SUP'COM
Higher School of Communication of Tunis

Rapport du projet tutoré

Sujet :

Développement d'un Chatbot RH Intelligent : Automatisation des Processus RH grâce à l'IA Générative

Travail proposé et réalisé en collaboration avec :

ACTIA®
ENGINEERING SERVICES

Réalisé par :

Nour LAABIDI

Nouha BEN HAMADA

Encadrées par :

Madame Ferdaous CHAABANE

Monsieur Fethi TLILI

Année universitaire : 2024/2025

Remerciements

Nous souhaitons tout d'abord exprimer notre profonde gratitude à ACTIA Engineering Services, et plus particulièrement à Madame Ferdaous CHAABANE et Monsieur Fethi TLILI, pour nous avoir offert cette opportunité et pour la confiance qu'ils nous ont témoignée. Leur encadrement et leurs précieux conseils ont largement contribué à notre développement personnel et à notre progression dans le domaine de l'intelligence artificielle.

Ce stage a constitué une étape clé de notre parcours universitaire, nous permettant de renforcer nos connaissances et d'élargir nos perspectives pour l'avenir professionnel.

Enfin, nous adressons un remerciement particulier à notre relecteur et correcteur, dont les avis pertinents et recommandations constructives ont considérablement amélioré la qualité et la structure de ce rapport de stage.

Résumé

Ce projet tutoré s'inscrit dans le cadre de notre cursus académique au cycle d'ingénieur. L'objectif principal est de nous préparer, au cours de cette dernière année d'études, à la transition vers la vie professionnelle.

Le sujet proposé par ACTIA Engineering Services porte sur la conception et l'implémentation d'un chatbot intelligent, basé sur l'intelligence artificielle générative, afin d'améliorer l'efficacité et la productivité du département des ressources humaines (RH) d'une entreprise spécialisée dans l'industrie automobile.

Ce chatbot a pour vocation de répondre de manière automatisée et précise aux questions fréquemment posées par les employés et vise à alléger la charge de travail des équipes RH, leur permettant de se concentrer sur des tâches à plus forte valeur ajoutée.

Mots clés :

Intelligence Artificielle, LLM, Chatbot, RAG, Fine-Tuning

Table des matières

Table des figures

Acronymes

API Application Programming Interface.

GPU Graphics Processing Unit.

LLM Large language model.

LoRA Low-Rank Adaptation.

NLP Natural Language Processing.

RAG Retrieval Augmented Generation.

RAGAS Retrieval-Augmented Generation Analysis Suite.

RH Ressources humaines.

Introduction

Les départements des ressources humaines (RH) jouent un rôle central dans le bon fonctionnement et le développement des entreprises. Cependant, en parallèle, les agents RH sont confrontés à une charge de travail importante et chronophage, due à la multitude de questions répétitives formulées quotidiennement par les employés. Ces questions concernent souvent des thématiques récurrentes, telles que les congés, les démarches administratives, les demandes de documents officiels, ou encore les processus de recrutement. Cette pression constante et ce volume élevé d'interactions réduisent leur disponibilité pour des activités à forte valeur ajoutée. Cela souligne la nécessité croissante de mettre en place des solutions technologiques pour automatiser les réponses à ces demandes récurrentes, simplifier les processus quotidiens, et ainsi optimiser leur efficacité.

Dans ce contexte, notre projet vise à concevoir et à développer un chatbot RH intelligent, reposant sur l'intelligence artificielle générative, les techniques de traitement automatique du langage naturel (NLP), et les modèles de langage de grande taille (LLM). L'objectif est d'automatiser et de simplifier les processus RH chez ACTIA Engineering Services, en permettant au chatbot de répondre de manière précise et instantanée aux questions fréquentes des employés. Ce projet ambitionne de renforcer l'efficacité opérationnelle et de libérer les agents RH des tâches répétitives pour qu'ils puissent se consacrer à des missions plus stratégiques.

Ce rapport documente le processus de conception et de mise en œuvre de ce chatbot RH. Il se structure comme suit :

— Le premier chapitre est consacré à la présentation l'entreprise ACTIA Engineering Services ainsi que des objectifs du projet.

