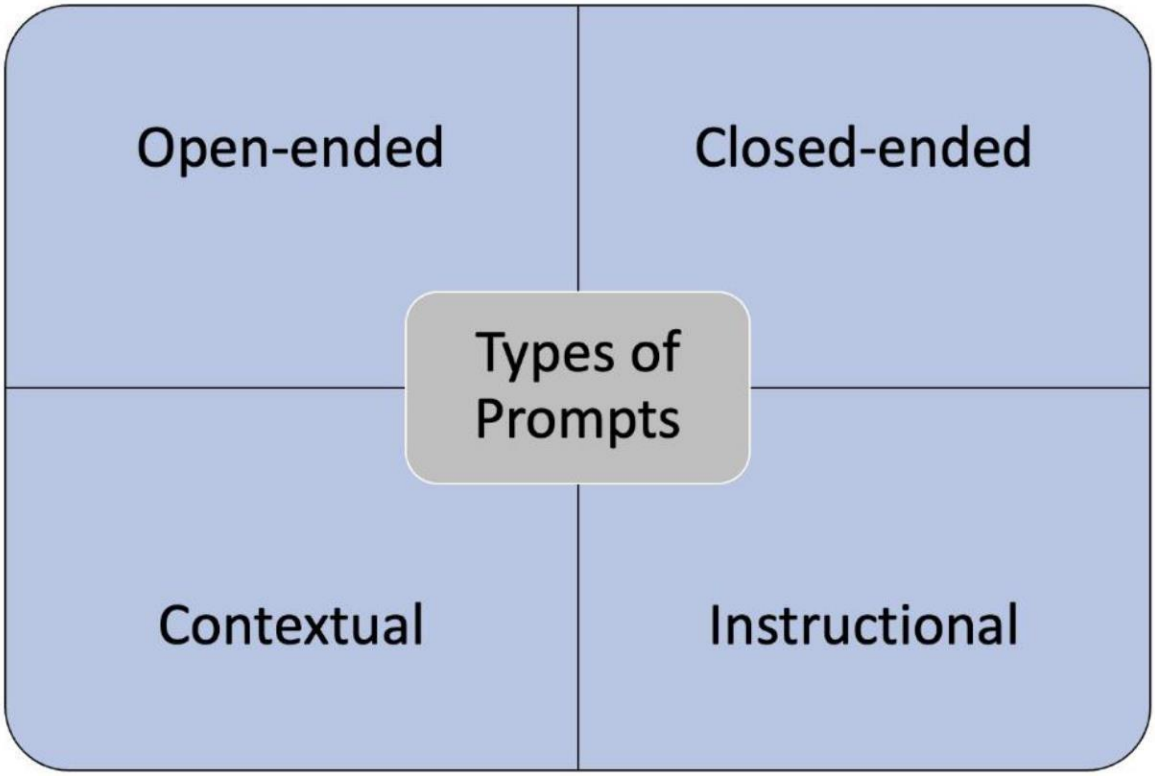

Figure II. 2: Les Types de Prompt Engineering [17]

- **Prompts ouverts (open-ended prompts)** : Les prompts ouverts sont essentiellement des questions ou des énoncés envoyés aux modèles d'IA sans contraintes spécifiques. Les modèles ont la liberté de générer des réponses sans limitations, et il est possible que les résultats générés ne soient pas toujours pertinents sur le plan contextuel. De plus, la longueur de la réponse est déterminée uniquement par le modèle, ce qui signifie qu'elle peut aller de concise à détaillée. Exemple d'un prompt ouvert : "Racontez-moi une histoiresur un célèbre roi qui a gouverné Rome." Dans cet exemple, le modèle d'IA a la

liberté de générer n'importe quelle histoire sur un roi renommé qui a gouverné Rome. La réponse peut varier largement en fonction de l'interprétation et de la créativité du modèle, entraînant des résultats divers et imprévisibles à chaque fois que le prompt est présenté au modèle. [17]

- **Prompts fermés (closed-ended prompts)**: Les prompts fermés imposent des contraintes aux modèles d'IA pour fournir des réponses strictement dans le contexte donné. Les modèles doivent générer des résultats qui abordent directement le sujet spécifique et fournissent des réponses directes. Par exemple, considérons le prompt fermé **:** "Quel est lerésultat attendu du modèle est "trois". Dans cet exemple, le résultat est spécifique au sujet,dans le contexte donné, et est en accord avec les contraintes fournies. C'est une réponse claire et directe à la question posée. [17]

- **Prompts instructifs** : Les prompts instructifs donnent des instructions claires aux LLMs pour fournir des réponses dans un sujet et un contexte spécifiques. Ce sont des prompts fermés, ce qui signifie qu'ils ont des contraintes définies. Par exemple, considérons le prompt **:** "Listez les 10 meilleurs médaillés d'or des Jeux olympiques d'été de 2020"**.** Ce prompt est entièrement défini, spécifique au sujet des Jeux olympiques d'été de 2020, et attend que le modèle liste les noms des 10 meilleurs médaillés d'or dans ce contexte. C'est une instruction claire pour le modèle à suivre, ce qui entraîne une réponse spécifique et ciblée. [17]

- **Prompts Contextuels** Les prompts contextuels enrichissent les modèles linguistiques en leur fournissant des informations contextuelles spécifiques pour générer des réponses adaptées à un contexte donné .Par exemple, considérons le prompt contextuel : "À quelle heure le vol AB12, en partance de New York à destination de la Floride, décolle-t-il de l'aéroport JFK demain matin ? " **;** Dans cet exemple, le contexte fourni contient des informations telles que le numéro de vol (AB12), l'itinéraire (de New York à la Floride), l'aéroport de départ (JFK) et l'heure (demain matin). Ces détails contraignent la réponse du modèle Al à rester dans le contexte donné, soulignant ainsi l'importance des prompts contextuels pour garantir des réponses pertinentes et précises. [17]

II.7 Techniques d'ingénierie de prompt

Les techniques d'ingénierie de prompt comprennent plusieurs approches *(Voir Figure II.3)* :

II.7.1 Le promptage zéro-shot : Il ne fournit aucun exemple et laisse le modèle se débrouiller seul. Il repose uniquement sur les données de pré-entraînement du modèle et sur les techniques d'entraînement pour générer une réponse. La réponse peut ne pas être parfaitement exacte, mais elle sera probablement cohérente. Exemple " Dans cette phrase incomplète **:** "Un trianglea trois….. ", une réponse que le modèle pourrait donner, même sans une expertise en géométrie, est : "Un triangle possède trois côtés".[18]

II.7.2 Le promptage few-shot : est une approche qui consiste à fournir au modèle de langage quelques exemples de la tâche à réaliser, appelés "shots". Ces exemples fournissent un cadre de référence pour diriger le modèle vers le format de réponse anticipé. Cette méthode permet de contextualiser la tâche pour le modèle, elle permet de mieux comprendre ce qui est attendu de lui. Elle conduit souvent à des réponses plus pertinentes et de meilleure qualité. [19]

II.7.3 Le promptage en chaîne : Ce type de prompt guide le modèle dans un raisonnement étape par étape. Il aide le modèle à décomposer le problème en une série de sous-étapes logiques. Cette approche s'avère utile pour résoudre des problèmes complexes qui demandent une explication détaillée du processus de réflexion. En suivant ce raisonnement, on obtient une meilleure compréhension de la réponse générée. [20]

II.7.4 Le promptage de cohérence interne : pousse davantage en demandant au modèle de produire plusieurs réponses ou raisonnements pour une question donnée, puis en cherchant à les harmoniser. Cette approche est particulièrement utile pour les questions comportant diverses approches possibles, visant ainsi à accroître la fiabilité et la précision des réponses en repérant les convergences entre les différentes solutions proposées. [20]

II.7.5 Le promptage de connaissances : généralement consiste simplement à demander au modèle d'IA une large gamme de questions ouvertes sans aucun contexte prédéfini. Le modèle d'IA vous fournit les réponses en se basant sur le modèle pré-entraîné à partir de milliards de points de données. [21]

II.7.6 Le promptage arborescent (ToT) : des pensées est similaire au promptage de cohérence interne, où la réponse reçue d'une question est transmise comme contexte à la question suivante. Par conséquent, on s'attend à ce que le modèle construise les réponses en se basant sur les réponses reçues à chaque étape et fournisse une réponse intégrée de manière logique. [17]

II.7.7 L'Ingénierie de Prompt Automatique : est une méthode avancée où un modèle de langage développe et choisit ses propres instructions de manière autonome. Elle utilise des algorithmes d'apprentissage automatique pour optimiser et évaluer les solutions proposées, permettant ainsi à l'IA de diviser des tâches et de générer des directives de suivi avec un minimum d'intervention humaine. [22]

