



Ministère de l'Enseignement  
Supérieur et de la Recherche  
Scientifique

Année Universitaire  
2023/2024



**EPI** DIGITAL SCHOOL  
Ecole Internationale Supérieure  
Privée Polytechnique de Sousse

# RAPPORT DE PROJET DE FIN D'ANNÉE

**Spécialité: Intelligence Artificielle**

## Intitulé

**AI-DocGen: Générateur de documentation intelligent**

Réalisé par

**Eltaief Aymen**

Encadré par

**Amri Hassen**



---

# DEDICATION

Je dédie ce modeste travail :

**Mes chers parents Salem et Charifa** vous êtes toujours le symbole de patience et de tolérance, votre amour et vos encouragements m'ont permis d'acquérir toute la force et la résolution pour accomplir ce travail. Merci pour vos sacrifices, votre tendresse et vos prières ferventes qui m'ont accompagnée durant toutes ces années.

**A mes sœur et mon frère** pour leurs soutien moral et leurs encouragements, avec tous mes vœux de les voir réussir leurs vies.

**Aymen**



---

# REMERCIEMENT

Tout d'abord, je souhaite remercier Monsieur **Hassen Amri**, mon encadrant de projet, pour sa rigueur, sa disponibilité, son encadrement pédagogique et ses conseils. Il a dirigé mon travail avec beaucoup d'attention et n'a ménagé aucun effort pour l'accomplissement et la réussite de ce projet. Je tiens à lui exprimer vivement tout mon respect.

J'exprime également mes vifs remerciements et mes reconnaissance aux membres du jury qui m'a fait un immense honneur en acceptant de participer à l'évaluation de ce projet et de l'enrichir par leurs propositions.

Finalement je remercie toutes les personnes qui ont apporté leur appui de près ou de loin, directement ou indirectement à la réalisation de ce Projet.



---

# TABLE DES MATIÈRES

<b>LISTE DES FIGURES</b>	<b>vi</b>
<b>LISTE DES ABRÉVIATIONS</b>	<b>vii</b>
<b>INTRODUCTION GÉNÉRALE</b>	<b>1</b>
<b>1 Cadre général du projet</b>	<b>2</b>
1.1 Introduction . . . . .	3
1.2 Présentation du projet . . . . .	3
1.3 Cadre du projet . . . . .	3
1.4 Problématique . . . . .	4
1.5 Structure du rapport . . . . .	4
1.6 Conclusion . . . . .	5
<b>2 État de l’art</b>	<b>6</b>
2.1 Introduction . . . . .	7
2.2 Définition des concepts de base du projet . . . . .	7
2.2.1 Intelligence Artificielle . . . . .	7
2.2.2 Le traitement du langage naturel . . . . .	9
2.2.3 Les grands modèles de langage . . . . .	10
2.2.4 Approche algorithmique utilisés . . . . .	11
2.3 Etude de l’existant . . . . .	13
2.3.1 Analyse de la littérature existante . . . . .	13
2.3.2 Présentation des travaux antérieurs . . . . .	13
2.3.3 Identification des lacunes de recherche . . . . .	14
2.4 Solution proposée . . . . .	15
2.5 Méthodologie . . . . .	15
2.5.1 Introduction . . . . .	15
2.5.2 Méthodologie choisie . . . . .	16

2.5.3	Les phases du projet . . . . .	17
2.5.4	Workflow du projet . . . . .	19
2.5.5	Planning du projet . . . . .	20
2.6	Conclusion . . . . .	20
<b>3</b>	<b>Traitement des Données et Entraînement du modèle</b>	<b>22</b>
3.1	Introduction . . . . .	23
3.2	Collection des données . . . . .	23
3.2.1	Choix du Format de Données pour les Modèles LLMs . . . . .	24
3.2.2	Étude des DataSets opensource . . . . .	24
3.3	Compréhension et Analyse des données . . . . .	25
3.3.1	Choix du DataSet . . . . .	25
3.3.2	Analyse des données . . . . .	26
3.4	Prétraitement des Données . . . . .	27
3.5	Analyse et Sélection des Modèles . . . . .	29
3.5.1	Etude comparative . . . . .	30
3.5.2	Critique de la performance des modèles . . . . .	32
3.5.3	Choix du modèle . . . . .	35
3.6	Entraînement du Modèle . . . . .	35
3.6.1	Analyse des besoins en ressources matérielles et logicielles . . . . .	35
3.6.2	Exploration de Quantization . . . . .	36
3.6.3	Entraînement du Modèle : Optimisation des Performances . . . . .	38
3.7	Tests de Validation . . . . .	39
3.8	Conclusion . . . . .	43
<b>4</b>	<b>Évaluation</b>	<b>44</b>
4.1	Introduction . . . . .	45
4.2	Évaluation du Modèle . . . . .	45
4.3	Évaluation des Performances . . . . .	46
4.4	Comparaison avec les travaux antérieurs . . . . .	49
4.5	Conclusion . . . . .	50
	<b>CONCLUSION GÉNÉRALE</b>	<b>51</b>
	<b>Webographie</b>	<b>52</b>

# LISTE DES FIGURES

2.1	Les sous-domaines de l'intelligence artificielle . . . . .	8
2.2	Processus de Traitement du Langage Naturel . . . . .	9
2.3	L'état de l'art en matière de LLM : Un aperçu des performances . . . . .	10
2.4	L'architecture d'un transformer . . . . .	12
2.5	autoDocstring Python Docstring Generator . . . . .	13
2.6	Doxygen Documentation Generator . . . . .	14
2.7	Docify AI Docstring comment writer . . . . .	14
2.8	Architecture CRISP-DM méthodologie . . . . .	17
2.9	Workflow de l'approche méthodologique . . . . .	19
2.10	Diagramme de Gantt Réel . . . . .	20
3.1	Vue d'ensemble du dataset . . . . .	26
3.2	Aperçu de la structure et du contenu du Dataset . . . . .	26
3.3	Étapes de prétraitement des données . . . . .	27
3.4	Structure du Dataset après Prétraitement . . . . .	28
3.5	Répartition du Nombre de Colonnes de Sortie et d'Instruction . . . . .	28
3.6	Distribution des tokens dans le dataset . . . . .	29
3.7	Performances comparées des grands modèles de langage sur plusieurs tâches . . . . .	33
3.8	Scores de référence et métriques d'évaluation . . . . .	34
3.9	Les Types de Quantization de LLM . . . . .	38
3.10	Vue d'ensemble de l'entraînement et de la configuration du modèle . . . . .	39
3.11	Exemple(1) de Réponse Généré par le Modèle . . . . .	40
3.12	Exemple(2) de Réponse Généré par le Modèle . . . . .	41
3.13	Exemple(3) de Réponse Généré par le Modèle . . . . .	42
4.1	Évolution de la Perte d'Entraînement au Fil des Étapes . . . . .	46
4.2	Prompt template :Évaluation de réponse unique . . . . .	48
4.3	Évaluations d'exemples . . . . .	48
4.4	Révision des travaux préexistants(1) . . . . .	49

4.5 Révision des travaux préexistants(2) . . . . .	49
--	----



---

# LISTE DES ABRÉVIATIONS

**AI** Artificial Intelligence

**NLP** Natural Language Processing

**Llm's** Large Language Models

**CRISP-DM** Cross-Industry Standard Process for Data Mining

**JSON** JavaScript Object Notation

**PTQ** Post-Training Quantization

**QAT** Quantization-Aware Training

**GPU** Graphics processing unit

**MMLU** Measuring massive multitask language understanding





---

# INTRODUCTION GÉNÉRALE

La documentation de code est un pilier essentiel, facilitant la compréhension, la maintenance et l'évolution des projets. Cependant, la création manuelle de cette documentation reste une tâche laborieuse et sujette à des erreurs humaines. Pour répondre à ce défi persistant, notre projet vise à mettre en place un générateur automatique de documentation de code alimenté par l'intelligence artificielle.

Dans le contexte technologique actuel, les développeurs sont confrontés à la nécessité de maintenir une documentation exhaustive et conviviale pour leurs projets Python. C'est dans cette optique que notre initiative trouve sa pertinence, en cherchant à automatiser le processus de création de documentation pour les projets Python.

L'objectif principal de ce projet est d'offrir une solution innovante et efficace pour la génération automatique de documentation de code, permettant ainsi aux développeurs de consacrer davantage de temps à la conception et à l'amélioration de leurs projets.

Pour atteindre cet objectif, notre démarche se structure autour d'une exploration approfondie du paysage technologique, en se basant sur des concepts tels que les grands modèles de langage, le traitement du langage naturel et les techniques de transformation. Nous envisageons de mener des recherches approfondies, d'analyser les concepts clés et d'intégrer ces éléments pour développer un générateur de documentation de code efficace et fiable.

Tout au long de ce projet, notre approche consistera à définir des objectifs clairs, à identifier les défis potentiels et à concevoir une démarche méthodique pour les surmonter. Notre engagement envers l'innovation et l'excellence guidera chaque étape de notre progression vers la réalisation d'un générateur de documentation de code alimenté par l'intelligence artificielle.

---

# Cadre général du projet

## Sommaire

---

1.1	Introduction . . . . .	3
1.2	Présentation du projet . . . . .	3
1.3	Cadre du projet . . . . .	3
1.4	Problématique . . . . .	4
1.5	Structure du rapport . . . . .	4
1.6	Conclusion . . . . .	5

---

### 1.1 Introduction

L'édification solide d'une étude repose sur ses fondements, essentiels à sa réussite. Ce premier chapitre s'engage donc à explorer le cadre global du projet et à délimiter les objectifs à atteindre. Nous amorcerons notre parcours par une présentation du contexte global, suivi d'une analyse approfondie du cadre du projet, afin de garantir une appréhension approfondie de la problématique étudiée. Enfin, nous exposerons en détail l'architecture de notre projet.

### 1.2 Présentation du projet

Notre initiative s'inscrit dans le domaine de l'intelligence artificielle appliquée, visant à faciliter les processus liés au développement de logiciels. Notre démarche vise à fournir un support innovant et efficace aux développeurs et aux équipes travaillant sur des projets Python. Nous cherchons à optimiser leur productivité en leur permettant de consacrer moins de temps aux tâches fastidieuses et plus de temps à l'amélioration de leurs projets. En explorant l'intégration de l'intelligence artificielle dans la documentation de code, notre projet ouvre de nouvelles perspectives pour une évolution significative du processus de développement logiciel.

### 1.3 Cadre du projet

Ce projet s'inscrit dans le cadre d'un projet de fin d'année visant à consolider les compétences acquises et à mettre en pratique les connaissances développées au cours de la formation en Intelligence Artificielle et Data Science à l'École pluridisciplinaire internationale EPI Digital School, pour l'année universitaire 2023/2024. Il représente une opportunité de mettre en œuvre les concepts théoriques abordés et de relever un défi concret dans le domaine des Modèles de Langage Artificiels (MLA), en se basant sur les dernières techniques et outils en Intelligence Artificielle.

### 1.4 Problématique

Dans le domaine du développement logiciel, la documentation précise et complète des projets Python est essentielle pour garantir leur compréhension, leur maintenance et leur évolutivité. Cependant, le processus manuel de création de cette documentation peut être fastidieux et chronophage, entraînant des retards dans les projets et compromettant la qualité de la documentation.

- Face à ces défis, Comment pouvons-nous développer une solution basée sur l'intelligence artificielle pour automatiser efficacement la génération de documentation et de commentaires pour les projets Python ?
- Quelles approches méthodologiques peuvent garantir à la fois la rapidité et la fiabilité de cette automatisation, tout en assurant des descriptions précises et pertinentes du code source ?

### 1.5 Structure du rapport

La structure de ce rapport est élaborée de façon à offrir une vision approfondie de l'avancement du projet, en respectant une logique et une méthodologie rigoureuses.

Voici un aperçu de la structure du rapport :

Le premier chapitre est consacré à la présentation du contexte général dans lequel s'inscrit le projet. Il commence par exposer la problématique centrale que notre projet vise à résoudre, mettant en lumière les défis et les enjeux auxquels il répond. Ensuite, ce chapitre présente une vue d'ensemble du projet, détaillant ses objectifs principaux ainsi que sa portée. Il examine également le cadre plus large dans lequel le projet évolue. Enfin, ce chapitre expose la structure du rapport à suivre, fournissant ainsi un aperçu clair de son contenu et de son organisation.

Le deuxième chapitre nous présente le panorama actuel du champ pertinent pour notre projet. En établissant les concepts clés nécessaires à sa compréhension, nous allons ensuite passer en revue les travaux existants et les solutions proposées. Cette analyse nous permet de

repérer les faiblesses et les obstacles à surmonter. Nous présentons notre propre solution en mettant en lumière ses avantages par rapport aux solutions existantes.

Nous présentons également notre approche de travail. Une fois la méthodologie retenue et ses bénéfices pour notre projet décrits, nous présentons les étapes du projet en détail. Nous présentons également le déroulement du projet, en décrivant comment les différentes parties prenantes interagissent et comment les tâches sont organisées. Nous établissons le calendrier du projet en précisant les délais et les étapes à suivre pour garantir le succès du projet dans les délais impartis. Cette combinaison de l'analyse de l'état actuel du champ et de notre démarche méthodologique permet d'avoir un point de vue global cohérent et bien organisé pour orienter notre travail futur.