— Le deuxième chapitre contient une étude théorique des concepts et des méthodologies employés, notamment les différentes composantes utilisées dans le projet.

— Le troisième chapitre approfondit les étapes ayant guidé la mise en œuvre de notre solution.

— Le dernier chapitre présente une évaluation de la performance du système réalisé dans le cadre du projet.

CHAPITRE 1

Contexte général du projet

Introduction

Ce chapitre a pour but d'exposer le contexte de notre projet de stage. Il présente l'organisme d'accueil, décrit le projet dans ses grandes lignes, expose le contexte ainsi que la problématique que nous avons identifiée, et détaille la solution que nous avons proposée pour résoudre cette problématique.

1.1 Présentation de l'organisme d'accueil

ACTIA Engineering Services est une filiale du groupe ACTIA, spécialisé dans la conception et la production de systèmes électroniques avancés pour des secteurs variés comme l'automobile, l'aéronautique, et l'énergie. L'entreprise se distingue par son expertise en ingénierie de systèmes embarqués et connectés, en offrant des solutions technologiques innovantes adaptées aux besoins des industriels. Avec une forte présence internationale, ACTIA Engineering Services se positionne comme un acteur clé dans le domaine des technologies émergentes, telles que l'intelligence artificielle et les systèmes IoT, tout en maintenant un engagement ferme envers l'excellence et la durabilité.

FIGURE 1.1 – Logo de ACTIA

1.2 Presentation du projet

Cette partie entamera le contexte général du projet en mettant l'accent sur la problematique et la solution proposée.

1.2.1 Problématique

Les départements des ressources humaines (RH) sont essentiels au fonctionnement des entreprises, mais ils sont souvent submergés par la gestion de nombreuses tâches administratives et répétitives, notamment les réponses aux questions fréquentes des employés sur des sujets tels que les congés, les procédures administratives ou le recrutement. Cette charge de travail engendre une perte de temps significative et limite la disponibilité des agents RH pour des missions stratégiques à plus forte valeur ajoutée, comme le développement des politiques internes ou l'accompagnement des collaborateurs. Dans ce contexte, il est crucial de mettre en place des solutions technologiques innovantes pour optimiser la gestion des tâches RH tout en améliorant la qualité et l'accessibilité des services proposés aux employés.

1.2.2 Solution proposée

La solution proposée consiste en la création d'un chatbot RH intelligent alimenté par l'intelligence artificielle générative et utilisant des modèles de langage de grande taille (LLM). Ce chatbot a pour objectif d'automatiser et de simplifier la gestion des demandes récurrentes des employés au sein du département des ressources humaines de ACTIA Engineering Services. Il sera capable de répondre de manière instantanée et précise aux questions fréquentes des employés concernant les congés, les procédures administratives, ou le recrutement. La solution inclut également la génération, la préparation et le traitement des données relatives aux ressources humaines et aux employés. Le chatbot sera implémenté en utilisant deux méthodes : RAG (Retrieval-Augmented Generation) et le Fine-Tuning d'un modèle LLM. Enfin, les résultats obtenus seront comparés afin d'évaluer l'efficacité et la précision des deux approches.

Conclusion

En conclusion de ce chapitre, nous avons présenté le contexte général de notre projet en mettant en avant l'entreprise collabporatrice, ACTIA Engineering Services. Nous avons également défini la solution proposée, qui vise à développer un chatbot RH implémenté avec l'IA générative, en comaparant les performances de deux approches : le RAG et le Fine-Tuning d'un modéle LLM.

CHAPITRE 2

Etude préliminaire

Introduction

Dans ce chapitre, nous effectuerons une étude théorique des technologies, modèles et méthodologies utilisées dans le cadre de ce projet, notamment les modèles de langage de grande taille (LLM), le Retrieval-Augmented Generation (RAG) et le Fine-Tuning. Ensuite, nous décrirons les étapes de conception mises en œuvre pour développer le chatbot intelligent. Enfin, nous présenterons les métriques d'évaluation utilisées pour mesurer la performance et la pertinence des solutions proposées, permettant ainsi une comparaison entre les différentes approches adoptées.