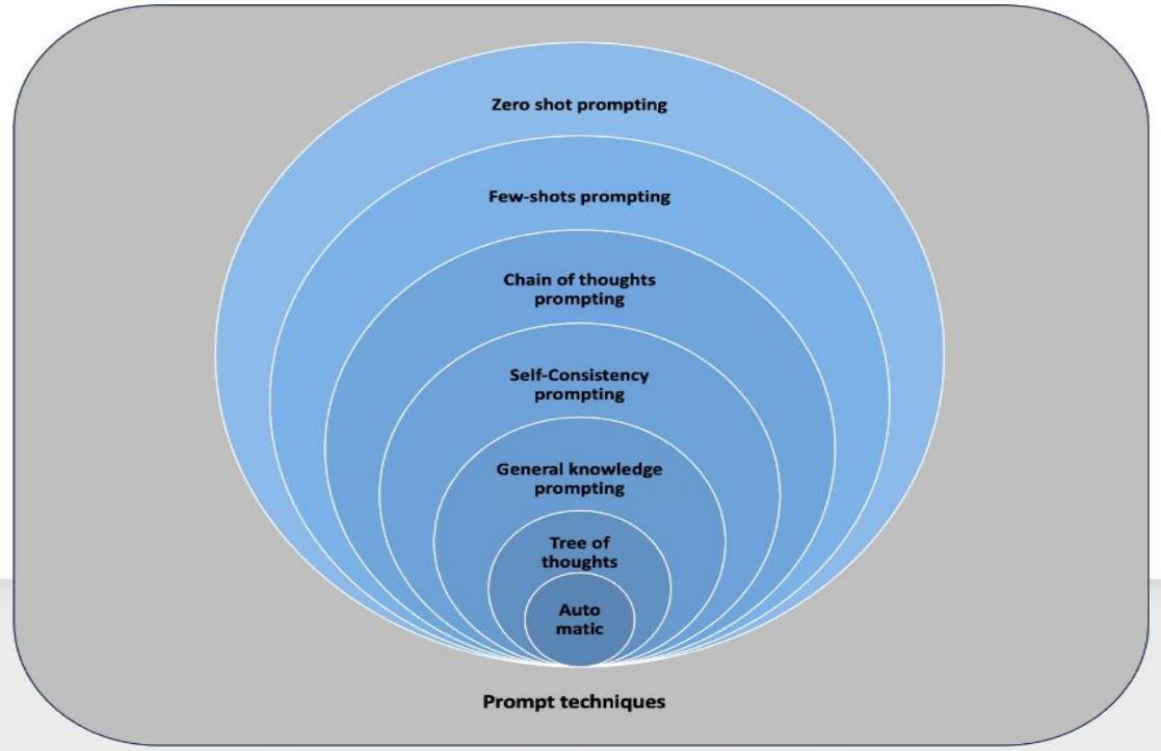

Figure II. 3: Les Différents Technique de Prompt Engineering [17]

II.8 Application d'ingénierie de Prompts

L'ingénierie de prompts est utilisée dans divers domaines de l'intelligence artificielle pour fournir des réponses précises et pertinentes aux utilisateurs. Voici les exemples essentiels :

- **Traitement du langage naturel :** Résumé de texte et traduction dans un langage accessible.
- **Chatbots et assistants virtuels :** Répondre aux questions spécifiques des utilisateurs demanière conversationnelle.
- **Génération de contenu :** Création de contenu écrit, y compris des réponses courtes oulongues et des histoires fictives.
- **Systèmes de question-réponse :** Aider les utilisateurs à obtenir des réponses précises à leursquestions.
- **Analyse de données :** Analyse des ensembles de données pour fournir des insights pour laprise de décision.
- **Extraction d'informations :** Obtenir un résumé rapide d'un article ou d'une vidéo en extrayant les informations clés.

Ces exemples démontrent la polyvalence de l'ingénierie de prompts pour répondre aux besoinsdes utilisateurs dans différents contextes. [23]

II.9 Avantages du Prompt Engineering

- Fondamental pour la recherche, les découvertes et les progrès.
- Contribue à l'exploration et à l'évaluation des capacités des modèles de langage (LLM).
- Ouvre la voie à une multitude d'applications innovantes tirant parti des LLM.
- Favorise une meilleure compréhension des comportements et des limites des LLM dansdivers contextes.
- Facilite l'adaptation des modèles de langage à des tâches spécifiques et à des languesmoins représentées. [24]

II.10 Défis du Prompt Engineering

- **Investissement en temps et compétences :** Le prompt engineering nécessite un investissement en temps important et des compétences solides en conception de prompts pour élaborer des instructions optimales.
- **Dépendance au cas d'usage :** Les prompts spécifiques à une tâche doivent être entièrement réécrits en cas de changement d'objectif ou de contexte, ce qui limite leur réutilisabilité.
- **Risque de réduction de la créativité :** Des prompts trop directifs ou détaillés peuvent réduire la créativité et l'autonomie des modèles NLP en les enfermant dans des schémasde réponse prédéfinis et stéréotypés.
- **Essais-erreurs et ajustements nécessaires :** Le processus de création de promptes efficaces implique beaucoup d'essais et d'erreurs, ainsi que des ajustements continus et des réglages précis pour atteindre un prompt optimal.
- **Complexité croissante :** La complexité des prompts tend à croître à mesure qu'on cherche à les optimiser, ce qui peut potentiellement devenir contre-productif s'ils sont surchargés d'instructions. [25]

II.11 Conclusion

En conclusion, l'ingénierie des prompts représente une avancée capitale dans le domaine Des grands modèles de langage et des systèmes d'IA conversationnelle. Son potentiel est immense, offrant des performances impressionnantes et ouvrant la voie à de nouvelles opportunités passionnantes. En restant attentifs aux développements et en investissant dans cette discipline, nous pouvons façonner un avenir où les interactions avec l'IA sont plus efficaces, plus personnalisées et plus gratifiantes pour tous.

Chapitre

3

Conception et Implémentation

III.1 Introduction

L'essor des technologies de traitement du langage naturel (NLP) a considérablement transformé la manière dont les interactions entre les utilisateurs et les bases de données sont conçues. Une des avancées notables dans ce domaine est l'approche Texte-à-SQL, qui permet de traduire des requêtes en langage naturel en instructions SQL exploitables par un système de gestion de bases de données (SGBD). Cette technique est essentielle pour rendre les systèmes de base de données plus accessibles à des utilisateurs non techniques, en leur permettant d'extraire et de manipuler des données sans nécessiter une connaissance approfondie du langage SQL.

Ce chapitre se concentre sur l'élaboration et la mise en œuvre d'une approche Texte-à-SQL, en détaillant les différentes composantes et technologies utilisées dans le processus de développement. Nous examinerons d'abord les principes fondamentaux de l'architecture de notre Application, de conversion Texte en SQL, avant de plonger dans l'environnement de développement et les outils utilisés dans notre Implémentation.

III.2 Approche Texte-à-SQL

III.2.1 Motivation et Principe

L'approche Texte-à-SQL comble le fossé entre le langage naturel et les bases de données relationnelles. Elle permet aux utilisateurs d'interagir avec les bases de données en posant des questions en français courant, éliminant ainsi la nécessité d'écrire directement du code SQL (Structured Query Language) complexe. [26]

Cette méthode révolutionnaire représente un pas en avant majeur dans la manière dont les individus interagissent avec les bases de données. Auparavant, seuls les programmeurs et les analystes de données familiarisés avec le SQL pouvaient extraire des informations des bases de données. Maintenant, grâce à l'approche Texte-à-SQL, cette capacité est étendue à un public beaucoup plus large, notamment les gestionnaires, les commerciaux et d'autres professionnels qui peuvent ne pas avoir des connaissances techniques approfondies.

L’adoption de l’approche Texte-à-SQL présente plusieurs avantages significatifs :

- elle facilite l’accès aux données pour des utilisateurs non techniques, en leur permettant de formuler des requêtes complexes en langage naturel. Cela réduit la dépendance à l’égard des experts en bases de données et accélère le processus de prise de décision. Par exemple, un responsable des ventes peut demander instantanément des données sur les ventes mensuelles par région sans avoir à attendre le support d’un analyste de données.
- De plus, cette approche améliore l’efficacité et l’accessibilité de l’interaction avec les bases de données. Plutôt que de devoir apprendre le SQL, les utilisateurs peuvent simplement poser des questions dans un langage familier, ce qui réduit les obstacles à l’entrée et favorise une adoption plus large des systèmes de gestion de bases de données.

Cependant, il convient de noter que l’approche Texte-à-SQL n’est pas sans ses défis. La conversion précise du langage naturel en requêtes SQL peut être complexe, en particulier pour les questions ambiguës ou mal formulées. De plus, les systèmes Texte-à-SQL peuvent avoir du mal à traiter les requêtes complexes ou les bases de données massives avec des schémas complexes.