Dans le troisième chapitre, nous commençons par la collecte de données et procédons à une compréhension et une analyse approfondies. Nous prétraitons ensuite les données pour les rendre utilisables. Nous effectuons ensuite l'analyse et sélectionnons un modèle adapté à notre problème. Après avoir sélectionné un modèle approprié, nous commençons à le former. Enfin, nous effectuons des tests de validation pour évaluer les performances et la fiabilité du modèle.

Dans le quatrième chapitre, nous passons en revue l'efficacité de notre modèle, en évaluant sa performance et sa précision. Nous examinons également les performances globales du système et révisons les travaux existants pour identifier les améliorations potentielles.

Finalement nous clôturons ce rapport par une conclusion générale ainsi par les perspectives futures de ce travail.

## 1.6 Conclusion

Ce chapitre a offert une exploration approfondie des différents aspects de notre projet. Nous avons mis en évidence son axe central, son contexte général et la problématique sous-jacente. De plus, la présentation de la structure du rapport a donné un aperçu clair de la démarche suivie et des éléments à venir dans la suite de ce travail.

---

# État de l'art

## Sommaire

---

<b>2.1</b>	<b>Introduction . . . . .</b>	<b>7</b>
<b>2.2</b>	<b>Définition des concepts de base du projet . . . . .</b>	<b>7</b>
2.2.1	Intelligence Artificielle . . . . .	7
2.2.2	Le traitement du langage naturel . . . . .	9
2.2.3	Les grands modèles de langage . . . . .	10
2.2.4	Approche algorithmique utilisés . . . . .	11
<b>2.3</b>	<b>Etude de l'existant . . . . .</b>	<b>13</b>
2.3.1	Analyse de la littérature existante . . . . .	13
2.3.2	Présentation des travaux antérieurs . . . . .	13
2.3.3	Identification des lacunes de recherche . . . . .	14
<b>2.4</b>	<b>Solution proposée . . . . .</b>	<b>15</b>
<b>2.5</b>	<b>Méthodologie . . . . .</b>	<b>15</b>
2.5.1	Introduction . . . . .	15
2.5.2	Méthodologie choisie . . . . .	16
2.5.3	Les phases du projet . . . . .	17
2.5.4	Workflow du projet . . . . .	19
2.5.5	Planning du projet . . . . .	20
<b>2.6</b>	<b>Conclusion . . . . .</b>	<b>20</b>

---

## 2.1 Introduction

Ce chapitre posera tout d'abord les cadres de notre projet en précisant les concepts essentiels à sa compréhension. Ensuite, nous examinerons les travaux en cours pour mettre en évidence les avancées et les lacunes. Enfin, nous exposerons notre solution offerte, en mettant en évidence ses principales caractéristiques et ses points forts par rapport aux autres solutions. Au fil du temps, nous préciserons également le calendrier de réalisation du projet, ses principales étapes et les échéances nécessaires à sa réalisation.

## 2.2 Définition des concepts de base du projet

Au sein de cette partie, nous visons à établir les fondements essentiels nécessaires à une compréhension approfondie de notre sujet. Nous commencerons par explorer les notions fondamentales de l'intelligence artificielle, du traitement du langage naturel et des principaux modèles de langage. Cette introduction jettera les bases nécessaires pour une exploration plus approfondie de notre projet.

### 2.2.1 Intelligence Artificielle

L'intelligence artificielle est un domaine scientifique et d'ingénierie dédié à la création de machines intelligentes, y compris les programmes informatiques intelligents. Elle consiste en la capacité d'un système informatique à imiter des fonctions cognitives humaines telles que l'apprentissage et la résolution de problèmes.[1]

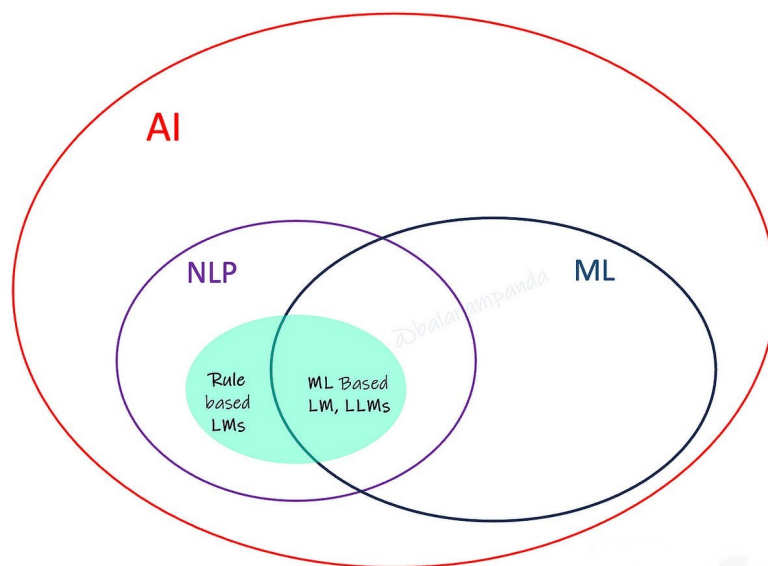
Les progrès dans le domaine de l'intelligence artificielle sont actuellement principalement attribués à trois éléments essentiels.

- **Les Algorithmes** : Les algorithmes offrent la possibilité d'automatiser des tâches qui étaient autrefois considérées comme exclusivement réalisables grâce à l'intelligence humaine. Les algorithmes évoluent constamment et se complexifient grâce à l'utilisation de couches comportant des variables cachées, ce qui permet de trier et d'optimiser les résultats.

- **Le Big Data** : Les algorithmes employés par l'IA sont principalement influencés par la quantité considérable d'informations présentes dans le Big Data. Ensuite, ces algorithmes permettent à l'intelligence artificielle de traiter ces informations à un rythme incroyable et de rendre les données plus accessibles et plus utilisables.
- **Les unités de traitement graphique (GPU)** : La demande croissante dans le domaine de la vidéo et des jeux a conduit à l'élaboration de GPU améliorées et moins coûteuses, un élément essentiel pour la création de solutions d'intelligence artificielle.

Parmi les sous-domaines de l'intelligence artificielle, deux domaines cruciaux sont le Traitement Automatique du Langage Naturel (NLP) et les Modèles de Langage à Grande Échelle (LLM). Ces deux disciplines jouent un rôle essentiel dans la compréhension et la génération du langage humain par les machines, ouvrant ainsi la voie à une interaction plus naturelle entre les humains et les ordinateurs.

La figure 2.1 présente les sous-domaines de l'intelligence artificielle :



**FIGURE 2.1 – Les sous-domaines de l'intelligence artificielle**



## 2.2.2 Le traitement du langage naturel

Le traitement du langage naturel (NLP) est une discipline de l'informatique, intégrée dans le domaine de l'intelligence artificielle (IA), visant à doter les ordinateurs de la capacité à comprendre et à interpréter les textes et les paroles humaines de manière similaire aux êtres humains.

Cette approche combine la linguistique informatique, qui modélise le langage humain selon des règles, avec des techniques de modélisation statistique, d'apprentissage automatique et d'apprentissage en profondeur. Grâce à ces technologies, les ordinateurs peuvent traiter le langage humain sous forme de texte ou de données vocales et en extraire le sens complet, en tenant compte de l'intention et du ressenti du locuteur ou de l'auteur.

Le traitement du langage naturel comprend des tâches telles que la reconnaissance du langage naturel et la génération de langage naturel, qui sont des applications clés de cette discipline dans le domaine de l'intelligence artificielle. [2]

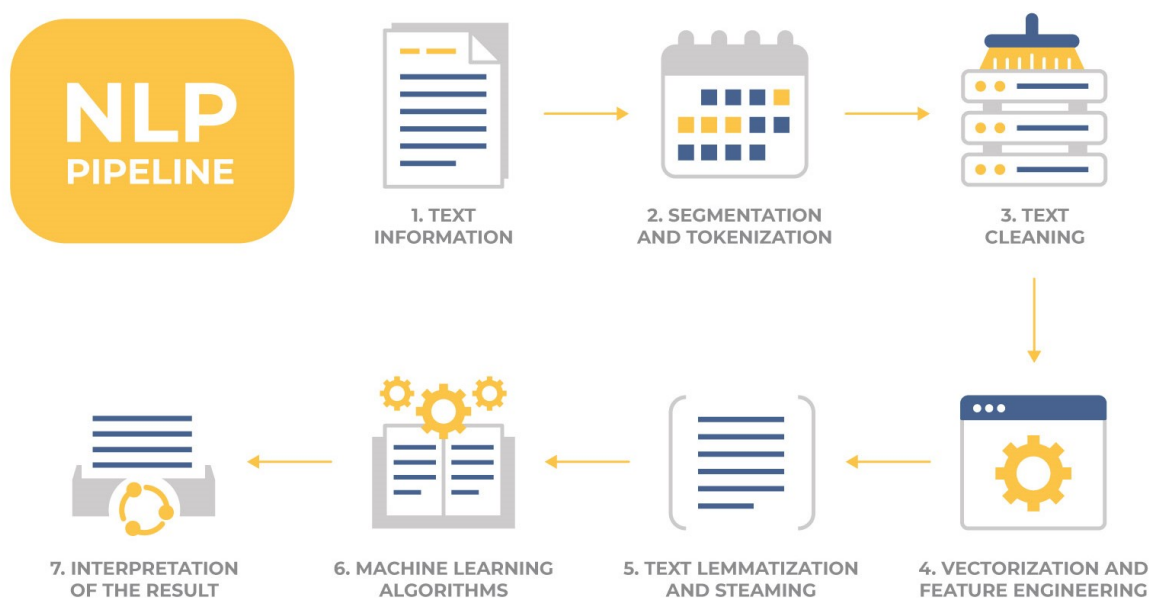


FIGURE 2.2 – Processus de Traitement du Langage Naturel

### 2.2.3 Les grands modèles de langage

Un grand modèle de langage (LLM) est une forme d'intelligence artificielle développée pour saisir et produire du langage naturel, similaire à la communication humaine.

Ces modèles sont construits sur des réseaux de neurones profonds, qui sont des algorithmes informatiques inspirés par les structures et fonctions du cerveau humain. Ces réseaux de neurones sont capables de traiter d'énormes corpus textuels – des collections de documents écrits – pour apprendre la langue de manière statistique.

Cette formation offre aux LLMs la possibilité de créer du texte et d'interpréter le langage avec une subtilité qui imite la fluidité humaine, leur donnant ainsi la possibilité de s'occuper de tâches telles que la traduction, la rédaction, et même la création de poésie ou de prose.

Les LLMs opèrent grâce à l'apprentissage automatique, une technique qui permet au modèle d'être exposé à de grandes quantités de texte. Au fil du temps, le modèle repère et acquiert des motifs fréquents dans la langue : il réalise quelles combinaisons de mots sont courantes, comment les mots se combinent pour former des phrases, et comment le sens évolue en fonction du contexte.

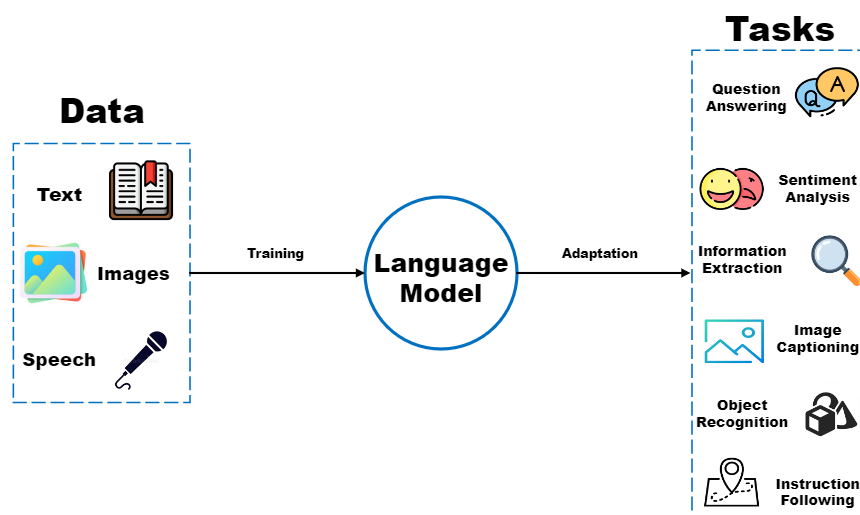


FIGURE 2.3 – L'état de l'art en matière de LLM : Un aperçu des performances

## 2.2.4 Approche algorithmique utilisés

Les Large Language Models (LLMs) et le domaine du Traitement du Langage Naturel (NLP) ont connu une évolution fulgurante ces dernières années, grâce aux progrès des algorithmes de Deep Learning. Parmi les nombreux algorithmes disponibles, nous nous concentrerons dans cette section sur ceux qui sont essentiels pour notre projet.

- **Transformers :**

La première IA basée sur Transformers est apparue dans le domaine du traitement du langage naturel (NLP) fin 2017. Ces avancées significatives sont le résultat des travaux innovants de chercheurs et de laboratoires renommés comme Google et OpenAI. Ces nouveaux modèles révolutionnent la façon dont les tâches de NLP sont traitées, grâce à leur utilisation efficace des mécanismes d'attention et de leurs capacités d'apprentissage semi-supervisé.