2.1 Etude théorique

2.1.1 Modèles de langage de grande taille

Les modèles de langage de grande taille (LLM) sont des réseaux neuronaux puissants conçus pour traiter et générer du texte en s'appuyant sur une vaste quantité de données textuelles. Ils sont entraînés sur d'énormes corpus de textes et sont capables de comprendre, analyser et produire des réponses en langage naturel. Dans ce projet, nous avons utilisé GPT-3.5 Turbo, LLAMA 3.2, Falcon 7B et all-MiniLM-L6-v2.

GPT 3.5 Turbo

GPT-3.5 Turbo est une version optimisée de GPT-3, développée par OpenAI. Ce modèle utilise l'architecture des transformeurs pour traiter les séquences de texte et est préalablement entraîné sur un large corpus de textes issus de diverses sources. Grâce à son architecture, GPT-3.5 Turbo est capable de générer des textes cohérents et contextuellement pertinents, tout en maintenant un faible temps de réponse, ce qui le rend adapté à des applications en temps réel comme les chatbots.

LLAMA 3.2

Le modèle LLAMA 3.2 (Large Language Model Meta AI) est une version avancée des modèles de traitement du langage naturel développés par Meta AI. LLAMA 3.2 s'appuie sur une architecture transformer optimisée, permettant un traitement efficace des séquences textuelles de grande longueur. Ce modèle est spécialement conçu pour fournir des performances élevées sur une variété de tâches NLP, notamment la génération de texte, la classification, la traduction automatique et la création de résumés.

Falcon 7B

Falcon 7B est un modèle de langage de grande taille développé par Hugging Face dans le cadre de la série Falcon. Avec ses 7 milliards de paramètres, Falcon 7B utilise l'architecture des transformeurs pour effectuer des tâches complexes de traitement du langage naturel, telles que la génération de texte et la compréhension contextuelle.

all-MiniLM-L6-v2

C'est est un modèle léger de transformer développé par Hugging Face et basé sur l'architecture MiniLM. Il est conçu pour générer des embeddings de phrases ou de textes, tout en offrant un bon équilibre entre performance et efficacité. Avec seulement 6 couches et des millions de paramètres, ce modèle est particulièrement optimisé pour les tâches de similarité sémantique, de classification de texte et de recherche.

2.1.2 Retrieval-Augmented Generation

Le Retrieval-Augmented Generation (RAG) est une approche hybride qui combine la récupération d'informations et la génération de texte. Contrairement aux modèles traditionnels de génération de texte, qui s'appuient uniquement sur les connaissances acquises lors de l'entraînement, RAG utilise une étape de recherche dans une base de données externe ou un corpus documentaire pour récupérer des informations pertinentes avant de générer une réponse.

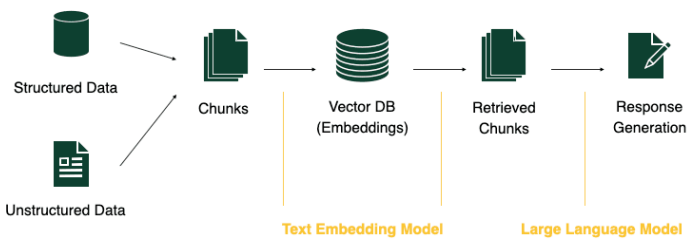

FIGURE 2.1 – Architecture d'un RAG simple

Le chunking sémantique

Le chunking sémantique des données consiste à diviser de grandes quantités de texte en unités plus petites et plus significatives, appelées chunks, en se basant sur leur contenu sémantique plutôt que sur des délimitations strictes comme les phrases ou les paragraphes. Une technique courante dans cette approche est le calcul de la similarité cosinus (cosine similarity), qui mesure la similarité entre deux vecteurs de texte en calculant l'angle entre eux dans un espace vectoriel. Cette méthode permet d'évaluer à quel point deux morceaux de texte, ou "chunks", sont similaires en termes de contenu sémantique.