III.2.2 Architecture de L’approche Texte-à-SQL Utilisé

L'approche Texte-à-SQL que nous avons adoptée s'appuie sur les capacités des modèles de langage génératifs (LLMs) tels que GPT, à comprendre et à traiter d’une manière très efficace le langage naturel. La figure suivante *(Voir Figure III.1)* résume l’architecture principale de notre approche.

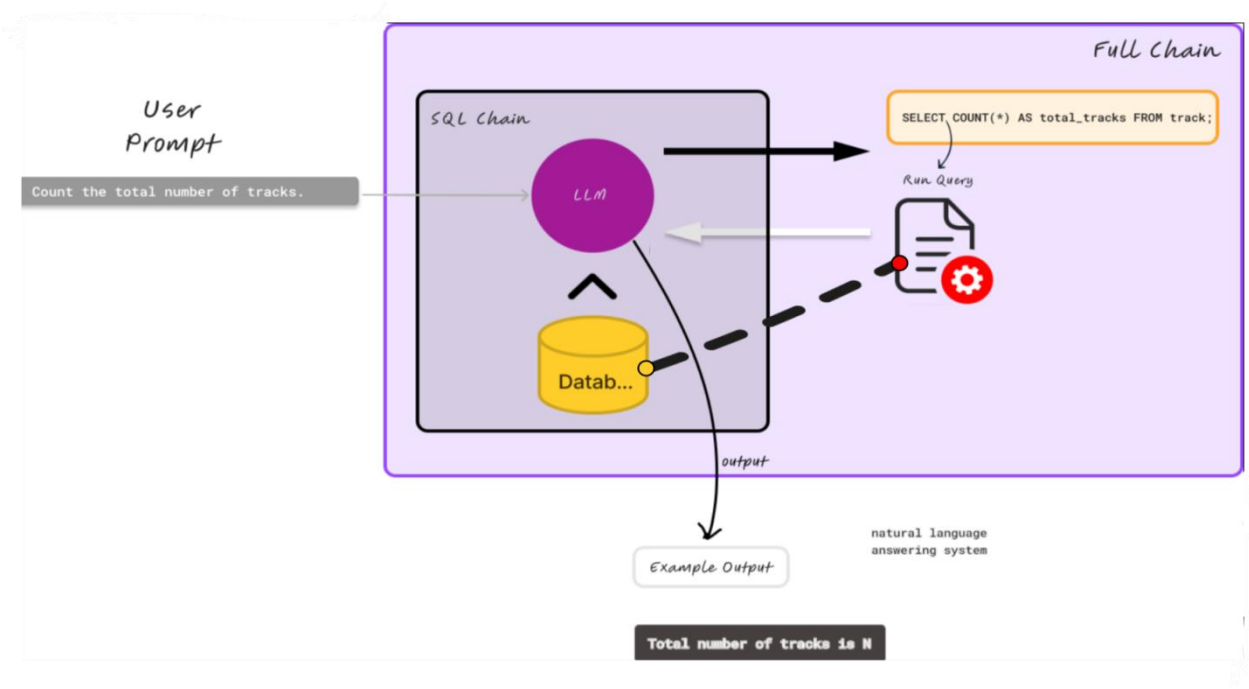

Figure III. 1: Schémas de Fonctionnement

Le processus derrière l'architecture de la figure précédente, inclut les étapes suivantes :

1. **Entrée de la requête en langage naturel (user prompt)** : L'utilisateur pose une question en langage naturel via l'interface utilisateur.
2. **Génération de la requête SQL (SQL chain)** : la question précédente est envoyée à la chaîne de traitement des requêtes où le modèle de langage génère une requête SQL correspondante.
3. **Exécution de la requête SQL** : La requête SQL générée est envoyée au module de gestion de la base de données, qui l'exécute sur la base de données locale.
4. **Retour des résultats** : Le résultat de la requête est renvoyé à la chaîne de traitement des requêtes, qui formate la réponse en langage naturel et la renvoie à l'interface utilisateur. Ce dernier affiche le résultat final à l'utilisateur.

Nous avons appelé « Full chain » les étapes : Génération de la requête SQL, Exécution de la requête SQL et le retour des résultats.

III.3 Environnement de Développement

III.3.1 Outils de Développement et Bibliothèque

- **Langage de Développement :**

  Python : a été choisi pour son large éventail de bibliothèques et son efficacité dans le développement d'applications de traitement du langage naturel. Sa syntaxe claire et sa facilité d'apprentissage en font un choix idéal pour ce projet. De plus, la communauté Python est très active, ce qui signifie qu'il existe de nombreuses ressources et bibliothèques disponibles pour répondre aux besoins du projet « python.org ». [27]

- **API et Bibliothèque :**

  - dotenv : Utilisé pour charger les variables d'environnement à partir d'un fichier .env. par exemple, l'utilisation des services externes comme OpenAI ou GROQ, nécessite le besoin de clés d'API pour y accéder. Nous avons utilisé la bibliothèque **python-dotenv** pour charger ces variables d'environnement à partir du fichier **.env**. [28]

  - streamlit : C'est une bibliothèque Python pour la création d'applications web interactives avec Python. Elle est utilisée ici pour crée interface utilisateur. [29]

  - LangChain : Une bibliothèque pour faciliter la création et la gestion des chaînes de traitement des modèles de langage. Elle nous permet de structurer les prompts et de gérer les interactions avec les modèles de langage. [30]

  - langchain_openai : L'intégration des modèles GPT d'OpenAI via cette bibliothèque a permis d'implémenter des fonctionnalités avancées telles que la génération de réponses basées sur les requêtes de l'utilisateur.

  - Clés d'API : On a utilisé des services externes comme OpenAI ou GROQ, on a besoin de clés d'API pour y accéder. On a utilisé la bibliothèque **python-dotenv** pour charger ces variables d'environnement à partir du fichier **.env**.

- SQLDatabase : Un utilitaire pour faciliter les connexions et les interactions avec les bases de données SQL. [31]
- SQLite : est une base de données simple et légère qui est facile à configurer et à utiliser. Elle est très utile à des fins de test et de développement. Pour le télécharger. [32]
- MySQL : est un système de gestion de base de données relationnelle open-source populaire. Il est largement utilisé dans le développement web et connu pour sa rapidité et sa fiabilité. Pour installer MySQL sur votre machine. [33]

III.3.2 Implementation

Dans cette section, nous détaillerons l'implémentation de notre système en mettant en lumière les éléments clés tels que le LLM utilisé, la base de données, ainsi que d'autres détails pertinents. *(Voir Figure III.2)*

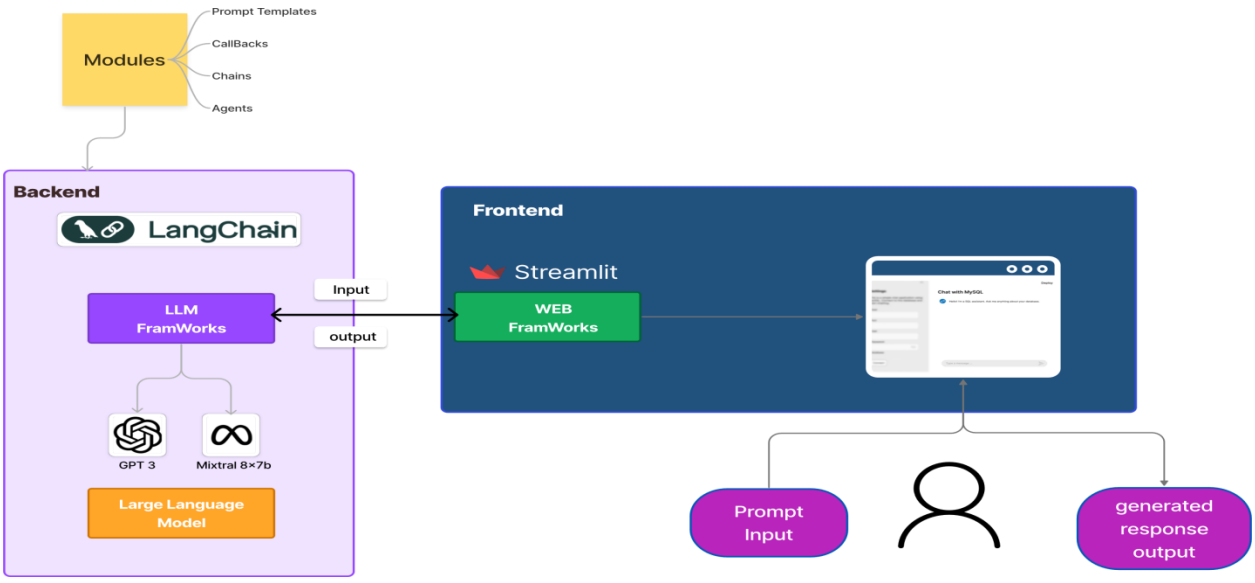

Figure III. 2: Détails de L'implémentation du Système

- Selon le schéma, notre application se divise en deux parties : le côté « Backend » et le côté « Frontend ». Le côté Frontend gère tout ce qui concerne l'interface et la communication avec l'utilisateur. Le côté Backend, prend en charge l'aspect fonctionnel de notre application. Il implémente les modules responsables des fonctions principales, à savoir la communication avec le LLM, la construction des prompts, ainsi que la récupération des résultats des opérations de conversion Texte-à-SQL et SQL-à-Texte.