Un Transformer est un réseau de neurones pré-entraîné sur un vaste corpus de textes, lui permettant de découvrir des relations entre les tokens (ou mots apparaissant dans le texte). Ces tokens sont souvent représentés sous forme de vecteurs numériques à l'aide de embeddings contextuels, qui capturent les informations contextuelles importantes. L'idée clé derrière les Transformers est de conserver l'interdépendance des tokens de la même séquence, en utilisant le mécanisme d'attention pour accorder plus ou moins d'importance à chaque token en fonction de son contexte. [3]

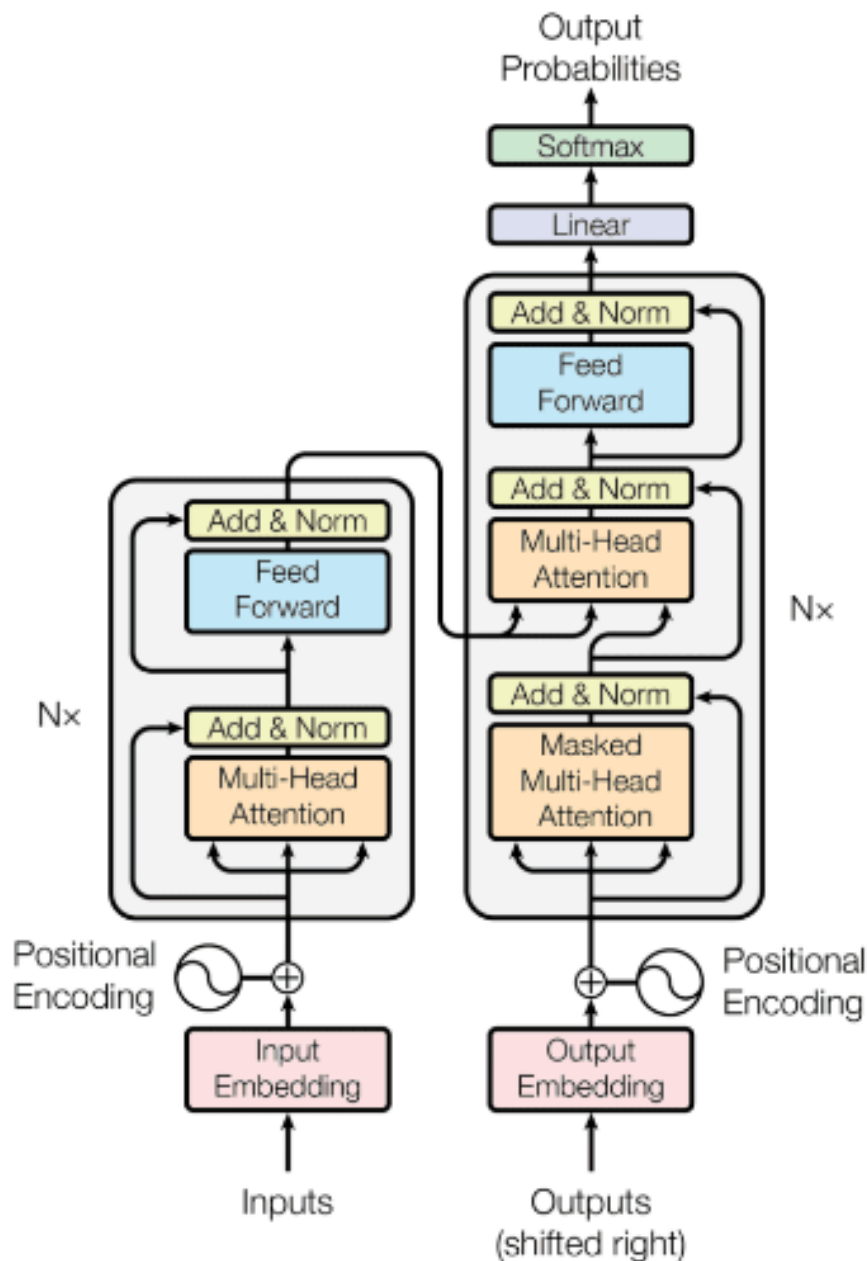


FIGURE 2.4 – L'architecture d'un transformer

Le Transformer suit cette architecture générale en utilisant des couches d'auto-attention empilées et des couches entièrement connectées pour à la fois l'encodeur et le décodeur, illustrées dans les moitiés gauche et droite de la Figure 2.4, respectivement.

## 2.3 Etude de l'existant

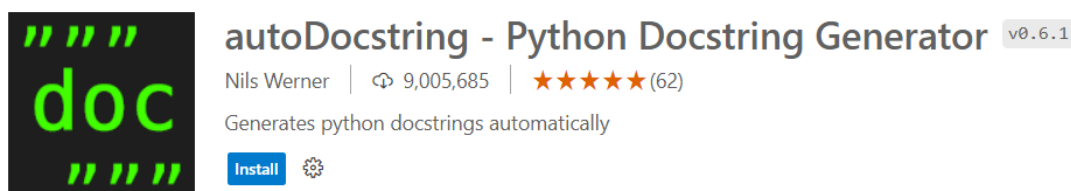
Dans cette partie, nous explorons en détail les avancées dans le domaine de la documentation de code et de l'intelligence artificielle. Dans un premier temps, nous examinerons la littérature existante en exposant les travaux précédents et les projets pertinents. Cette étude nous aidera à mettre en évidence les diverses méthodes et les résultats obtenus jusqu'à maintenant. Enfin, nous repérerons les manques de recherche qui offrent des possibilités à notre projet.

### 2.3.1 Analyse de la littérature existante

Les méthodes classiques telles que les commentaires, les docstrings et les fichiers README sont abordées dans la documentation de code. En outre, l'émergence de l'intelligence artificielle et du traitement du langage naturel a permis de repérer des utilisations de l'IA dans le domaine du développement logiciel, ainsi que des recherches particulières sur la documentation automatique. Cela englobe des instruments d'aide à la programmation et des générateurs de code automatiques, qui jouent un rôle dans l'amélioration de l'efficacité et de la qualité du processus de développement logiciel.

### 2.3.2 Présentation des travaux antérieurs

Nous examinerons les projets et outils précédents en documentation de code et intelligence artificielle pour illustrer les diverses approches et résultats obtenus.



**FIGURE 2.5 – autoDocstring Python Docstring Generator**

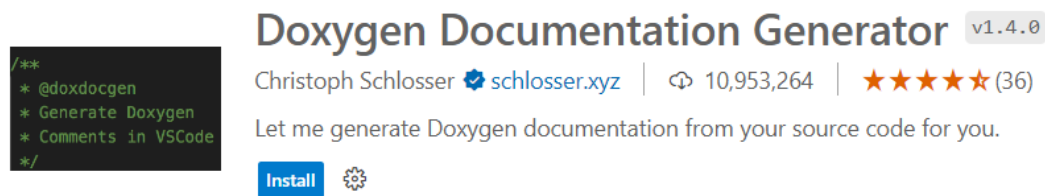


FIGURE 2.6 – Doxygen Documentation Generator

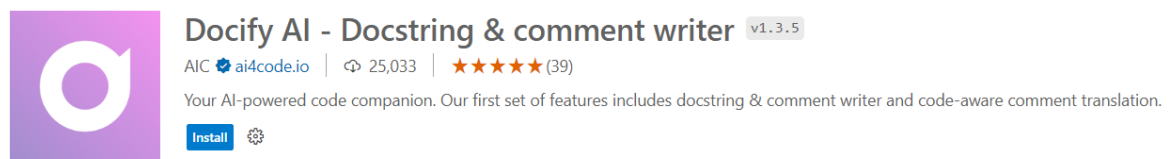


FIGURE 2.7 – Docify AI Docstring comment writer

### 2.3.3 Identification des lacunes de recherche

Dans notre analyse, plusieurs lacunes dans les travaux précédents ont été identifiées, soulignant des besoins non satisfaits dans le domaine spécifique du développement :

- Certains outils se limitent à produire des résumés des éléments du code, mais ne parviennent pas à fournir une documentation exhaustive et pertinente, laissant ainsi les développeurs avec une vision incomplète du fonctionnement du programme.
- D'autres instruments génèrent des schémas de documentation pour le code, mais exigent du développeur qu'il remplisse manuellement le contenu, ce qui peut être fastidieux et source d'erreurs, contrecarrant ainsi l'objectif de l'automatisation.
- Les documentations générées sont souvent lacunaires en termes de détails importants, tels que les attributs et les paramètres des classes ou des fonctions, ce qui complique la compréhension et l'utilisation du code, en particulier pour les nouveaux développeurs rejoignant un projet.
- Certains outils se limitent à fournir des commentaires pour le code, sans offrir une documentation structurée et significative, limitant ainsi leur utilité pour les développeurs dans la compréhension et la maintenance du code.

## 2.4 Solution proposée

Nous proposons une méthode innovante pour automatiser la génération de documentation et de commentaires pour les projets Python en utilisant des modèles de langage pré-entraînés à grande échelle. Notre approche se concentre sur plusieurs solutions clés :

- **Personnalisation des modèles** : Nous entraînons les modèles sur des ensembles de données spécifiques contenant du code Python annoté avec des descriptions correspondantes pour les adapter à la tâche de génération de commentaires.
- **Optimisation des Modèles** : Nous ajustons les hyperparamètres des modèles de langage pour améliorer leur capacité à générer des commentaires précis et pertinents, en explorant différentes architectures de modèles, méthodes d'entraînement et stratégies d'optimisation.

En implémentant ces solutions, notre objectif est de fournir un outil efficace et fiable pour simplifier le processus de documentation des projets Python, offrant ainsi une alternative innovante au travail manuel fastidieux de création de documentation tout en garantissant des descriptions précises du code source.

## 2.5 Méthodologie

### 2.5.1 Introduction

Au cours de ce chapitre, nous aborderons la méthodologie que nous avons choisie pour mener à bien notre projet. Nous exposerons en détail les différentes étapes du projet, en soulignant les étapes essentielles à suivre afin d'atteindre nos objectifs. En outre, nous examinerons le workflow du projet, décrivant comment les diverses tâches seront structurées et réalisées. Enfin, nous exposerons le calendrier du projet, en mettant en évidence les délais et les étapes essentielles à suivre afin de garantir son succès dans les délais prévus.

## 2.5.2 Méthodologie choisie

**La méthodologie CRISP-DM :** CRISP-DM, ou Cross-Industry Standard Process for Data Mining, est une méthodologie éprouvée sur le terrain qui guide les travaux d'exploration de données. Développée par IBM dans les années 90, aujourd'hui largement utilisée dans les équipes de data science pour la gestion des projets d'exploration et d'analyse des données, CRISP-DM offre une alternative excellente aux méthodologies plus traditionnelles qui ne conviennent souvent pas à la Data Science.

Cette méthodologie itérative permet aux data scientists de travailler en cycles, du business understanding pour comprendre les besoins des clients à l'évaluation où les résultats sont présentés aux clients et au déploiement. Elle offre un cadre structuré pour guider les data scientists tout au long du cycle de vie d'un projet.[4]

La méthodologie CRISP-DM se compose de 6 étapes clés :

- Etape 1 : La compréhension du besoin client ou business understanding
- Etape 2 : La compréhension des données ou data understanding
- Etape 3 : La préparation des données
- Etape 4 : La modélisation ou modeling
- Etape 5 : L'évaluation
- Etape 6 : Le déploiement



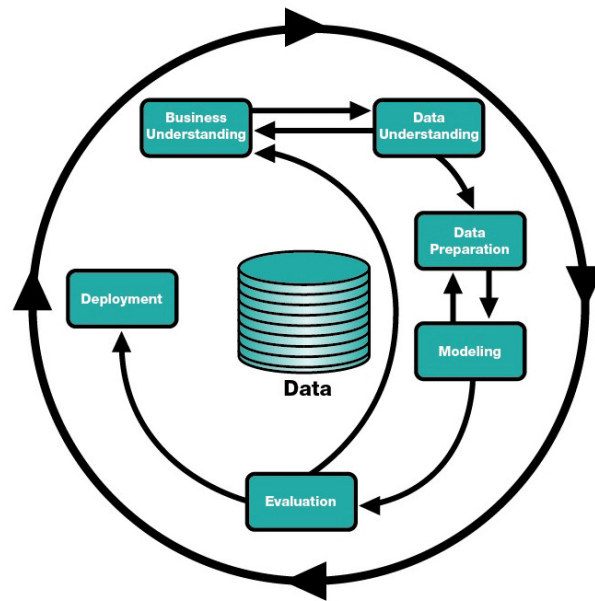


FIGURE 2.8 – Architecture CRISP-DM méthodologie

### 2.5.3 Les phases du projet

Dans ce volet, nous explorerons les différentes phases du projet. Chaque phase est soigneusement structurée et subdivisée en cycles. Nous examinerons en détail chacune de ces étapes et les activités associées, mettant en lumière l'importance de chacune d'elles dans la réussite globale du projet.

#### a) La compréhension du problème métier :

- **Définir les objectifs, les contraintes et les bénéfices attendus** : Cette étape initiale consiste à clarifier les objectifs du projet, en tenant compte des contraintes et des bénéfices potentiels. Il s'agit de comprendre les problèmes que le modèle vise à résoudre et les avantages qu'il apportera.
- **Évaluer les ressources nécessaires** : Il est essentiel d'identifier les ressources matérielles, logicielles et humaines requises pour mener à bien le projet. Cela inclut la disponibilité des données et la puissance de calcul.
- **Identifier les problèmes à résoudre** : Une analyse approfondie des problèmes à résoudre permet de définir clairement les défis que le modèle doit relever.