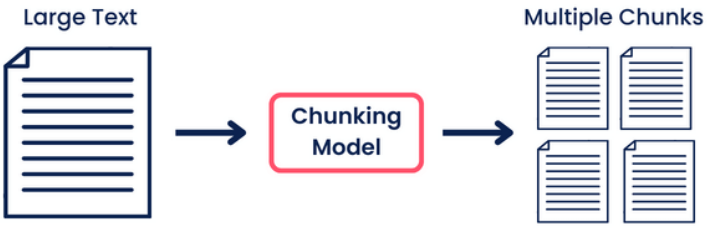

FIGURE 2.2 – Découpage des données en chunks

Sentence Transformer Embeddings

Ce sont des représentations vectorielles d'entités textuelles, telles que des phrases ou des paragraphes, générées à l'aide de modèles préentraînés comme all-MiniLM-L6-v2. Contrairement aux modèles traditionnels qui génèrent des embeddings au niveau du mot, les Sentence Transformers produisent des vecteurs d'une dimension fixe pour des unités plus longues de texte, telles que des phrases, tout en capturant leur signification contextuelle.

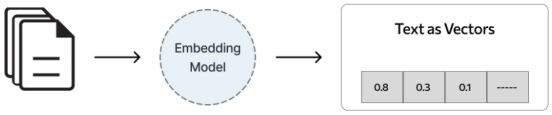

FIGURE 2.3 – Sentence Transformer Embeddings

2.1.3 Le Fine-Tuning

Le fine-tuning est une technique d'entraînement de modèles de machine learning, qui consiste à adapter un modèle préexistant aux spécificités d'un nouveau domaine ou d'une tâche particulière. Contrairement à un entraînement complet depuis zéro, le fine-tuning modifie uniquement les couches du modèle qui nécessitent des ajustements pour le rendre

plus précis sur des données spécifiques. Cette approche permet de tirer parti des connaissances générales déjà apprises par le modèle, tout en l'adaptant rapidement et efficacement à des exigences particulières, ce qui est souvent plus rapide et plus économiquement viable.

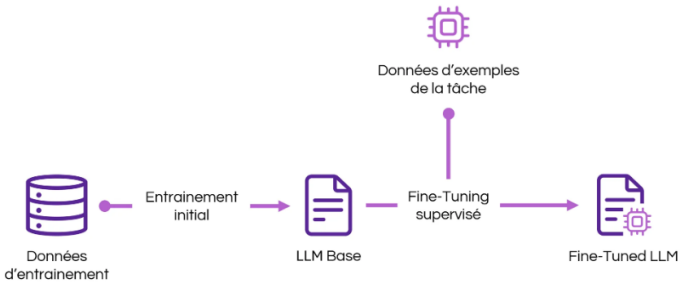

FIGURE 2.4 – Fine-Tuning d'un modèle LLM

Les techniques d'optimisation du Fine-Tuning

Les modèles de langage de grande taille (LLM) sont souvent constitués de milliards, voire de trillions de paramètres, ce qui rend leur fine-tuning difficile en termes de temps de calcul et de consommation mémoire. Pour surmonter ces défis et rendre le fine-tuning plus abordable, plusieurs techniques d'optimisation ont été proposées, parmi lesquelles la Low-Rank Adaptation (LoRA) et la quantification à 4 bits.

- **LoRA (Low-Rank Adaptation) :** C'est une méthode d'optimisation du fine-tuning qui permet d'adapter un modèle préexistant tout en réduisant considérablement les besoins en ressources. L'idée principale de LoRA est de modifier uniquement des sous-espaces de faible dimension dans les matrices de poids du modèle. Plutôt que de modifier tous les paramètres d'un modèle LLM, LoRA insère des matrices de faible rang au sein des couches du réseau, permettant de limiter le nombre de poids à ajuster tout en préservant les performances du modèle.