La suite de cette section décrit en premier lieu les modèles et la base de données utilisés dans notre implémentation, avant de détailler l'implémentation de certaines fonctionnalités de l'application, telles que la création des prompts et l'interaction avec les modèles LLM.

- Enfin, nous présentons brièvement les principales fonctionnalités de l'application.

III.3.2.1 Les Modèles LLM utilisés

Pour le traitement du langage naturel, nous avons utilisé deux modèles : le modèle **"gpt-3.5-turbo"** d'Open AI et le modèle **"mixtral-8x7b-32768"** de Meta.

- **gpt-3.5-turbo :** Ce modèle d'OpenAI est une version améliorée de GPT-3. En utilisant ce modèle, nous bénéficions de la capacité de GPT-3.5-turbo à comprendre et à générer du texte de manière fluide et naturelle.
- **mixtral-8x7b-32768 :** le modèle "mixtral-8x7b-32768" est conçu pour traiter des données et fournir des réponses en langage naturel avec une grande précision. Le paramètre de température a été défini à 0 lors de l'initialisation de ce modèle, garantissant des réponses cohérente et précise en minimisant les variations.

La combinaison de ces deux modèles permet d'assurer une expérience utilisateur de haute qualité.

III.3.2.2 La Base de Données Utilisée

Dans notre application nous avons utilisé la base de données relationnelle **Chinook** *(Voir Figure III.3)***,** est une base de données contient des informations sur un magasin de musique, Elle fournit des informations sur les médias disponibles dans le magasin, ainsi que des détails sur les artistes,

les albums, les clients, les employés et les transactions de facturation. Elle représente un standard souvent utilisée dans le cadre de démonstrations et de tests pour divers systèmes de gestion de bases de données, tels que SQL Server, Oracle et MySQL. Le motif principal derrière le choix de la base Chinook est la disponibilité des requêtes en langage naturel et de leurs correspondantes en SQL. Cela facilite son utilisation comme benchmark pour tester l'efficacité de notre implémentation.

Elle Comprend : [34]

- 11 tables
- Une variété d'index, de contraintes de clé primaire et étrangère
- Plus de 15 000 lignes de données

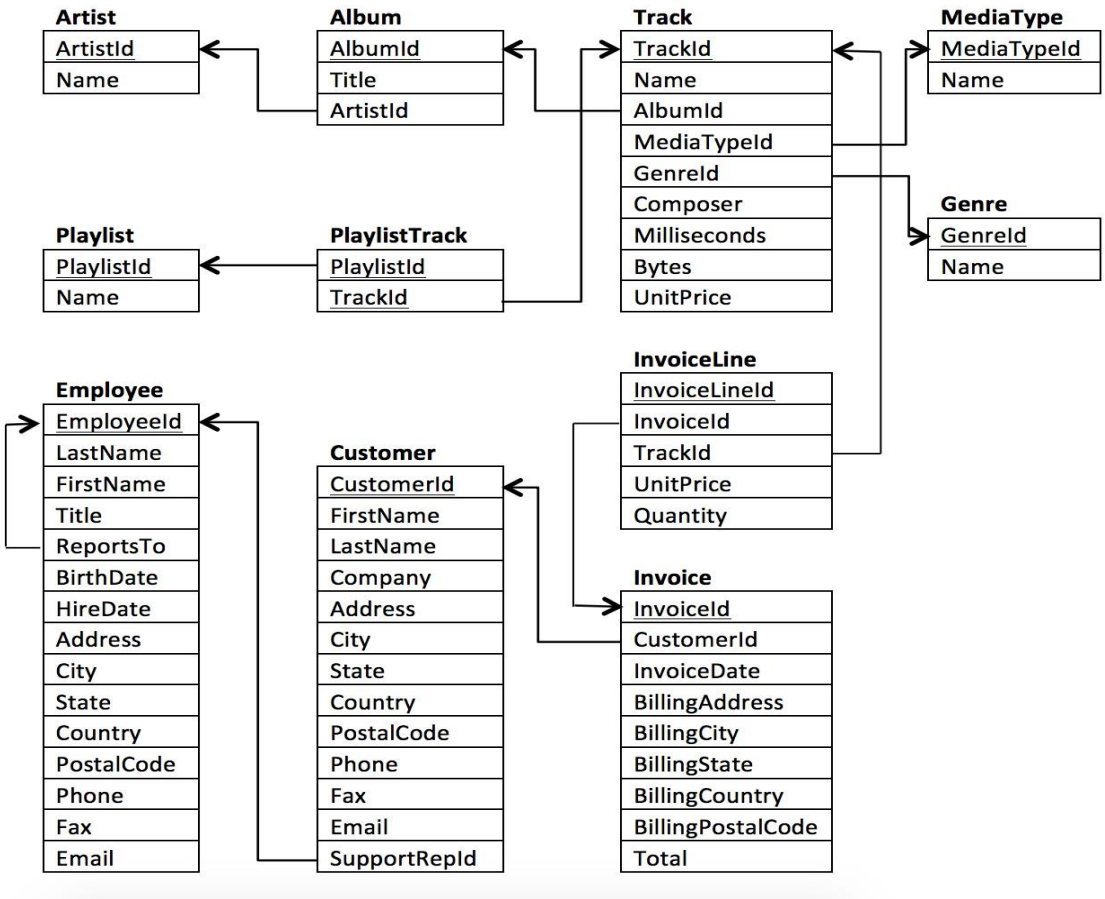

Figure III. 3: Schémas de la Base de Données Chinook [35]

III.3.2.3 Interaction avec les Modèles de Langage

Pour générer les requêtes SQL et les réponses en langage naturel, nous utilisons deux modèles de langage : GPT-3.5 de OpenAI et Mixtral de Meta. Chaque modèle est invoqué à travers une chaîne de traitement définie dans LangChain. Cette chaîne combine les prompts et les modèles de langage pour produire des réponses précises et pertinentes. La température du modèle est réglée à 0 pour obtenir des réponses déterministes et cohérentes.

III.3.2.4 Création de Prompt

Les prompts sont essentiels pour structurer les interactions avec les modèles de langage. Dans notre implémentation, Nous avons conçu un prompt dynamique qui incite l'utilisateur à poser des questions sur la base de données Chinook et à fournir des réponses spécifiques en SQL. On a utilisés deux types de prompts principaux :

1. Prompt pour la génération des requêtes SQL:

- Ce prompt est conçu pour interpréter les questions de l'utilisateur en tenant compte du schéma de la base de données et de l'historique des conversations. Il génère une requête SQL précise qui peut être exécutée sur la base de données.

2. Prompt pour la génération des réponses en langage naturel :

- Ce prompt prend en entrée la question de l'utilisateur, la requête SQL générée et les résultats de la requête SQL pour créer une réponse en langage naturel. Cela permet de transformer des résultats SQL bruts en une réponse compréhensible pour l'utilisateur.

III.3.2.1.1 Etapes Principales de l’implémentation

Dans cette section, nous allons examiner de plus près les principales étapes de l’implémentation de notre Application « Chat with MySQL » en mettant en évidence les différentes composantes et les étapes clés du processus.

- Initialisation de la base de données : La fonction init_database permet d'initialiser et de se connecter à la base de données MySQL. Nous fournissons les informations de connexion telles que l'utilisateur, le mot de passe, l'hôte, le port et le nom de la base de données pour établir la connexion.

- Configuration du pipeline de traitement : Nous avons mis en place un pipeline de traitement robuste pour gérer les requêtes utilisateur et générer des réponses appropriées. Cela implique l'utilisation des classes et des méthodes fournies par les bibliothèques

- langchain_core, langchain_community, et langchain_groq. Ce pipeline comprend des étapes telles que la définition des invitations de chat, la manipulation des messages, l'exécution des requêtes SQL, et la génération de réponses naturelles en fonction du contexte de la conversation.

- Interface utilisateur avec Streamlit : Pour créer une interface utilisateur interactive, conviviale et intuitive, nous utilisons la bibliothèque streamlit. Cela nous permet d'afficher les messages de chat, de saisir les requêtes utilisateur, et d'afficher les réponses générées par le système de manière claire et structurée.