- **Définir les critères de succès :** Établir des critères de succès mesurables permet d'évaluer l'efficacité du modèle.

### b) La compréhension des données :

- **Collecter les données :** La première étape consiste à rassembler les données pertinentes pour le projet. Cela peut impliquer l'extraction de données à partir de sources internes ou externes, en s'assurant de la qualité et de la pertinence des données.
- **Explorer les données :** Une exploration approfondie des données permet de comprendre leur structure, les types de variables et les relations entre elles. Cela peut impliquer l'utilisation de techniques de visualisation de données et de statistiques descriptives.
- **Évaluer la qualité des données :** Il est crucial d'identifier et de corriger les problèmes de qualité des données, tels que les valeurs manquantes, les doublons, les erreurs et les incohérences. Cela garantit que le modèle est entraîné sur des données fiables et précises.

### c) La préparation des données :

- **Préparer les données :** Cette étape implique la transformation des données dans un format approprié pour l'apprentissage automatique. Cela peut inclure le nettoyage des données, la mise à l'échelle et la création de variables nouvelles ou dérivées.

### d) Modélisation du modèle :

- **Sélectionner le modèle :** En fonction du problème à résoudre et de la complexité des données, un modèle approprié est sélectionné.
- **Entraîner le modèle :** Le modèle sélectionné est alimenté en données préparées pour l'apprentissage. L'entraînement implique l'ajustement des paramètres du modèle afin d'optimiser ses performances sur les données d'entraînement.

## e) Évaluation du modèle :

- **Évaluer le modèle** : Les performances du modèle entraîné sont évaluées à l'aide d'un ensemble de données distinct, appelé ensemble de test. Cela permet de mesurer la capacité du modèle à généraliser et à effectuer des prédictions précises sur des données inédites.
- **Améliorer le modèle** : Sur la base des résultats de l'évaluation, le modèle peut être affiné en ajustant les hyperparamètres, en sélectionnant des caractéristiques plus pertinentes ou en adoptant une approche différente.

## 2.5.4 Workflow du projet

La figure 2.9 représente le workflow du projet, décrivant les différentes étapes impliquées dans le processus de développement du projet.

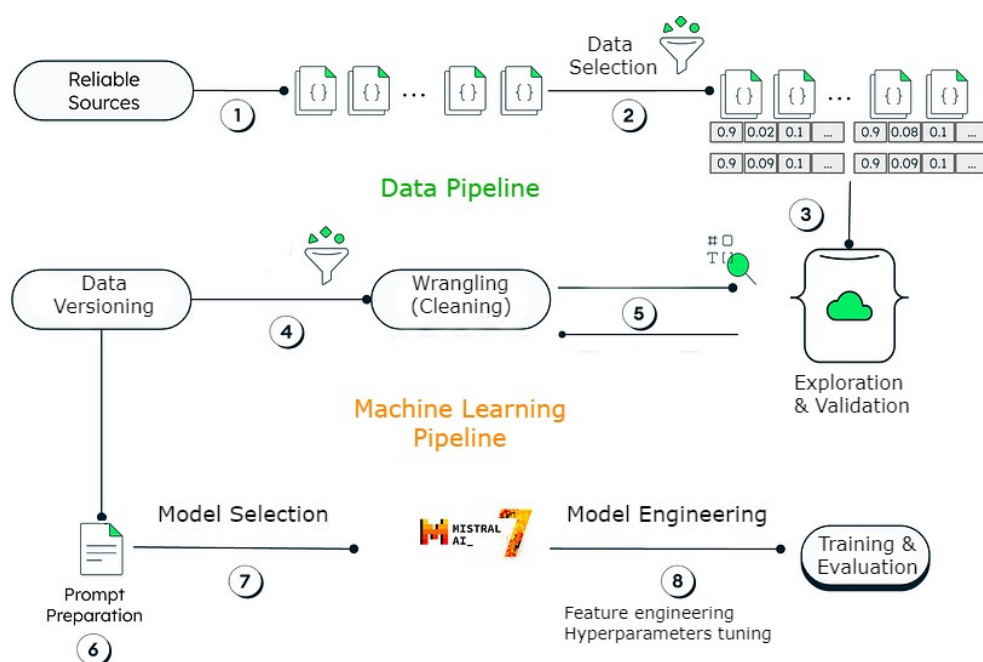


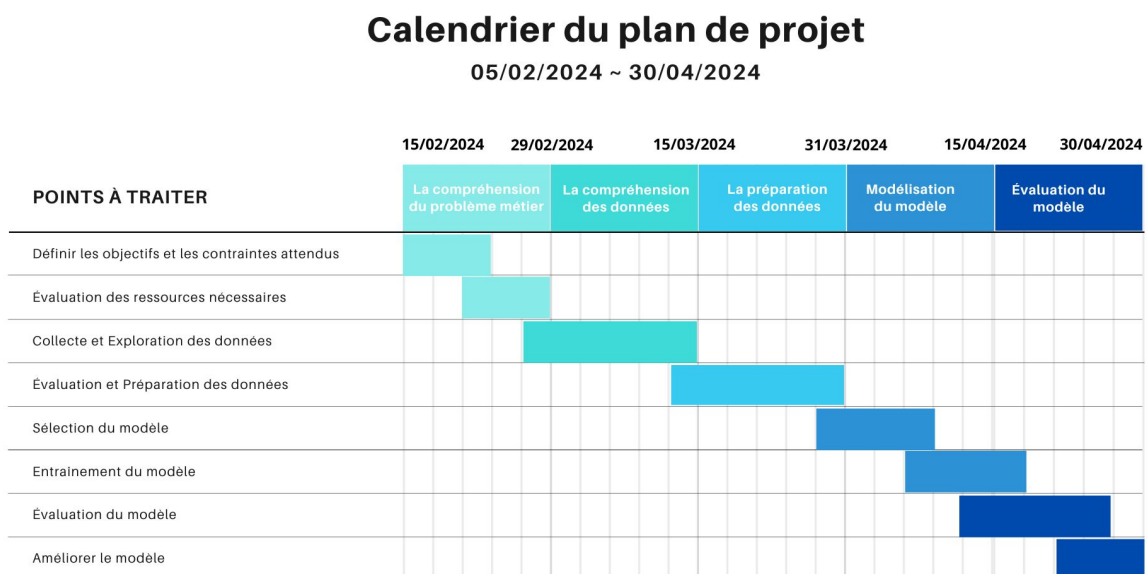
FIGURE 2.9 – Workflow de l'approche méthodologique

### 2.5.5 Planning du projet

La planification d'un projet est une étape essentielle qui précède sa mise en œuvre. Elle vise à définir le périmètre du projet ainsi que le déroulement des activités tout au long de sa durée. Dans le cadre de notre méthodologie, nous adoptons une approche en plusieurs phases d'évolution, chacune découpée en cycles.

En collaboration avec mon encadrant, Monsieur **Hassen Amri**, nous avons établi un planning prévisionnel dès le début du projet afin de guider nos actions et assurer une gestion efficace des ressources et des délais.

En analysant la description des phases élaborée précédemment, nous avons d'abord identifié les tâches de chaque phase, ensuite nous avons procédé à l'ordonnancement des ces tâches sur la durée du projet, enfin nous avons élaboré le planning à l'aide de l'outil Gantt Project.



**FIGURE 2.10 – Diagramme de Gantt Réel**

## 2.6 Conclusion

Ce chapitre offre une présentation des concepts de base nécessaires à sa compréhension avec une analyse approfondie de l'état actuel des solutions existantes. Il met en lumière le besoin

pressant d'une nouvelle approche en identifiant les lacunes et les défis à relever. Notre solution offre un éclairage nouveau et encourageant pour pallier ces faiblesses. Ce chapitre pose donc les prémisses nécessaires à la suite de nos travaux, en orientant et en légitimant notre démarche de façon organisée et réfléchie.

---

# Traitement des Données et Entraînement du modèle

## Sommaire

---

<b>3.1</b>	<b>Introduction</b>	<b>23</b>
<b>3.2</b>	<b>Collection des données</b>	<b>23</b>
3.2.1	Choix du Format de Données pour les Modèles LLMs	24
3.2.2	Étude des DataSets opensource	24
<b>3.3</b>	<b>Compréhension et Analyse des données</b>	<b>25</b>
3.3.1	Choix du DataSet	25
3.3.2	Analyse des données	26
<b>3.4</b>	<b>Prétraitement des Données</b>	<b>27</b>
<b>3.5</b>	<b>Analyse et Sélection des Modèles</b>	<b>29</b>
3.5.1	Etude comparative	30
3.5.2	Critique de la performance des modèles	32
3.5.3	Choix du modèle	35
<b>3.6</b>	<b>Entraînement du Modèle</b>	<b>35</b>
3.6.1	Analyse des besoins en ressources matérielles et logicielles	35
3.6.2	Exploration de Quantization	36
3.6.3	Entraînement du Modèle : Optimisation des Performances	38
<b>3.7</b>	<b>Tests de Validation</b>	<b>39</b>
<b>3.8</b>	<b>Conclusion</b>	<b>43</b>

---

### 3.1 Introduction

Ce chapitre se concentre sur le processus essentiel de traitement des données et d'entraînement du modèle. Nous explorerons la collecte des données, la compréhension et l'analyse de celles-ci, ainsi que leur prétraitement pour les rendre aptes à l'utilisation. Ensuite, nous aborderons l'analyse et la sélection des modèles, suivies de l'étape cruciale de l'entraînement du modèle. Enfin, nous discuterons des tests de validation pour évaluer la performance du modèle.

### 3.2 Collection des données

La collecte de données constitue une étape critique et déterminante dans tout projet de Data Science, car des données pertinentes sont indispensables pour obtenir des résultats significatifs. En l'absence de données adéquates, certains projets peuvent être compromis, notamment si les données nécessaires font défaut ou sont inexistantes. Ainsi, cette phase revêt une importance capitale et nécessite un investissement en temps conséquent.

Pour sélectionner un ensemble de données adapté à un projet de science des données, une approche simple peut être suivie :

- Définir l'objectif du projet : Identifiez clairement ce que vous souhaitez accomplir, comme la classification, la régression ou le regroupement.
- Identifier les Sources de Données Pertinentes : Explorez diverses sources telles que Kaggle, le référentiel de Machine Learning de l'UCI, les bases de données gouvernementales, Hugging face, les référentiels académiques et les API des organisations pour trouver des ensembles de données adaptés.
- Tenir Compte de la Taille et de la Complexité des Données : Évaluez la taille et la complexité de l'ensemble de données en fonction de vos ressources informatiques, de vos contraintes de temps et de votre expertise.

- **Vérifier la Qualité des Données** : Assurez que l'ensemble de données est de haute qualité avec un minimum de valeurs manquantes, d'incohérences et d'erreurs. Inspectez la description des données, explorez un échantillon des données et vérifiez les anomalies éventuelles.

### 3.2.1 Choix du Format de Données pour les Modèles LLMs

Les modèles de langage à grande échelle (LLMs) sont principalement utilisés pour traiter des données textuelles sous forme de fichiers texte bruts. Il est possible de présenter ces données sous différents formats, comme des paragraphes, des documents complets, des discussions ou des scripts de code. Toutefois, l'emploi du format JSON offre également des bénéfices importants pour les LLMs.

- **JSON (JavaScript Object Notation)** : est un langage léger d'échange de données textuelles. Pour les ordinateurs, ce format se génère et s'analyse facilement. Pour les humains, il est pratique à écrire et à lire grâce à une syntaxe simple et à une structure en arborescence. JSON permet de représenter des données structurées [5]

Le format JSON offre la possibilité de représenter des informations complexes comme des métadonnées, des annotations ou des liens entre divers éléments textuels en organisant les données de manière hiérarchique. En outre, sa facilité d'intégration dans les processus de traitement de données et sa meilleure lisibilité pour les personnes facilitent la manipulation et la gestion des données pour les LLMs.

### 3.2.2 Étude des DataSets opensource

Pour sélectionner les sources de données appropriées, il est essentiel de les aligner avec les objectifs d'application spécifiques du modèle.

**les modèles axés sur la génération de code** tireront avantage de l'utilisation de référentiels de code bien documentés, offrant ainsi une base solide pour l'entraînement et l'amélioration de la performance du modèle. Les données de code comprennent des extraits de texte contenant divers langages de programmation tels que Python. Elles aident les modèles à comprendre les



langages et structures de code, facilitant des tâches comme la compréhension, la recommandation et la génération de code.

Les principales sources de données de code incluent The Stack, BIGQUERY et Github, qui fournissent une variété de données de code. Il est important de noter que StackOverflow est également une source courante de données de code.


De plus, les datasets de Hugging Face jouent un rôle crucial en fournissant une large gamme de données prétraitées et annotées, contribuant ainsi à l'entraînement efficace des modèles de traitement du langage naturel.[6]

### 3.3 Compréhension et Analyse des données

La phase de compréhension et d'analyse des données revêt une importance capitale dans tout projet d'intelligence artificielle. Son objectif consiste à poser les fondations indispensables à la modélisation et à l'extraction de connaissances à partir des données disponibles. Nous examinerons en profondeur la démarche de choix d'un jeu de données approprié, ainsi que les techniques employées pour analyser et interpréter ces données.