- Gestion de l'historique des conversations : Nous maintenons un historique des conversations dans la session pour suivre le contexte des interactions précédentes entre l'utilisateur et le système. Cela nous aide à adapter les réponses en fonction du contexte et à fournir des réponses plus pertinentes.

III.3.2.2 Présentation de l'application

Notre application est un assistant interactif de requêtes SQL conçu pour simplifier l'interaction avec les bases de données MySQL. Voici un aperçu des fonctionnalités principales et les résultats que nous pouvons obtenir :

Connexion à la Base de Données : L'application permet aux utilisateurs de se connecter à une base de données MySQL en spécifiant les détails de connexion tels que l'hôte, le port, le nom d'utilisateur, le mot de passe et le nom de la base de données *(Voir Figure III.4)*. Une fois

connecté, l'application affiche un message de confirmation de la connexion réussie.

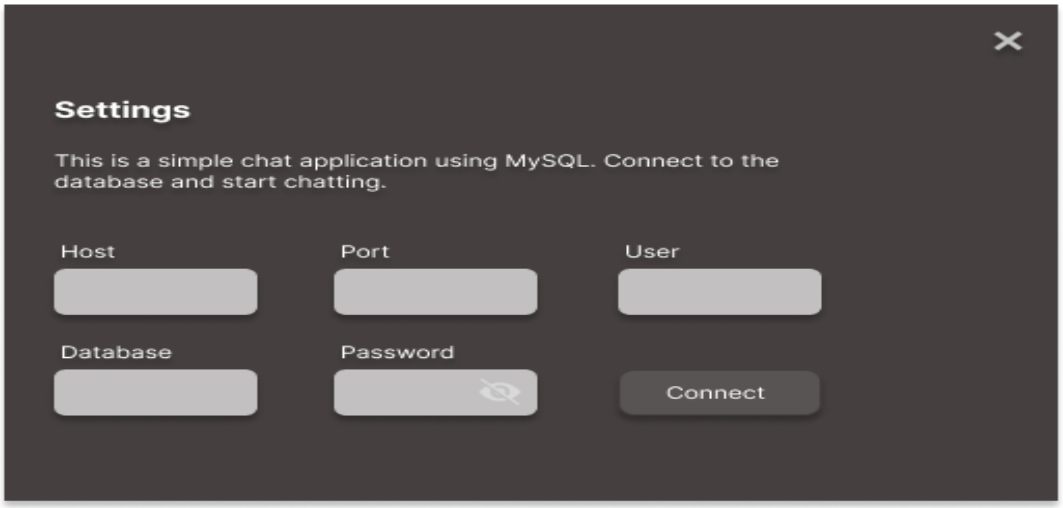

Figure III. 4: Connexion à la base

- **Interface de Chat interactif :** Une fois connecté à la base de données *(Voir Figure III.5)*, les utilisateurs peuvent poser des questions en langage naturel à l'assistant de requêtes SQL. L'interface de chat permet une communication fluide où les utilisateurs peuvent saisir leurs requêtes dans la zone de texte et recevoir des réponses immédiates de l'assistant.

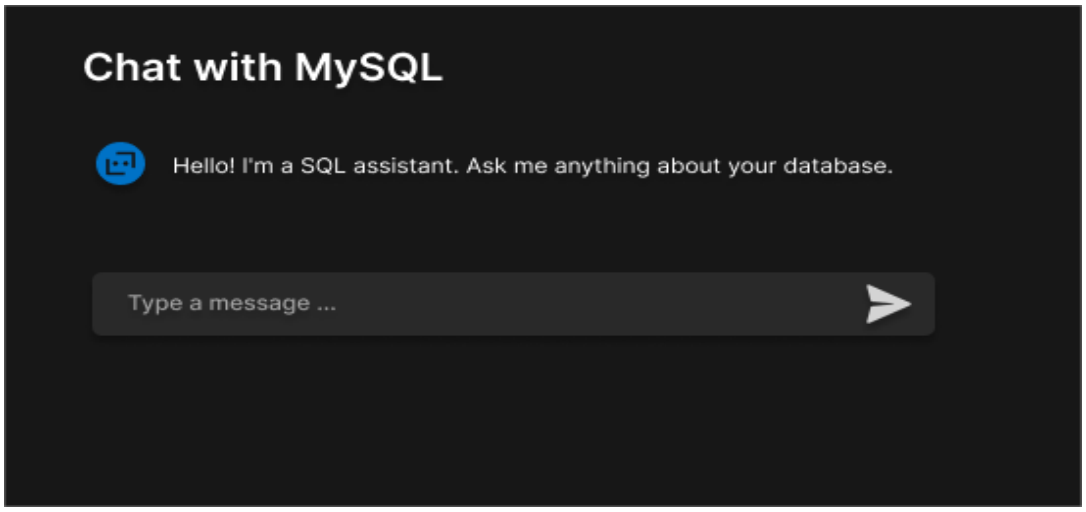

Figure III. 5: Connexion réussite à la base

- **Génération de Requête SQL et affichage de Résultat :** L'assistant analyse les questions des utilisateurs et génère automatiquement des requêtes SQL correspondantes en fonction du schéma de la base de données et de l'historique de la conversation. Les requêtes SQL générées sont affichées dans l'interface de chat pour examen par l'utilisateur. Une fois que l'utilisateur valide une requête SQL générée, l'assistant exécute la requête sur la base de données et affiche les résultats correspondants dans l'interface de chat. Les résultats sont présentés de manière claire et organisée pour une meilleure compréhension *(Voir Figure III.6, Figure III.7).*

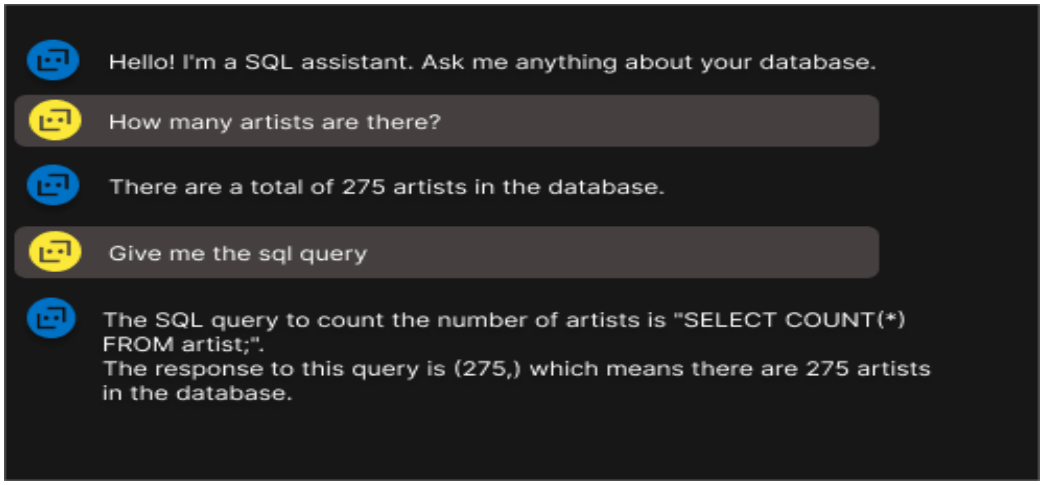

Figure III. 6: Résultat pour le Modèle Chat OpenAI

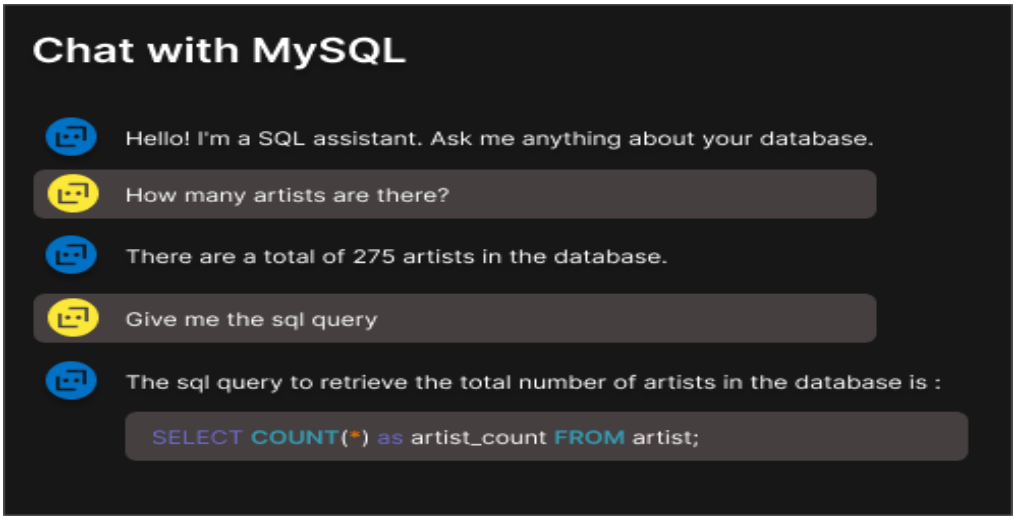

Figure III. 7: Résultat pour le Modèle Mixtral

L'application conserve un historique de la conversation entre l'utilisateur et l'assistant, ce qui permet de suivre les requêtes posées et les réponses fournies. L'historique de la conversation est affiché dans l'interface de chat pour référence future.

III.4 Résultats et Discussion

III.4.1 Méthode de Test

Pour tester l'efficacité de notre implémentation, nous avons effectué un ensemble de tests à l'aide du benchmark de la base de données Chinook. Le schéma suivant résume la méthode de notre test :

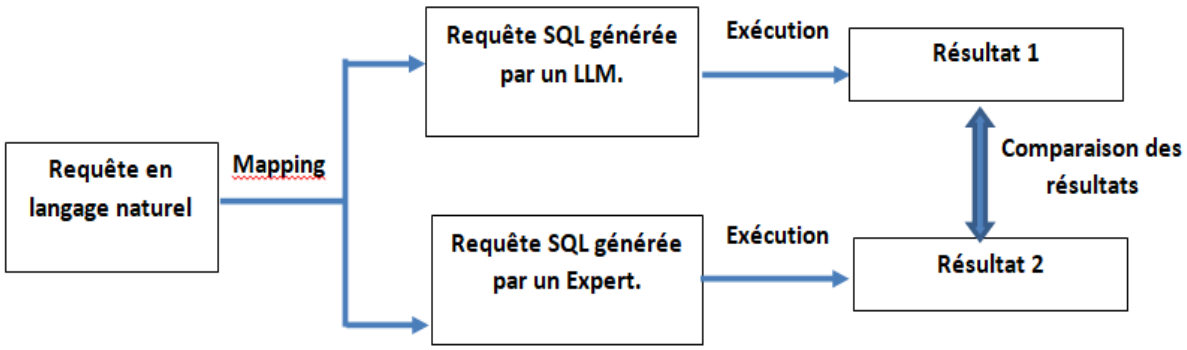

Figure III. 8: Schéma de la Méthode de Test

Donc, en partant d'une requête exprimée en langage naturel, à partir de laquelle on obtient deux requêtes SQL. Une requête générée par un LLM et une autre générée par un expert.

La suite de tests consiste à comparer le résultat obtenu par l'exécution de la requête générée par le modèle LLM, avec celui de la requête de l'expert. Si les deux résultats correspondent, on considère que le modèle a généré une réponse correcte, sinon c'est une réponse erronée.

Le tableau suivant donne un exemple des requêtes utilisées dans nos tests :

SQL Query	Human Question	Results		Human Results
		LLM1	LLM2	
SELECT COUNT(EmployeeId) FROM Employee	how many employees are there	There are a total of 8 employees in the company.	Based on the SQL response, there are 8 employees in the database.	COUNT(DISTINCT FirstName) 8
SELECT a.AlbumId, COUNT(t.TrackID) as Tracks FROM Album a JOIN Track t ON a.AlbumID =t.AlbumID GROUP BY a.AlbumId ORDER BY Tracks desc LIMIT 10	show the top 10 albums with the highest number of tracks. The result should contain 2 columns: album ID and number of tracks in the album (name the column "Tracks").	The top 10 albums with the highest number of tracks are: 1-Album ID: 141, Tracks: 57 2-Album ID: 23, Tracks: 34 3-Album ID: 73, Tracks: 30 4-Album ID: 229, Tracks: 26 5-Album ID: 230, Tracks: 25 6-Album ID: 251, Tracks: 25 7-Album ID: 253, Tracks: 24 8-Album ID: 231, Tracks: 24 9-Album ID: 83, Tracks: 24 10-Album ID: 24, Tracks: 23	AlbumId Tracks 141 57 23 34 73 30 229 26 230 25 251 25 253 24 231 24 83 24 24 23	AlbumId Tracks 141 57 23 34 73 30 229 26 230 25 251 25 83 24 231 24 253 24 24 23
SELECT EmployeeId, (firstName || ' ' || lastName) as FullName, HireDate FROM Employee WHERE strftime('%Y',HireDate) = '2002'	give me the list of employees hired after the year 2001 & before the year 2003 with their full names, employeeId & the date of hiring.	Sure! Here is the list of employees hired after the year 2001 & before the year 2003 with their full names, employeeId, and the date of hiring: Employee ID: 1 Full Name: Andrew Adams Hire Date: August 14, 2002 Employee ID: 2 Full Name: Nancy Edwards Hire Date: May 1, 2002 Employee ID: 3 Full Name: Jane Peacock Hire Date: April 1, 2002 These are the employees who were hired within the specified timeframe. Let me know if you need any more information!	This query will return the employee ID, last name, first name, and hire date for all employees who were hired between the years 2002 and 2003.The SQL response to this query is: [(1, 'Adams', 'Andrew', datetime.datetime(2002, 8, 14, 0, 0)), (2, 'Edwards', 'Nancy', datetime.datetime(2002, 5, 1, 0, 0)), (3, 'Peacock', 'Jane', datetime.datetime(2002, 4, 1, 0, 0)), (4, 'Park', 'Margaret', datetime.datetime(2003, 5, 3, 0, 0)), (5, 'Johnson', 'Steve', datetime.datetime(2003, 10, 17, 0, 0)), (6, 'Mitchell', 'Michael', datetime.datetime(2003, 10, 17, 0, 0))].This response indicates that there were 6 employees who were hired between the years 2002 and 2003. Their employee IDs, last names, first names, and hire dates are included in the response.	EmployeeId FullName HireDate 1 Andrew Adams 2002-08-14 00:00:00 2 Nancy Edwards 2002-05-01 00:00:00 3 Jane Peacock 2002-04-01 00:00:00

Tableau III. 1:Exemple de requêtes de test

En plus du test précédent, il est important de mentionner que nous avons relevé plusieurs observations concernant la précision et la clarté des réponses, ainsi que le temps de réponse des modèles utilisés. Toutes ces observations seront discutées dans la section suivante.

III.4.2 Test et Résultat

Dans nos tests nous avons utilisé la base de données Chinook. Nous avons sélectionné un ensemble diversifié de 50 requêtes couvrant différents aspects de la base de données Chinook, incluant des requêtes simples, intermédiaires et complexes. Chaque requête en langage naturel était associée à sa requête SQL correspondante et aux résultats attendus, ce qui a permis de créer un benchmark fiable pour l'évaluation.

Pour les modèles, nous avons utilisé GPT-3.5-turbo et Mixtral 8x7B Le tableau Suivant présente une comparaison entre ces deux modèles :

Caractéristique	GPT-3.5 Turbo	Mixtral 8x7B
Fournisseur L'entité qui fournit ce modèle.	OpenAI	Mistral
Fenêtre contextuelle d'entrée Le nombre de jetons pris en charge par la fenêtre contextuelle d'entrée.	4,096 tokens	32K tokens
Nombre maximum de jetons en sortie Le nombre maximal de jetons pouvant être générés par le modèle dans une seule requête.	4,096 tokens	4,096 tokens
Date de sortie Quand le modèle a été publié pour la première fois.	November 28th, 2022 plus d'un an	December 11th, 2023 Il y a 6 mois
MMLU Évaluation de l'acquisition de connaissances LLM en configuration zéro-shot et few-shot.	70.0 (5-shot)	70.6 (5-shot)
HellaSwag Un banc d'essai difficile pour la complétion de phrases	85.5 (10-shot)	Benchmark not available.

Tableau III. 2: Comparaison entre Mixtral 8x7B et GPT-3.5Turbo [36]

La figure suivante *(Voir Figure III.9)* comptabilise le nombre de réponses correctes pour chaque modèle sur l'ensemble des tests effectués.