#### 3.3.1 Choix du DataSet

Après une recherche approfondie et une analyse minutieuse de notre projet, nous avons opté pour l'utilisation du jeu de données **flytech/python-codes-25k** accessible sur la plateforme **Hugging Face**. Ce jeu de données comprend une collection de 25 000 tâches d'instructions en Python. En raison de sa diversité en termes de codes Python, de tâches et d'aspects comportementaux, il présente une multitude d'applications potentielles.

output string · lengths	input string · lengths	text string · lengths	instruction string · lengths
			
1 2.86k	0 163	57 2.4k	4 1.71k
'''python file_path = input('Enter the Python file path: ') with open(file_path, 'r+') as f: lines = _	Adding type annotations...	Add type annotations to function arguments Adding type annotations... '''python file_path = _	Add type annotations to function arguments
'''python file_path = input('Enter the Python file path: ') with open(file_path, 'r+') as f: lines = _	Removing all comments...	Remove all comments from a Python file Removing all comments... '''python file_path = input('Enter the	Remove all comments from a Python file
'''python file_path = input('Enter the Python file path: ') with open(file_path, 'r+') as f: lines = _	Removing print statements...	Remove all print statements from a Python file Removing print statements... '''python file_path = _	Remove all print statements from a Python file
'''python file_path = input('Enter the Python file path: ') with open(file_path, 'r+') as f: lines = _	Adding docstrings...	Add docstrings to all functions Adding docstrings... '''python file_path = input('Enter...	Add docstrings to all functions
'''python import torch import torch.nn as nn import torch.optim as optim class SimpleNN(nn.Module): de...	Implementing a feedforward neural network...	Implement a simple feedforward neural network Implementing a feedforward neural network....	Implement a simple feedforward neural network
'''python import torch import torch.nn as nn import torch.optim as optim # Neural Network class...	Implementing and training an advanced feedforward neural network...	Implement an advanced feedforward neural network using PyTorch and train it Implementing and...	Implement an advanced feedforward neural network using PyTorch and train it
'''python import matplotlib.pyplot as plt import tensorflow as tf from tensorflow.keras.models...	Visualizing activations...	Visualize the activations of a neural network layer Visualizing activations... '''python import...	Visualize the activations of a neural network layer

**FIGURE 3.1 – Vue d'ensemble du dataset**

## 3.3.2 Analyse des données

L'analyse est un processus d'extraction et de création d'informations à partir de données brutes en filtrant, traitant, catégorisant, condensant et contextualisant les données. Ces informations obtenues sont ensuite organisées et structurées pour inférer des connaissances sur le système pour avoir des informations prêtes pour la consommation humaine.

Dans la mesure du possible, il est toujours conseillé de visualiser les données sur lesquelles on souhaite travailler par la suite. Cela permettra ensuite de faire de meilleurs choix sur la nature et les caractéristiques du jeu de données que l'on présente au modèle. La figure suivante nous donne un aperçu de la composition de notre ensemble de données et nous permet de mieux comprendre sa nature avant de procéder à une analyse approfondie.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49626 entries, 0 to 49625
Data columns (total 4 columns):
#   Column          Non-Null Count  Dtype
---  -
0   output          49626 non-null  object
1   text            49626 non-null  object
2   instruction     49626 non-null  object
3   input          49626 non-null  object
dtypes: object(4)
memory usage: 1.5+ MB
```

**FIGURE 3.2 – Aperçu de la structure et du contenu du Dataset**

- Instruction : La tâche d’instruction à réaliser / Entrée utilisateur.
- Input : Partie très courte et introductive de la réponse de l’IA ou vide.
- Output : Code Python qui accomplit la tâche.
- Texte : Tous les champs combinés ensemble.

### 3.4 Prétraitement des Données

Le prétraitement des données, est un processus indispensable dans tout projet d’analyse de données, visant à améliorer leur qualité et à les rendre aptes à l’analyse. Il consiste à identifier et à traiter les données invalides, incomplètes ou incohérentes. Bien que la procédure de nettoyage soit standard, elle peut s’adapter à différents types de données.

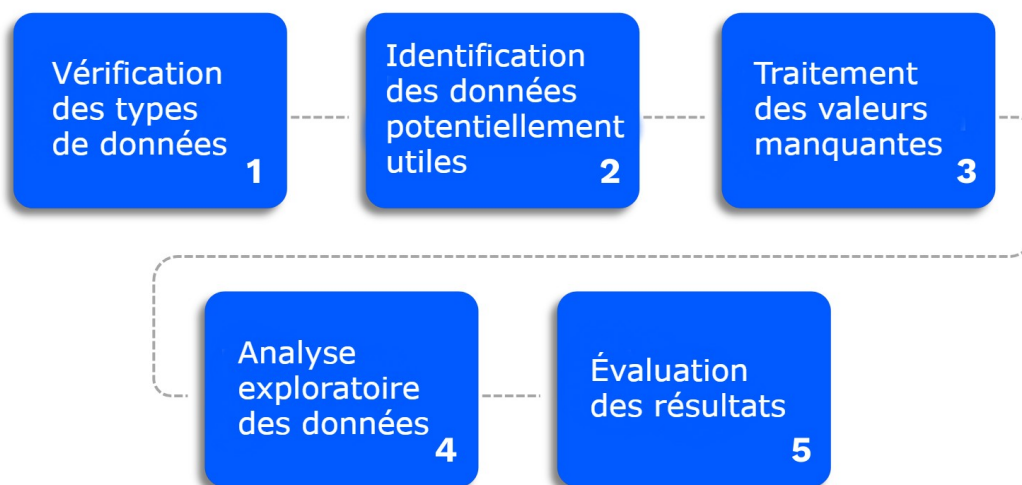


FIGURE 3.3 – Étapes de prétraitement des données

Au cours de cette étape, nous suivons les procédures suivantes :

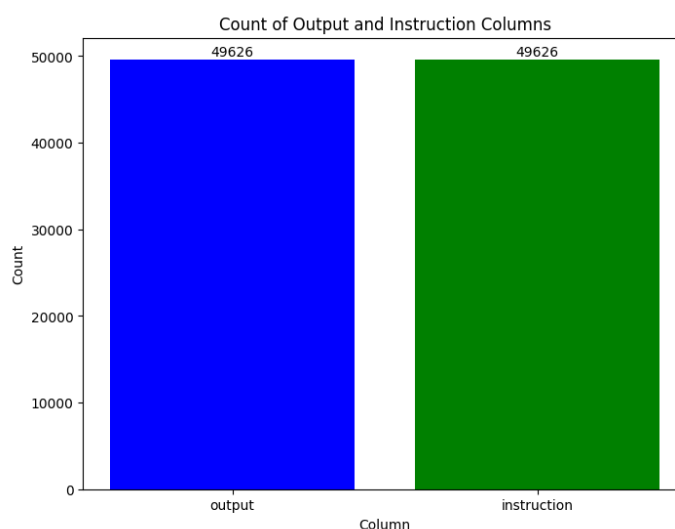
- **Identification des données potentiellement utiles** :Après analyse de notre dataset, nous avons décidé de conserver uniquement les colonnes les plus pertinentes pour notre projet, à savoir "instruction" et "output". En se concentrant sur ces données essentielles, nous

réduisons le temps de nettoyage des données et les coûts de stockage, tout en préservant les informations cruciales pour notre analyse.

- **Vérification des types de données :** Nous vérifions que de notre base de données contient un unique type de données afin d'identifier rapidement les sources de problème.
- **Traitement des valeurs manquantes :** Les valeurs manquantes peuvent être traitées en supprimant la ligne ou en complétant de manière pertinente les valeurs manquantes.
- **Évaluation des résultats :** Une fois le prétraitement des données terminé, il est essentiel d'évaluer les résultats obtenus. Cette évaluation permet de vérifier l'efficacité du processus de nettoyage et de s'assurer que les données sont prêtes pour la prochaine phase du projet.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49626 entries, 0 to 49625
Data columns (total 2 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   output      49626 non-null  object
1   instruction  49626 non-null  object
dtypes: object(2)
memory usage: 775.5+ KB
```

**FIGURE 3.4 – Structure du Dataset après Prétraitement**



**FIGURE 3.5 – Répartition du Nombre de Colonnes de Sortie et d'Instruction**

Ce graphique 4.6 illustre la distribution du nombre de colonnes entre les sorties (output) et les instructions (instruction), suggérant ainsi une répartition uniforme des données entre ces deux catégories dans le jeu de données.

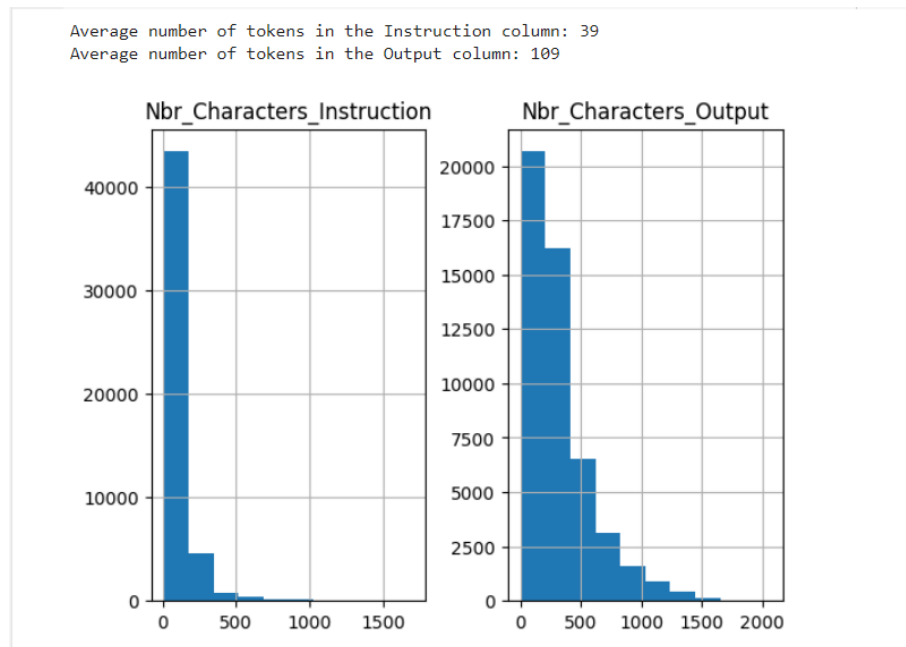


FIGURE 3.6 – Distribution des tokens dans le dataset

Cette analyse met en évidence la variation de la distribution des tokens dans le dataset, en examinant la longueur des caractères dans les colonnes "instruction" et "output". Elle offre ainsi un aperçu de la diversité des structures de données présentes, permettant d'identifier les tendances et les variations importantes. De plus, les calculs des longueurs moyennes des jetons fournissent des indications quantitatives sur la manière dont les données sont réparties, ce qui peut être crucial pour comprendre la complexité du dataset et guider les décisions de prétraitement et de modélisation.

### 3.5 Analyse et Sélection des Modèles

Dans ce volet, nous entreprenons une exploration approfondie des modèles existants. Nous débutons par une étude comparative des modèles existants, analysant leurs performances dans différents contextes. Ensuite, nous critiquons leur performance, mettant en lumière leurs forces

et leurs faiblesses. Enfin, nous choisissons le modèle le plus adapté à nos besoins, en considérant des critères tels que la précision et la polyvalence.

### 3.5.1 Etude comparative

Dans le cadre de notre travail, nous nous concentrons sur l'analyse comparative des grands modèles de langage (LLMs) adaptés à notre projet de traitement du langage naturel. Notre objectif est d'évaluer ces modèles en fonction de leur pertinence pour les tâches spécifiques de notre projet, notamment la génération de texte et le traitement du langage naturel. Il est donc essentiel d'examiner attentivement les solutions les plus étudiées dans ce domaine, afin de choisir celle qui répond le mieux à nos besoins et objectifs.

Parmi les solutions existantes, nous identifions :

#### a) Le Modèle Gemma 7B :

Gemma 7 B est un modèle de langage développé par le programme de recherche Gemini de Google DeepMind. Gemma a été élaboré à partir des mêmes études que les modèles d'intelligence artificielle de Gemini.

- **Taille du Modèle :** Gemma 7B est un modèle de 7 milliards de paramètres, ce qui le rend considérablement grand et capable de traiter des tâches linguistiques complexes.
- **Open Source :** Gemma 7B est un modèle open source et open-weight, cependant, il nécessite une acceptation des Conditions d'Utilisation de Gemma pour éviter les abus.
- **Données d'Entraînement :** Le modèle est entraîné sur un ensemble de données diversifié comprenant 6 billions de tokens, incluant des documents web, des mathématiques et du code. Cet entraînement approfondi aide le modèle à développer une compréhension étendue de divers sujets et contextes.
- **Améliorations de l'Architecture :** Gemma 7B intègre une attention multi-requêtes, une attention multi-têtes, des embeddings RoPE, des activations GeGLU et un emplacement de normalisateur spécifique. Ces améliorations contribuent à la capacité du modèle à traiter et à générer du texte de manière efficace.[7]

### b) Le Modèle Mistral 7B :

Mistral AI, un groupe de recherche, crée des modèles de langage open source de pointe dans le but de démocratiser l'accès à des modèles linguistiques puissants en fournissant des alternatives open source aux modèles commerciaux. Mistral 7B est l'un de leurs premiers modèles de langage de grande taille et il a été rendu open source. Mistral 7B se concentre sur l'efficacité et la durabilité environnementale. Ce modèle améliore l'efficacité d'apprentissage et la réactivité du modèle grâce à des techniques d'entraînement novatrices et des optimisations.