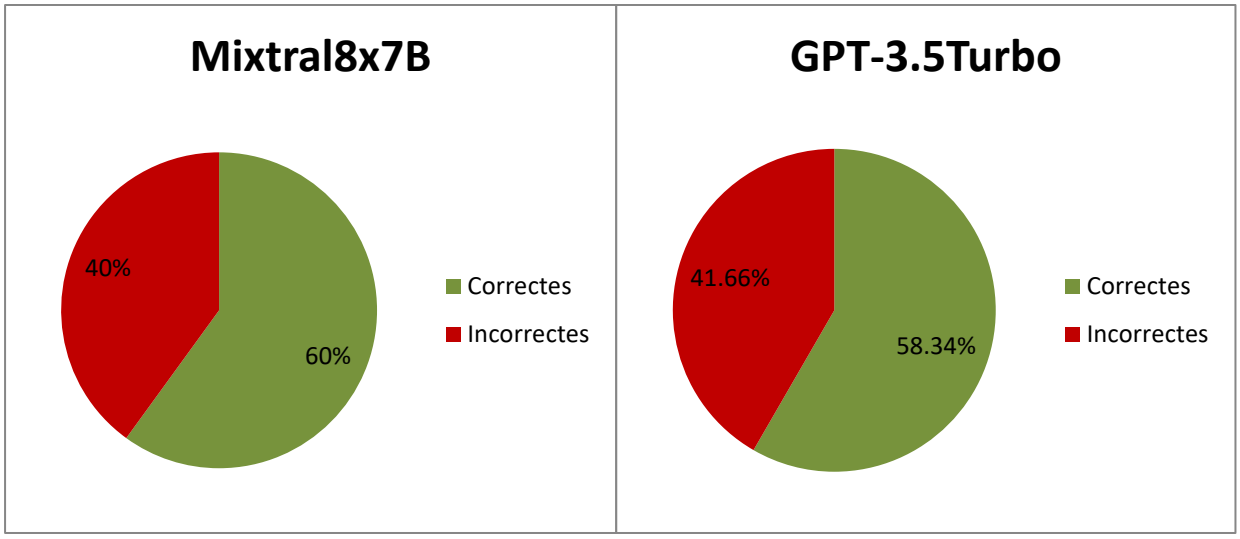

Figure III. 9: Résultats de Test

Nous avons également mesuré le temps de réponse moyen pour générer une requête SQL correcte et évalué la clarté et la pertinence des réponses en langage naturel fournies par les modèles. Le tableau suivant résume le temps moyen de réponses pour chaque modèle :

Modèle	**Temps de Réponse**
GPT-3.5-turbo	1,2secondes
Mixtral-8x7b-32768	1,5secondes

Tableau III. 3: le temps moyen de réponses pour chaque modèle

En lisant les résultats pour les deux modèles, Mixtral et GPT-3, nous constatons plusieurs tendances visibles :

- Dans l'ensemble des tests, Mixtral a produit 60 % de requêtes correctes.

- Le nombre de requêtes incorrectes représentaient 40 % du total. Cela montre une performance plus élevée de Mixtral par rapport à GPT-3 dans la traduction du langage naturel en requêtes SQL.

- GPT-3 a montré une performance légèrement inférieure à Mixtral, avec 58.34% de requêtes correctes.

- Le nombre de requêtes incorrectes représentaient 41.66 % du total.

Ces résultats indiquent que, pour cette tâche spécifique, Mixtral pourrait offrir une expérience utilisateur plus satisfaisante et précise que GPT-3. Cependant, il est important de noter que les performances peuvent varier en fonction des types de requêtes et des domaines d'application, et d'autres tests pourraient être nécessaires pour confirmer ces conclusions dans différents contextes.

Le taux élevé de réponses incorrectes pour les deux modèles, approchant les 40%, s'explique par le fait que les modèles utilisés sont génériques et ne sont pas spécialement entraînés pour la génération de requêtes SQL. Une solution à cette situation pourrait être l'utilisation d'un modèle spécifique déjà pré-entraîné pour la génération de requêtes SQL comme SQLCoder-34B [37]. Une autre solution serait d'effectuer un affinement (finetuning) des modèles utilisés.

GPT-3.5-turbo a montré des temps de réponse généralement plus rapides que Mixtral, ce qui peut être attribué à son optimisation par OpenAI.

Enfin, la qualité des réponses en langage naturel a été plus élevée avec GPT-3.5-turbo, ses réponses étant généralement plus claires et pertinentes. Mixtral-8x7b-32768 a également fourni des réponses précises, mais parfois avec une clarté légèrement inférieure.

Cette analyse met en lumière les forces et les faiblesses de chaque modèle, offrant ainsi des informations précieuses pour guider le choix du modèle le plus adapté à des applications spécifiques.

III.5 Conclusion

Dans ce chapitre, nous avons présenté notre travail qui consiste à développer un système appelé « Chat with MySQL ». Ce système permet aux utilisateurs de poser des questions en langage naturel et d'obtenir des réponses SQL appropriées en temps réel. À travers l'utilisation de différentes bibliothèques et modules, nous avons construit un pipeline de traitement robuste qui gère efficacement les requêtes utilisateur, l'exécution de requêtes SQL et la génération de réponses.

L'application offre une interface utilisateur conviviale grâce à l'utilisation de Streamlit, ce qui permet aux utilisateurs de naviguer facilement et d'interagir avec l'assistant de requêtes SQL. De plus, la gestion de l'historique des conversations permet de maintenir le contexte des interactions précédentes, ce qui améliore la pertinence des réponses fournies.

En conclusion, notre système « Chat with MySQL » offre une solution pratique et efficace pour simplifier l'interaction avec les bases de données, en particulier pour les utilisateurs moins familiers avec le langage SQL. Les fonctionnalités de génération automatique de requêtes SQL et d'affichage clair des résultats contribuent à améliorer l'efficacité et la convivialité de l'application, ouvrant ainsi la voie à de nombreuses applications potentielles dans le domaine de l'analyse de données et de la science des données.

Conclusion Générale

Dans ce mémoire, nous avons abordé la problématique de l'interrogation des bases de données relationnelles en langage naturel. Nous avons constaté que les méthodes traditionnelles, principalement basées sur le langage SQL, présentent des défis majeurs pour les utilisateurs non techniques, notamment en termes de complexité et de temps nécessaire pour formuler des requêtes efficaces.

Pour pallier ces défis, nous avons exploré l'utilisation des grands modèles de langage (LLMs) comme solution prometteuse. Les LLMs, grâce à leur compréhension approfondie du langage naturel, permettent aux utilisateurs de poser des questions complexes sans avoir à se soucier de la syntaxe spécifique des requêtes SQL. Cela ouvre la voie à une approche plus intuitive et accessible à tous les utilisateurs, quelle que soit leur expertise technique.

Du fait, nous avons mis en œuvre une application nommée "Chat with MySQL", permettant aux utilisateurs d'interroger et de tirer des informations d'une base de données en langage naturel. Cette application témoigne de la faisabilité et de l'utilité de l'interrogation en langage naturel des bases de données, en utilisant les LLMs comme moteur de transformation des requêtes.

Les résultats de nos tests indiquent que notre système, basé sur les modèles GPT-3.5-turbo et Mixtral-8x7b-32768, excelle dans la génération de requêtes SQL précises et dans la fourniture de réponses en langage naturel de haute qualité. L'utilisation de la base de données Chinook comme référence a été cruciale pour évaluer l'efficacité de notre système dans la compréhension des questions en langage naturel et dans la production de résultats pertinents et exacts.

En envisageant l'avenir de notre système, plusieurs implications et perspectives émergent. Tout d'abord, une optimisation continue des prompts utilisés peut améliorer la capacité des modèles à traiter des requêtes complexes, offrant ainsi une expérience utilisateur plus fluide et intuitive. De plus, l'enrichissement des modèles avec des ensembles de données supplémentaires pourrait renforcer leur compréhension contextuelle et leur capacité à répondre à une gamme plus large de questions.

L'intégration d'un mécanisme de feedback permettrait aux utilisateurs de signaler les réponses incorrectes, contribuant ainsi à une amélioration continue du système. Ce processus itératif d'amélioration basé sur les retours utilisateurs est essentiel pour garantir que notre système reste pertinent et performant dans un environnement en évolution constante.

En conclusion, ce travail a jeté les bases d'une approche novatrice et prometteuse pour l'interrogation des bases de données en langage naturel. Bien que des défis subsistent, notamment en termes de compréhension sémantique et d'ambiguïté des requêtes, ce mémoire ouvre la voie à de futures recherches et applications dans ce domaine émergent des LLMs et de l'interrogation en langage naturel des bases de données.

Bibliographie

[1] «What are Large Language Models(LLMs)?,» 10 january 2024. [En ligne]. Available: https://www.geeksforgeeks.org/large-language-model-llm/. [Accès le 27 April 2024].

[2] M. Shervin , M. Tomas et N. Narjes , «Large Language Models:A Survey,» p. 43, 09 february 2024.

[3] S. Talebi, «How to Build an LLM from Scratch,» 21 september 2023. [En ligne]. Available: https://towardsdatascience.com/how-to-build-an-llm-from-scratch-8c477768f1f9. [Accès le 2024 april 23].

[4] A. N. S. N. P. J. U. L. J. A. N. G. L. K. I. P. Vaswani, «Attention Is All You Need,» p. 15, 02 August 2017.

[5] S. Mahapatra, «The Computer was Born to solve Problems,» 01 December 2023. [En ligne]. Available: https://in.linkedin.com/in/shivmahapatra?trk=article-ssr-frontend-pulse_publisher-author-card.

[6] N. Pallav, «NLP, LLM and Generative AI: The differences and overlaps,» 19 August 2023. [En ligne]. Available: https://www.linkedin.com/pulse/nlp-llm-generative-ai-differences-overlaps-nav-pallav. [Accès le 27 april 2024].

[7] «Large Language Models (LLM): Guide complet en 2023,» 2023. [En ligne]. Available: https://fr.shaip.com/blog/a-guide-large-language-model-llm/.

[8] «Elastic Key Components of Large Language Models,» [En ligne]. Available: https://www.elastic.co/fr/what-is/large-language-models/#key-components-of-large-language-models.

[9] «Grand Modele de Langage,» [En ligne]. Available: https://www.hpe.com/emea_africa/fr/what-is/large-language-model.html.

[10] «Grands modèles de langage (LLM),» [En ligne]. Available: https://www.databricks.com/fr/glossary/large-language-models-llm. [Accès le 28 april 2024].

[11] S. M. Kerner, «What are large language models (LLMs)?,» [En ligne]. Available: https://www.techtarget.com/whatis/definition/large-language-model-LLM.

[12] M. Myer, «Are Generative AI and Large Language Models the Same Thing?,» 12 mai 2023. [En ligne]. Available: https://www.elastic.co/fr/what-is/large-language-models/#cas-dutilisation-des-grands-mod%C3%A8les-de-langage. [Accès le 23 april 2024].

[13] S. Karra, «Exploring the role of a Prompt Engineer,» 05 jun 2023. [En ligne]. Available: https://kindlingthoughts.medium.com/exploring-the-role-of-a-prompt-engineer-9bc2fae94ed.

[14] «Prompt Engineering,» [En ligne]. Available: https://www.promptingguide.ai/fr/introduction/elements. [Accès le 29 april 2024].

[15] M. Ggaliwango, H. Nakayiza, J. Daudi et N.-N. Joyce , «Prompt Engineering in Large Language,» p. 16, 09 janvier 2024.

[16] YUHANITO, «Prompt Engineering – Get it right,» [En ligne]. Available: https://yuhanito.com/prompt-engineering/. [Accès le 2024].

[17] S. Venkataraman, Crafting Effective Prompts_ A Guide to Prompt Engineering, 2024.

[18] A. Kirkovska, «Zero-Shot vs Few-Shot prompting: A Guide with Examples,» 21 december 2023. [En ligne]. Available: https://www.vellum.ai/blog/zero-shot-vs-few-shot-prompting-a-guide-with-examples.

[19] Y. Minne, «Few-shot Prompting – Guide Sydologie ChatGPT,» 2024 april 29. [En ligne]. Available: https://sydologie.com/2024/04/few-shot-prompting-guide-sydologie-chatgpt/.

[20] G. Jeandot, «Zero-shot prompting : la forme la plus élémentaire du prompting,» 29 december 2023. [En ligne]. Available: https://fr.linkedin.com/pulse/zero-shot-prompting-la-forme-plus-%C3%A9l%C3%A9mentaire-du-gr%C3%A9gory-jeandot-obdne.

[21] S. R.-E. A. strategist, «Master Prompting Techniques: Knowledge Generation Prompting,» 26 april 2023. [En ligne]. Available: https://promptengineering.org/knowledge-generation-prompting/.

[22] C. C, «Guide sur le Prompt Engineering : Définitions, Utilisations et Limites,» 28 march 2024. [En ligne]. Available: https://www.hostinger.fr/tutoriels/prompt-engineering#15_Prompt_Engineering_automatique.

[23] J. Schmitt, «What is Prompt Engineering? Definitions, Applications, and Limits,» 12 January 2024. [En ligne]. Available: https://circleci.com/blog/prompt-engineering/. [Accès le 01 mai 2024].

[24] E. Saravia, «Prompt Engineering,» p. 40.

[25] «Qu'est-ce que le Prompt engineering : définition, applications et limites,» [En ligne]. Available: https://www.salesforce.com/fr/resources/definition/prompt-engineering/. [Accès le 01 mai 2024].

[26] Y. Li, Alicia Sun, Liangliang Wang, Zhaopeng Zhang , Scott J Rennie et and Daniel Jurafsky , «Aunified neural network for sql parsing and semantic parsing,» 2017.

[27] «Welcome to Python.org,» Python, [En ligne]. Available: https://www.python.org/.

[28] «motdotla/dotenv,» [En ligne]. Available: https://github.com/motdotla/dotenv. [Accès le 2024].

[29] «Streamlit • A faster way to build and share data apps,» [En ligne]. Available: https://streamlit.io/. [Accès le mai 2024].

[30] «Applications that can reason. Powered by LangChain,» [En ligne]. Available: https://www.langchain.com/.

[31] «sqlcmd (utilitaire),» [En ligne]. Available: https://learn.microsoft.com/fr-fr/sql/tools/sqlcmd/sqlcmd-utility?view=sql-server-ver16&tabs=go%2Cwindows&pivots=cs1-bash.

[32] «SQLite Download Page,» 10 october 2023. [En ligne]. Available: https://www.sqlite.org/download.html.

[33] «MySQL HeatWave,» Oracle, [En ligne]. Available: https://www.mysql.com/downloads/.

[34] «Chinook sample database,» [En ligne]. Available: https://docs.yugabyte.com/preview/sample-data/chinook/.

[35] R. O. Rauf, «Chinook Database — Querying a Digital Music Store Database.",» 15 November 2023. [En ligne]. Available: https://medium.com/@raufrukayat/chinook-database-querying-a-digital-music-store-database-8c98cf0f8611.

[36] «compare GPT-3.5 Turbo vs. Mistral 8x7B Instruct,» [En ligne]. Available: https://context.ai/compare/gpt-3-5-turbo/mistral-8x7b-instruct. [Accès le 2024].

[37] «defog-ai/sqlcoder,» [En ligne]. Available: https://github.com/defog-ai/sqlcoder. [Accès le 06 05 2024].

Résumé

Notre projet a pour objectif de simplifier l'interrogation des bases de données relationnelles en langage naturel, une tâche souvent complexe pour les utilisateurs non techniques. À cette fin, nous avons développé une application nommée "Chat avec MySQL", permettant aux utilisateurs d'interroger et d'extraire des informations d'une base de données en langage naturel. Cette application démontre la faisabilité et l'utilité de cette approche, offrant ainsi une solution intuitive et efficace pour accéder et exploiter les données sans nécessiter une expertise technique approfondie. Dans cette application, l'utilisateur sélectionne la base de données à interroger et pose ses questions via une interface conviviale. Le système traite ensuite la requête, récupère les informations pertinentes, les formate en langage naturel et les affiche à l'utilisateur. Ce processus d'interaction se déroule de manière transparente, facilitant ainsi l'accès aux données pour tous les utilisateurs.

Mots-clés :texte-à-SQL, BDD relationnelle, requêtes en langage naturel, NLP.

Abstract

Our project aims to simplify querying relational databases using natural language, a task often complex for non-technical users. To achieve this goal, we developed an application called 'Chat with MySQL' that enables users to query and extract information from databases using everyday language. This application demonstrates the feasibility and utility of our approach by providing an intuitive and efficient solution for accessing and leveraging data without requiring extensive technical expertise. Within the application, users select the database they wish to query and ask questions through a user-friendly interface. The system then processes the query, retrieves relevant information, converts it into natural language, and presents it to the user. This seamless interaction makes data access more accessible for all users.

Keywords: text-to-SQL, relational database, natural language queries, NLP.

ملخص

مشروعنا يهدف إلى تبسيط استعلام قواعد البيانات العلاقية بلغة طبيعية، وهو مهمة غالبًا ما تكون معقدة للمستخدمين غير التقنيين ، الذي يتيح للمستخدمين استعلام واستخراج "Chat with MySQL" ولتحقيق هذا الهدف، قمنا بتطوير تطبيق يسمى المعلومات من قاعدة بيانات باستخدام اللغة الطبيعية. يُظهر هذا التطبيق إمكانية وفائدة هذا النهج، حيث يوفر حلاً بديهيًا وفعالًا للوصول إلى البيانات واستغلالها دون الحاجة إلى خبرة تقنية متقدمة. في هذا التطبيق، يُختار المستخدم قاعدة البيانات التي يرغب في الاستعلام عنها ويطرح الأسئلة من خلال واجهة مستخدم سهلة الاستخدام. يقوم النظام بمعالجة الاستعلام، واسترداد المعلومات ذات الصلة، وتنسيقها بلغة طبيعية، وعرضها للمستخدم. يتم هذا العملية التفاعلية بسلاسة، مما يسهل على جميع المستخدمين الوصول إلى البيانات.

كلمات المفتاحية: نص الىSQL ، قاعدة بيانات علاقية، استعلامات باللغة الطبيعية ،NLP.