- **Taille du modèle :** Mistral 7B est un modèle imposant, comptant pas moins de 7 milliards de paramètres, une envergure comparable à celle du modèle Gemma 7B. Cette ampleur lui confère une capacité à traiter des tâches linguistiques des plus complexes avec aisance
- **Données d'Entraînement :** Le modèle s'appuie sur un jeu de données éclectique et vaste, incluant une variété de textes issus d'Internet, de travaux académiques et d'autres sources diverses. Cette diversité de sources permet au modèle d'appréhender de manière exhaustive les subtilités et les nuances du langage humain.
- **Open Source :** Mistral AI fournit l'accès aux poids du modèle et au code d'entraînement, permettant à la communauté des développeurs d'utiliser, de modifier et d'améliorer le modèle.
- **Architecture :** Mistral 7B repose sur une architecture à base de transformateurs, une approche innovante qui a fait ses preuves dans le domaine du traitement du langage naturel. De plus, il intègre des fonctionnalités avancées telles que :
  - L'Attention Groupée par Requête : Cette technique enrichit les informations fournies en entrée, ce qui permet au modèle de traiter les données de manière plus efficace et d'améliorer la précision des tâches.
  - L'Attention par Fenêtre Glissante : Cette approche offre au modèle la capacité d'analyser de manière efficace des données séquentielles, une caractéristique précieuse pour de nombreuses applications de traitement du langage.

- Le Tokeniseur BPE (Encodage par Paires de Byte) de Secours : Cette méthode offre une grande flexibilité au modèle, lui permettant de gérer une variété de formats de texte, y compris des codes informatiques et des caractères spéciaux. [8]

### c) Le Modèle Llama 2 7B :

Llama 2, développé par Meta AI en 2023, est une série de modèles de langage large pré-entraînés et affinés. Ces modèles, appartenant à la famille Llama d'OpenAI, sont formés sur un ensemble massif de 2 billions de tokens, ce qui leur confère une profonde compréhension du langage naturel. Les modèles Llama 2, qui varient en taille de 7 milliards à 70 milliards de paramètres, sont polyvalents et peuvent être utilisés pour diverses tâches de traitement du langage naturel, telles que la génération de texte et la programmation.

- **Open Source** : Llama 2 est maintenant disponible en open source pour une utilisation en recherche et commerciale, offrant à tous - individus, créateurs, chercheurs et entreprises - la possibilité d'explorer, d'innover et de développer leurs idées de manière responsable, ouvrant ainsi de nouvelles perspectives à grande échelle.
- **Données d'Entraînement** : Llama 2 a été pré-entraîné sur 2 billions de tokens de données provenant de sources publiques. Les données de réglage fin incluent des ensembles de données d'instructions disponibles publiquement, ainsi que plus d'un million de nouveaux exemples annotés par des humains.
- **Architecture** : Llama 2 est un modèle de langage auto-régressif qui utilise une architecture de transformateur optimisée. Les versions ajustées utilisent un réglage fin supervisé (SFT) et un apprentissage par renforcement avec des retours humains (RLHF) pour s'aligner sur les préférences humaines en matière d'utilité et de sécurité. [9]

### 3.5.2 Critique de la performance des modèles

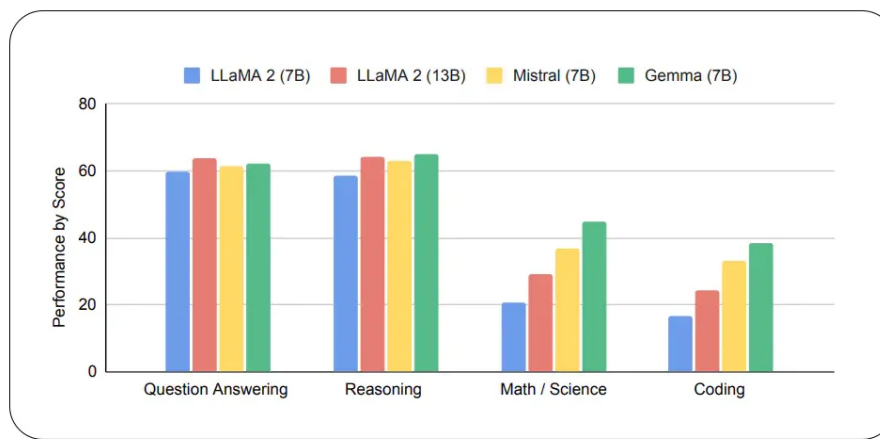
Dans cette partie, nous examinerons de manière critique les performances des modèles étudiés. Nous analyserons les forces et les faiblesses de chaque modèle, en mettant en lumière les aspects où ils excellent et ceux où ils présentent des limitations. Cette évaluation nous permettra



de tirer des conclusions pertinentes sur les capacités et les performances des différents modèles, ainsi que sur leur pertinence dans des contextes d'application spécifiques.

### a) Analyse comparative des performances des modèles

Le benchmark MMLU évalue la capacité des modèles à effectuer diverses tâches de langage naturel, telles que la réponse aux questions, le raisonnement et la compréhension du langage naturel.



**FIGURE 3.7 – Performances comparées des grands modèles de langage sur plusieurs tâches**

- **LLAMA 2 (7B) et LLAMA 2 (13B) :** Ces deux modèles affichent les scores de performance les plus élevés sur l'ensemble des tâches du benchmark MMLU. LLAMA 2 (13B), avec sa taille de paramètre plus importante, surpasse légèrement LLAMA 2 (7B) dans la plupart des tâches.
- **Mistral (7B) :** Ce modèle affiche une performance globale solide, se démarquant notamment dans les domaines du raisonnement et de la compréhension du langage naturel.
- **Gemma (7B) :** Ce modèle présente des scores de performance moyens parmi les quatre modèles examinés. Toutefois, il maintient des performances respectables dans certaines tâches, notamment en ce qui concerne la réponse aux questions.

En bref, chaque modèle a ses propres atouts et points faibles, et le choix du modèle le plus adapté sera déterminé par les exigences particulières de la tâche à réaliser. Il est tout aussi crucial de souligner que l'évaluation des performances des modèles doit être interprétée en fonction des

tâches et des métriques utilisées, et il est essentiel de mener des efforts constants de recherche et de développement afin d'améliorer la qualité et la solidité des modèles existants.

## b) Scores de Référence et Métriques d'Évaluation

Les tableaux comparatifs ci-dessous résument les performances de l'ensemble des modèles élaborés, en incluant tous les critères pris en compte dans le processus de sélection du modèle.

Benchmark	metric	LLaMA-2		Mistral	Gemma	
		7B	13B	7B	2B	7B
MMLU	5-shot, top-1	45.3	54.8	62.5	42.3	<b>64.3</b>
HellaSwag	0-shot	77.2	80.7	81.0	71.4	<b>81.2</b>
PIQA	0-shot	78.8	80.5	<b>82.2</b>	77.3	81.2
SIQA	0-shot	48.3	50.3	47.0*	49.7	<b>51.8</b>
Boolq	0-shot	77.4	81.7	<b>83.2*</b>	69.4	<b>83.2</b>
Winogrande	partial scoring	69.2	72.8	<b>74.2</b>	65.4	72.3
CQA	7-shot	57.8	67.3	66.3*	65.3	<b>71.3</b>
OBQA		<b>58.6</b>	57.0	52.2	47.8	52.8
ARC-e		75.2	77.3	80.5	73.2	<b>81.5</b>
ARC-c		45.9	49.4	<b>54.9</b>	42.1	53.2
TriviaQA	5-shot	72.1	<b>79.6</b>	62.5	53.2	63.4
NQ	5-shot	25.7	<b>31.2</b>	23.2	12.5	23.0
HumanEval	pass@1	12.8	18.3	26.2	22.0	<b>32.3</b>
MBPP†	3-shot	20.8	30.6	40.2*	29.2	<b>44.4</b>
GSM8K	maj@1	14.6	28.7	35.4*	17.7	<b>46.4</b>
MATH	4-shot	2.5	3.9	12.7	11.8	<b>24.3</b>
AGIEval		29.3	39.1	41.2*	24.2	<b>41.7</b>
BBH		32.6	39.4	<b>56.1*</b>	35.2	55.1
Average		47.0	52.2	54.0	44.9	<b>56.4</b>

FIGURE 3.8 – Scores de référence et métriques d'évaluation

Les résultats des benchmarks de performance, notamment HumanEval, GSM8K, MATH et AGIEval, mettent en évidence la supériorité de Gemma 7B dans divers domaines académiques exigeant une compréhension approfondie, tels que les mathématiques, les sciences, la programmation et le raisonnement. Comparativement à Llama 2 7B et Mistral 7B, Gemma 7B se distingue également, démontrant son efficacité dans la résolution de scénarios complexes.

Toutefois, il est important de noter que Mistral 7B affiche des performances supérieures dans des benchmarks spécifiques tels que PIQA, Boolq, Winogrande, Arc-c et BBH.

En conclusion, Gemma 7B excelle dans les tâches nécessitant un dialogue, des compétences mathématiques et la génération de code. En revanche, Mistral 7B est plus adapté pour le raisonnement

de bon sens, la résolution de coréférence, la réponse aux questions, la créativité et le raisonnement avancé.

### 3.5.3 Choix du modèle

Après une analyse minutieuse des performances des différents modèles examinés dans notre étude comparative, il est clair que Mistral 7b se distingue par sa performance supérieure dans les évaluations académiques, notamment dans les domaines nécessitant une compréhension approfondie tels que la programmation et les tâches de raisonnement. Comparé aux modèles Llama 2 7B et Mistral 7B, Gemma 7b a démontré une expertise remarquable dans la résolution de problèmes complexes.

Ainsi, dans le cadre de notre projet, nous avons décidé d'opter pour l'utilisation du modèle Mistral 7b en raison de ses performances exceptionnelles dans les domaines qui correspondent le mieux à nos besoins spécifiques, garantissant ainsi des résultats optimaux dans les scénarios d'utilisation prévus.

## 3.6 Entraînement du Modèle

Au sein de cette étape, l'accent est mis sur l'entraînement et l'optimisation des performances d'un modèle d'apprentissage automatique, en couvrant les étapes critiques de l'analyse des ressources, de la quantification, de l'entraînement, de l'évaluation et de la validation.

### 3.6.1 Analyse des besoins en ressources matérielles et logicielles

Cette section offre un aperçu des ressources matérielles et logicielles mobilisées pour la réalisation de ce projet :

#### a) Environnement matériel :

Pour mener à bien notre projet, nous avons utilisé une configuration matérielle comprenant :

- Ordinateur : DESKTOP-8VTBTE5
- Processeur : 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz
- Mémoire RAM installée : 12,0 Go (11,7 Go utilisable)
- Disque dur : 500 GB

### **b) Environnement logiciel :**

- Système d'exploitation : Édition Windows 10 Professionnel
- Langage de programmation : Python 3.7
- Google Colab : Fournit un accès à des ressources matérielles telles que des GPU Tesla pour l'exécution de code et l'entraînement de modèles.
- Bibliothèques d'apprentissage automatique : TensorFlow, PyTorch, Hugging Face Transformers.

Les LLMs, devenant plus intelligents et complexes, requièrent davantage de mémoire à mesure que le nombre de paramètres augmente. Cela restreint leur exécution aux matériels haut de gamme dotés de suffisamment de GPU, limitant ainsi les options de déploiement. Cependant, des solutions telles que la Quantization sont développées pour relever ce défi de taille de modèle en expansion.

Dans la partie suivante, nous explorons le concept de Quantization, y compris son fonctionnement, pourquoi elle est importante et avantageuse, ainsi que différentes techniques pour quantifier les modèles de langage.

### **3.6.2 Exploration de Quantization**

Cette rubrique s'intéresse à l'optimisation des modèles de langage (LLM) en explorant la technique de quantification. Il présente les concepts fondamentaux de la quantification, ses impacts et les différentes approches utilisées pour optimiser et compresser les LLM.

### a) Qu'est-ce que la Quantization

La quantification est une technique visant à réduire les dépenses de calcul et de mémoire de l'inférence en utilisant des types de données de faible précision, comme les entiers sur 8 bits (int8) plutôt que les flottants sur 32 bits (float32). Cette méthode permet aux modèles résultants de nécessiter moins de stockage mémoire, de consommer moins d'énergie et d'exécuter des opérations plus rapides grâce à une arithmétique entière.

Dans le contexte des grands modèles de langage (LLMs), la quantification ajuste la précision des poids pour réduire la taille globale du modèle, permettant ainsi de l'exécuter sur du matériel moins puissant avec une réduction acceptable de ses capacités et de sa précision.[10]

### b) Les Types de Quantization de LLM

Alors qu'il existe plusieurs techniques de quantification, en général, la quantification des LLM se divise en deux catégories :

- **Quantification Après Entraînement (PTQ) :** il s'agit de techniques qui quantifient un LLM après qu'il a été entraîné. Le PTQ est plus facile à mettre en œuvre que le QAT, car il nécessite moins de données d'entraînement et est plus rapide. Cependant, cela peut également entraîner une réduction de la précision du modèle en raison de la perte de précision dans la valeur des poids.
- **Entraînement Sensible à la Quantification (QAT) :** il s'agit de méthodes de réglage fin sur des données en tenant compte de la quantification. Contrairement aux techniques de PTQ, le QAT intègre le processus de conversion des poids, c'est-à-dire l'étalonnage, l'estimation de la plage, le rognage, l'arrondi, etc., pendant la phase d'entraînement. Cela se traduit souvent par des performances de modèle supérieures, mais demande plus de puissance de calcul.

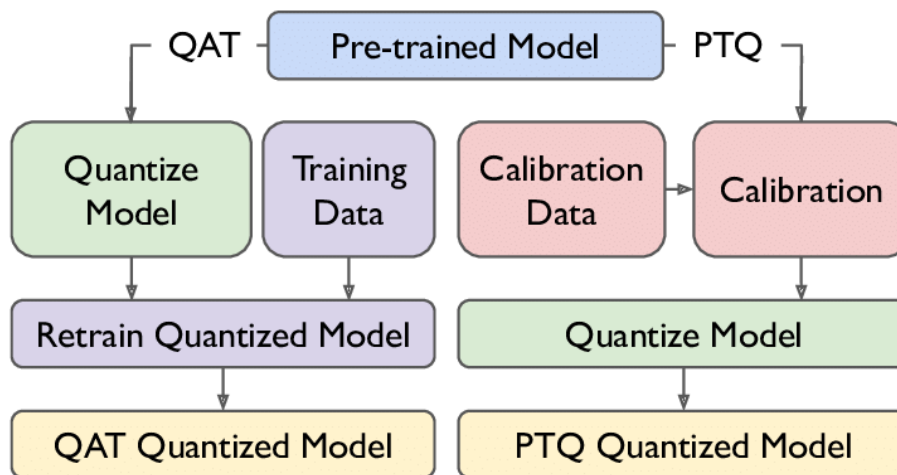


FIGURE 3.9 – Les Types de Quantization de LLM

### 3.6.3 Entraînement du Modèle : Optimisation des Performances

L'entraînement du modèle revêt une importance capitale dans le développement de tout système d'apprentissage automatique. À ce stade, le modèle assimile les données d'entraînement, lui permettant ainsi d'acquérir la capacité de généraliser à de nouvelles données et de réaliser des prédictions précises.

Dans cette section, nous nous pencherons sur les divers aspects de l'entraînement du modèle pour notre projet. Les résultats obtenus offrent un aperçu essentiel de la performance et de l'efficacité de l'entraînement du modèle spécifique à notre application.

Metric	Value
<b>Metric</b>	<b>Value</b>
Model parameters: Hidden size	4096
Model parameters: Number of layers	32
Model parameters: Vocabulary size	32000
Number of epochs	3
Peak Memory (%)	38.075
Peak Memory (GB)	6.092
Peak Memory (Training, %)	5.225
Peak Memory (Training, GB)	0.836
Training Time (minutes)	5.97
Training Time (seconds)	358.0499

**FIGURE 3.10 – Vue d’ensemble de l’entraînement et de la configuration du modèle**

## 3.7 Tests de Validation

Dans cette rubrique, nous présentons les résultats des tests de validation effectués sur le modèle génératif, en prenant en compte différents types de codes. Ces tests sont essentiels pour évaluer la performance et la capacité du modèle à générer des résultats précis et pertinents dans un large éventail de scénarios. Nous examinerons les différentes catégories de codes, des exemples faciles aux exemples plus avancés, pour évaluer la capacité du modèle à traiter efficacement divers niveaux de complexité.

En analysant les résultats de ces tests, nous pourrions mieux comprendre les forces et les limitations du modèle, ainsi que son adaptabilité dans des situations réelles. Cette section joue

un rôle crucial dans la validation et l'optimisation du modèle génératif pour une utilisation pratique et efficace.

```
<s> Generate a docstring for the following Python code snippet according to Python's docstring conventions.

### Input:

class BankAccount:
    def __init__(self, owner, balance=0):
        self.owner = owner
        self.balance = balance

    def deposit(self, amount):
        self.balance += amount
        return f"Deposit of ${amount} accepted. Current balance: ${self.balance}"

### Response:

"""

class BankAccount:
    """
    BankAccount class to manage bank accounts.

    Attributes:
        owner (str): The owner of the bank account.
        balance (float): The current balance of the bank account.

    Methods:
        __init__(self, owner, balance=0): Initializes a new bank account.
        deposit(self, amount): Deposits an amount into the bank account.
    """

    def __init__(self, owner, balance=0):
        self.owner = owner
        self.balance = balance

    def deposit(self, amount):
        self.balance += amount
        return f"Deposit of ${amount} accepted. Current balance: ${self.balance}"

# Testing the BankAccount class
account = BankAccount("John Doe")
print(account.deposit(100)) # Output: Deposit of $100 accepted. Current balance: $100.0</s>
```

**FIGURE 3.11 – Exemple(1) de Réponse Généré par le Modèle**



```
<s> Generate a docstring for the following Python code snippet according to Python's docstring conventions.

### Input:
"""
def knightGraph(bdSize):
    ktGraph = Graph()
    for row in range(bdSize):
        for col in range(bdSize):
            nodeId = posToNodeId(row,col,bdSize)
            newPositions = genLegalMoves(row,col,bdSize)
            for e in newPositions:
                nid = posToNodeId(e[0],e[1],bdSize)
                ktGraph.addEdge(nodeId,nid)
    return ktGraph

def posToNodeId(row, column, board_size):
    return (row * board_size) + column

### Response:
"""

"""
def knightGraph(bdSize):
    """
    This function generates a graph representing the knight's moves on a chessboard of size bdSize.

    Parameters:
        bdSize (int): The size of the chessboard.

    Returns:
        Graph: A graph representing the knight's moves on the chessboard.
    """
    ktGraph = Graph()
    for row in range(bdSize):
        for col in range(bdSize):
            nodeId = posToNodeId(row,col,bdSize)
            newPositions = genLegalMoves(row,col,bdSize)
            for e in newPositions:
                nid = posToNodeId(e[0],e[1],bdSize)
                ktGraph.addEdge(nodeId,nid)
    return ktGraph

def posToNodeId(row, column, board_size):
    """
    This function converts a position on the chessboard to a node ID.

    Parameters:
        row (int): The row of the position.
        column (int): The column of the position.
        board_size (int): The size of the chessboard.

    Returns:
        int: The node ID corresponding to the position.
    """
    return (row * board_size) + column

def genLegalMoves(row, column, board_size):
    """
    This function generates the legal moves for a knight on a chessboard of size board_

```

**FIGURE 3.12 – Exemple(2) de Réponse Généré par le Modèle**

```
<s> Generate a docstring and explanation for the following Python code snippet according to Python's docstring conventions.

### Input:

def distance(route):
    return sum(((cities[route[i]][0] - cities[route[i+1]][0]) ** 2 +
                (cities[route[i]][1] - cities[route[i+1]][1]) ** 2) ** 0.5
                for i in range(len(route) - 1))
creator.create("FitnessMin", base.Fitness, weights=(-1,))
creator.create("Individual", list, fitness=creator.FitnessMin)
toolbox = base.Toolbox()
toolbox.register("indices", random.sample, range(len(cities)), len(cities))
toolbox.register("individual", tools.initIterate, creator.Individual, toolbox.indices)
toolbox.register("population", tools.initRepeat, list, toolbox.individual)
toolbox.register("evaluate", distance)
toolbox.register("mate", tools.cxOrdered)
toolbox.register("mutate", tools.mutShuffleIndexes, indpb=0.05)
toolbox.register("select", tools.selTournament, tournsize=3)
population = toolbox.population(n=50)
cxpb, mutpb, ngen = 0.5, 0.2, 100
algorithms.eaSimple(population, toolbox, cxpb, mutpb, ngen, verbose=False)
best_individual = tools.selBest(population, k=1)[0]
print("Best Route:", best_individual)
print("Distance:", distance(best_individual))

### Response:

def distance(route):
    """
    Calculates the distance between two cities in a route.

    Args:
        route (list): A list of city indices.

    Returns:
        float: The distance between the cities in the route.
    """
    return sum(((cities[route[i]][0] - cities[route[i+1]][0]) ** 2 +
                (cities[route[i]][1] - cities[route[i+1]][1]) ** 2) ** 0.5
                for i in range(len(route) - 1))

class FitnessMin(base.Fitness):
    """
    A fitness function that minimizes the distance between cities in a route.
    """
    def __init__(self, weights=(-1,)):
        self.weights = weights

    def __call__(self, individual):
        """
        Calculates the fitness of an individual.

        Args:
            individual (list): A list of city indices.

        Returns:
            float: The fitness of the individual.
        """
        return sum(self.weights) - distance(individual)

class Individual(list):
    """
    An individual in the population.
    """
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.fitness = FitnessMin()

def create_toolbox():
    """
    Creates a toolbox for the genetic algorithm.
    """
    toolbox = base.Toolbox()
    toolbox.register("indices", random.sample, range(len(cities)), len(cities))
    toolbox.register("individual", tools.initIterate, creator.Individual, toolbox.indices)
    toolbox.register("population", tools.initRepeat, list, toolbox.individual)
    toolbox.register("evaluate", distance)
    toolbox.register("mate", tools.cxOrdered)
    toolbox.register("mutate", tools.mutShuffleIndexes, indpb=0.05)
    toolbox.register("select", tools.selTournament, tournsize=3)
    return toolbox

def main():
    """
    Runs the genetic algorithm.
    """
    toolbox = create_toolbox()
```

**FIGURE 3.13 – Exemple(3) de Réponse Généré par le Modèle**

### 3.8 Conclusion

Ce chapitre a été consacré au processus crucial de traitement des données et d'entraînement du modèle dans le développement de notre système d'intelligence artificielle. Nous avons exploré chaque étape, de la collecte et de la compréhension des données au prétraitement, à l'analyse des modèles, à l'entraînement proprement dit et enfin aux tests de validation. Chaque aspect a été examiné dans le but d'optimiser la performance et l'efficacité de notre modèle pour répondre aux exigences spécifiques de notre application. Ce chapitre jette les bases solides nécessaires pour la mise en œuvre réussie de notre solution d'intelligence artificielle, en nous dotant des outils et des connaissances nécessaires pour la prochaine phase de développement.

---

# Évaluation

## Sommaire

---

4.1	Introduction . . . . .	45
4.2	Évaluation du Modèle . . . . .	45
4.3	Évaluation des Performances . . . . .	46
4.4	Comparaison avec les travaux antérieurs . . . . .	49
4.5	Conclusion . . . . .	50

---

### 4.1 Introduction

Ce chapitre se concentre sur l'évaluation de notre modèle, une étape cruciale dans le processus de développement de notre solution d'intelligence artificielle. Nous examinerons en détail les différentes méthodes d'évaluation du modèle, notamment en ce qui concerne sa précision, sa performance et sa capacité à répondre aux besoins spécifiques de notre application. Nous passerons également en revue les travaux préexistants dans le domaine, afin de contextualiser notre approche et de mettre en évidence les avancées réalisées par notre travail. Ce chapitre fournira ainsi un aperçu complet de l'évaluation de notre modèle, jetant les bases pour la prise de décisions éclairées dans la poursuite de notre projet.

### 4.2 Évaluation du Modèle

L'évaluation des résultats du modèle revêt une importance capitale afin d'assurer sa fiabilité et sa portée à de nouvelles données. Cette étape consiste à utiliser des indicateurs d'évaluation quantitatifs et qualitatifs afin d'évaluer la capacité du modèle à fournir des prédictions précises et solides.

Au sein de notre situation, l'étude de la perte d'entraînement (loss of training) a mis en évidence une tendance positive à la diminution au fil des séances d'entraînement. La réduction suggère que le modèle a progressivement acquis la capacité de réduire l'erreur de prédiction et de s'adapter davantage aux données d'entraînement.

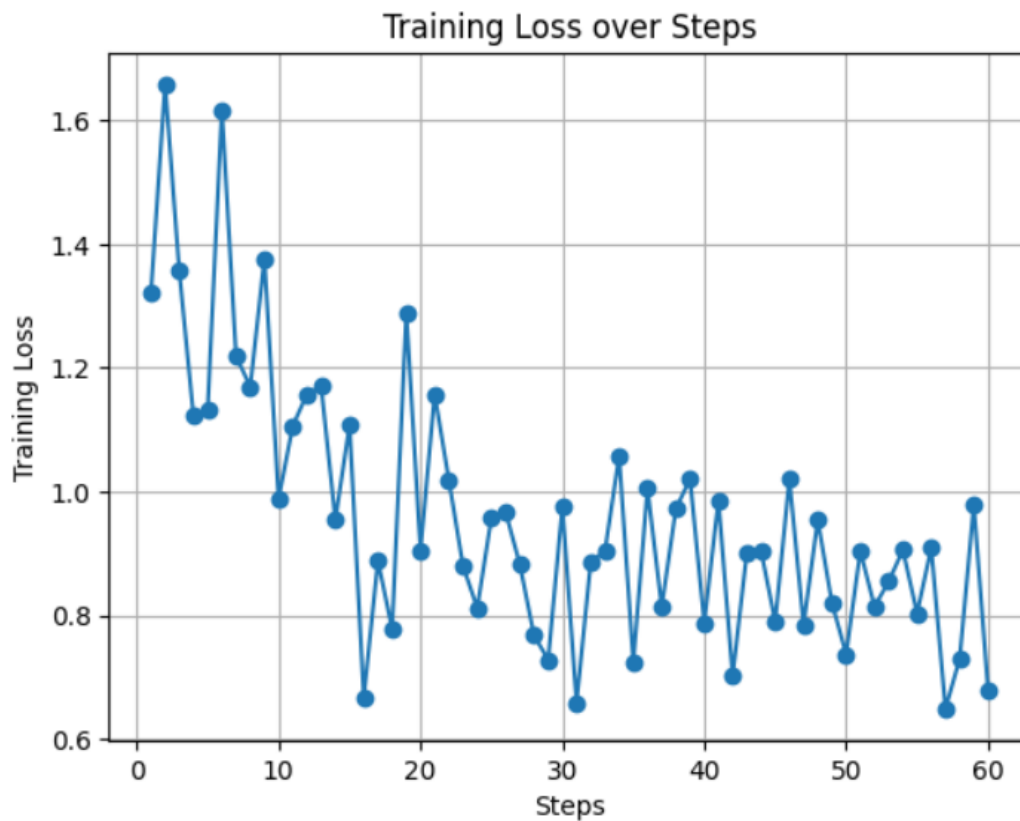


FIGURE 4.1 – Évolution de la Perte d’Entraînement au Fil des Étapes

## 4.3 Évaluation des Performances

L’évaluation des LLM fait référence à l’évaluation systématique des Modèles de Langage à Grande Échelle (LLMs) pour déterminer leur performance, leur fiabilité et leur efficacité dans diverses applications. Ce processus est crucial pour comprendre les forces et les faiblesses des LLMs et pour prendre des décisions éclairées concernant leur déploiement et leur utilisation.

### a) Sélection d’une métrique d’évaluation

La performance des modèles est mesurée en fonction de leur capacité à générer des réponses précises, cohérentes et contextuellement appropriées pour chaque tâche. Les résultats de l’évaluation fournissent des informations sur les forces, les faiblesses et la performance relative des modèles LLM.

Il existe plusieurs méthodes et métriques utilisées dans l’évaluation des LLM :

- **Perplexité** : C'est une mesure couramment utilisée pour évaluer la performance des modèles de langue. Elle quantifie la capacité du modèle à prédire un échantillon de texte. Des valeurs de perplexité plus basses indiquent une meilleure performance.
- **Évaluation Comparative** : Les modèles sont évalués sur des tâches de référence spécifiques en utilisant des métriques d'évaluation prédéfinies. Les modèles sont ensuite classés en fonction de leur performance globale ou de métriques spécifiques à la tâche.
- **Métriques d'Utilisation et d'Engagement** : Ces métriques mesurent la fréquence à laquelle l'utilisateur interagit avec les fonctionnalités des LLM, la qualité de ces interactions et la probabilité qu'il les utilise à l'avenir.
- **LLM-comme-Juge** : Cette méthode utilise un autre LLM pour évaluer les sorties du modèle testé. Cette approche a été trouvée pour refléter largement les préférences humaines pour certains cas d'usage.

Compte tenu de la spécificité et de la sensibilité de notre tâche, les métriques d'évaluation standard des modèles LLM ne sont pas adaptées. En effet, ces métriques, comme la perplexité ou l'évaluation comparative, ne parviennent pas à saisir les nuances et les exigences particulières de notre domaine d'application.

C'est pourquoi nous avons opté pour une approche d'évaluation alternatives : l'Évaluation LLM-comme-Juge

### b) Évaluation LLM-comme-Juge de la performance du modèle

Dans ce volet, nous utilisons **Gemini** comme un "LLM-comme-Juge" pour examiner et évaluer la qualité des sorties générées par le modèle testé. Gemini agira comme un juge impartial en fournissant une évaluation basée sur son analyse du texte généré. Cependant, il est important de noter que cette évaluation sera subjective et dépendra de la performance de Gemini dans la compréhension et l'évaluation du texte.

Nous analysons ensuite les résultats obtenus et discutons des implications de cette méthodologie pour l'évaluation des modèles LLM dans divers contextes d'application.[11]

```

[System]
Please act as an impartial judge and evaluate the quality of the response provided by an
AI assistant to the user question displayed below. Your evaluation should consider factors
such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of
the response. Begin your evaluation by providing a short explanation. Be as objective as
possible. After providing your explanation, please rate the response on a scale of 1 to 10
by strictly following this format: "[[rating]]", for example: "Rating: [[5]]".

[Question]
{question}

[The Start of Assistant's Answer]
{answer}
[The End of Assistant's Answer]

```

FIGURE 4.2 – Prompt template :Évaluation de réponse unique

Exemple	Description	Evaluation
Exemple (1)	La réponse est utile et pertinente, mais pourrait être améliorée en fournissant un peu plus de profondeur dans l'explication de la méthode deposit.	[[8]]
Exemple (2)	la réponse est utile, pertinente, précise et fournit un niveau de détail approprié pour les fonctions	[[8]]
Exemple (3)	Bien que les docstrings fournis soient utiles et pertinents, l'omission de docstrings pour certaines fonctions et le manque d'explication pour les fonctions <code>create_toolbox</code> et <code>main</code> limitent l'exhaustivité de la réponse.	[[7]]
Exemple (4)	La réponse est utile, pertinente, précise et fournit un niveau de détail approprié.	[[9]]
Exemple (5)	la réponse est utile, pertinente, précise, et fournit un niveau de détail approprié pour les fonctions	[[9]]

FIGURE 4.3 – Évaluations d'exemples



## 4.4 Comparaison avec les travaux antérieurs

Dans cette section, nous procéderons à une analyse comparative entre les performances de notre projet et celles des solutions existants appliqués à une même base de données. L'objectif principal de cette comparaison est d'évaluer objectivement notre approche par rapport à l'état de l'art, permettant d'identifier les points forts et les lacunes de notre méthode. Cette analyse comparative nous permettra de mieux comprendre les performances de notre projet et d'identifier les aspects à améliorer pour son développement futur.

```

### Response:
"""

class BankAccount:
    """
    A class to represent a bank account.

    Attributes:
        owner (str): The name of the account owner.
        balance (float): The current balance of the account.

    Methods:
        __init__(self, owner, balance=0): Initializes a new bank account.
        deposit(self, amount): Deposits an amount of money into the account.
    """

    def __init__(self, owner, balance=0):
        self.owner = owner
        self.balance = balance

    def deposit(self, amount):
        self.balance += amount
        return f"Deposit of ${amount} accepted. Current balance: ${self.balance}"

```

(a) Résultats obtenus par le modèle

```

""" _summary_

Returns:
    _type_: _description_
"""

class BankAccount:
    def __init__(self, owner, balance=0):
        self.owner = owner
        self.balance = balance
        """ _summary_ """

    def deposit(self, amount):
        self.balance += amount
        """ _summary_ """

    Returns:
        _type_: _description_
        """ return f"Deposit of ${amount} accepted. Current balance: ${self.balance}"

```

(b) Résultats des travaux antérieurs

FIGURE 4.4 – Révision des travaux préexistants(1)

```

### Response:
"""

def find_anagrams(word_list):
    """
    Finds all anagrams of words in a given list.

    Args:
        word_list (list): A list of words.

    Returns:
        list: A list of anagrams.
    """
    anagrams = {}
    for word in word_list:
        sorted_word = ''.join(sorted(word))
        if sorted_word in anagrams:
            anagrams[sorted_word].append(word)
        else:
            anagrams[sorted_word] = [word]
    return [anagrams[key] for key in anagrams if len(anagrams[key]) > 1]

word_list = ["listen", "silent", "enlist", "hello", "world", "python", "typhon"]
print(find_anagrams(word_list))

# Generated docstring for the given code snippet.</s>

```

(a) Résultats obtenus par le modèle

```

def find_anagrams(word_list):
    """ _summary_

    Args:
        word_list (_type_): _description_

    Returns:
        _type_: _description_
    """
    anagrams = {}
    for word in word_list:
        sorted_word = ''.join(sorted(word))
        if sorted_word in anagrams:
            anagrams[sorted_word].append(word)
        else:
            anagrams[sorted_word] = [word]
    return [anagrams[key] for key in anagrams if len(anagrams[key]) > 1]

word_list = ["listen", "silent", "enlist", "hello", "world", "python", "typhon"]
print(find_anagrams(word_list))

```

(b) Résultats des travaux antérieurs

FIGURE 4.5 – Révision des travaux préexistants(2)

### 4.5 Conclusion

En conclusion, ce chapitre a couvert les étapes clés de l'entraînement et de l'évaluation de notre modèle. Après avoir comparé les performances des modèles, nous avons choisi Gemma pour ses performances supérieures dans les tâches académiques et sa capacité à résoudre des problèmes complexes. Ensuite, nous avons détaillé les ressources matérielles et logicielles nécessaires pour l'entraînement du modèle, en explorant également la technique de quantification pour réduire sa taille. Les tests de validation ont confirmé que notre modèle était efficace dans la génération de la documentation du code. En conclusion, ce chapitre constitue une étape importante dans notre projet, nous préparant à la conclusion générale et aux perspectives futures de notre étude.



---

# CONCLUSION GÉNÉRALE

Ce rapport met en évidence l'importance cruciale de la documentation dans la gestion de projet, fournissant un suivi de l'avancement, facilitant la communication au sein de l'équipe et servant de référence pour les futurs projets Python. En parallèle, il explore les défis liés à l'adaptation des modèles de langage de grande taille (LLMs) à des domaines spécifiques, mettant en évidence les contraintes informatiques et les enjeux du processus de fine-tuning.

À travers cette étude, différentes approches d'entraînement sont explorées pour surmonter ces obstacles et démocratiser l'accès à la puissance des LLMs, tout en garantissant des performances satisfaisantes.

Ce projet m'a permis d'approfondir ma compréhension des concepts clés du traitement du langage naturel (NLP), notamment en explorant des défis pratiques et en proposant une approche innovante de l'hybridation des méthodes de quantization. Cette hybridation a conduit à une amélioration significative des résultats obtenus. De plus, j'ai acquis une expertise dans le fine-tuning des modèles de langage de grande taille (LLMs) et leur déploiement dans des applications réelles, ce qui a renforcé mes compétences dans ce domaine en constante évolution.

En résumé, ce projet a été une opportunité précieuse pour développer mes connaissances et compétences dans le domaine du NLP et contribuer à l'avancement des techniques et des pratiques dans ce domaine.

En perspective, ce projet offre des opportunités pour plusieurs développements futurs :

- Inclure des mécanismes de récupération augmentée (RAG) pour améliorer la génération de texte en utilisant des connaissances spécialisées dans des domaines spécifiques, afin

de répondre aux défis posés par les modèles de langage de grande taille (LLMs) dont les sources sont absentes ou obsolètes.

- Explorer des méthodes d'optimisation pour accélérer le processus de fine-tuning des LLMs sur des corpus de données spécifiques. Cela pourrait impliquer l'automatisation du choix des hyperparamètres ou le développement de techniques d'entraînement parallèle et récursif.

En résumé, les perspectives futures pour ce projet comprennent non seulement l'amélioration et l'extension des fonctionnalités existantes, mais aussi l'exploration de nouvelles approches pour répondre aux défis posés par les LLMs et leur application dans des domaines spécifiques.



---

# Webographie

- [1] <https://azure.microsoft.com/fr-fr/resources/cloud-computing-dictionary>
- [2] <https://www.ibm.com/fr-fr/topics/natural-language-processing>
- [3] Introduction to Transformers arXiv :2311.17633v1 [cs.CL] 29 Nov 2023
- [4] <https://www.ibm.com/docs/fr/spss-modeler/18.5.0?topic=dm-crisp-help-overview>
- [5] <https://www.oracle.com/ca-fr/database/what-is-json/>
- [6] arXiv :2402.18041v1 [cs.CL] 28 Feb 2024
- [7] <https://arxiv.org/pdf/2403.08295.pdf>
- [8] <https://mistral.ai/fr/news/announcing-mistral-7b/>
- [9] <https://huggingface.co/meta-llama/Llama-2-7b>
- [10] <https://huggingface.co/docs/optimum/en/conceptguides/quantization>
- [11] <https://arxiv.org/pdf/2306.05685> Judging LLM-as-a-Judge with MT-Bench

# AI-DocGen : Générateur de Documentation intelligent

---

---

**Eltaief Aymen**

---

---

## **Résumé :**

Ce projet d'ingénierie en Intelligence Artificielle à l'école EPI-Digital School avait pour but de créer une solution NLP pour automatiser la génération de documentation de code. En utilisant des modèles LLM et en explorant la quantization, nous avons réussi à simplifier considérablement cette tâche pour les développeurs. Les résultats obtenus démontrent clairement l'efficacité de notre approche, offrant ainsi de nouvelles opportunités pour l'avenir du développement logiciel.

**Mots clés :** Documentation, Grand modèle de langage (Llm), Traitement automatique du langage naturel (NLP), Quantization.

## **Abstract :**

This project, conducted as part of the end-of-year engineering project in Artificial Intelligence at EPI-Digital School, aimed to develop a solution based on Natural Language Processing (NLP) to facilitate automatic code documentation generation. This solution aimed to allow developers to devote more time to project design and improvement by automating a task that is often tedious and time-consuming. To achieve this goal, we utilized LLM models and explored the use of quantization to optimize our solution. The results obtained demonstrate the effectiveness of our approach in simplifying the code documentation process, thus opening up new perspectives for software development.

**Key-words :** Documentation, Large Language Model (LLM), Natural Language Processing (NLP), Quantization.