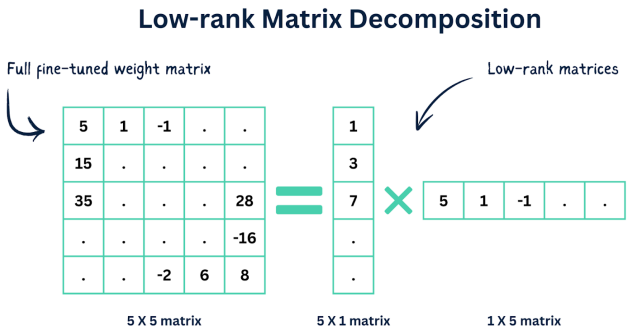

FIGURE 2.5 – Décomposition d'une matrice low-rank

- **La quantification à 4 bits :** La quantification à 4 bits est une autre méthode d'optimisation qui consiste à réduire la précision des poids du modèle, ce qui diminue la taille nécessaire pour stocker ces derniers. En utilisant seulement 4 bits pour représenter chaque poids, cette technique permet de réduire considérablement la mémoire consommée par le modèle, ce qui facilite son fine-tuning même sur des ressources limitées.

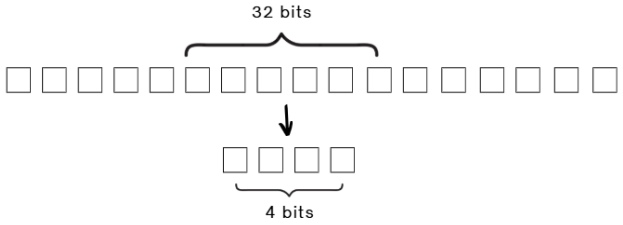

FIGURE 2.6 – Quantification à 4 bits

2.2 Conception du projet

Ce projet a été conçu en suivant une série d'étapes bien définies, regroupant à la fois la génération des données, l'implémentation de différentes méthodologies, et la comparaison des performances obtenues :

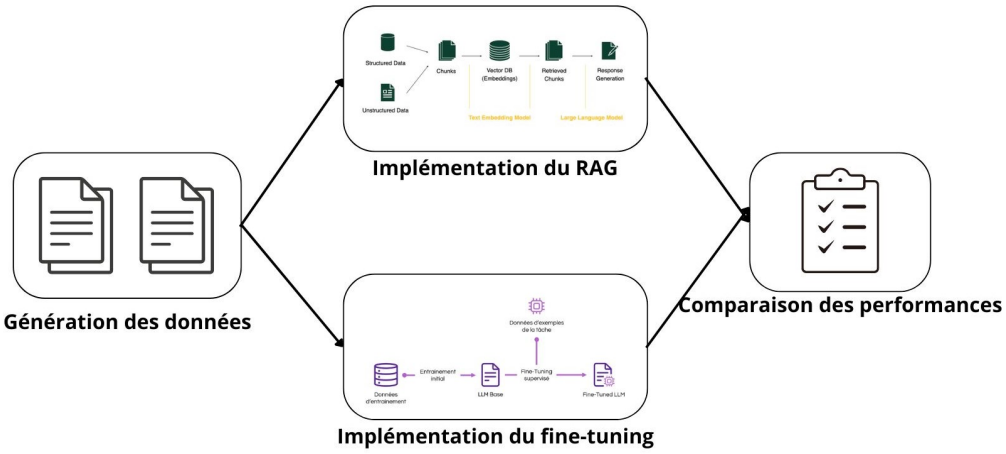

FIGURE 2.7 – Etapes de la conception du projet

- **Génération des données :** La première étape consiste à créer un ensemble de données fictives adaptées aux besoins du projet.
- **Implémentation du pipeline RAG :** Cette étape consiste à configurer et intégrer un pipeline RAG permettant d'augmenter le modèle génératif grâce à des données externes.

- **Implémentation du fine-tuning :** Mise en œuvre d'un entraînement spécialisé sur un modèle LLM pour personnaliser ses réponses aux besoins spécifiques du projet RH.
- **Comparaison des performances :** Une évaluation est réalisée pour comparer les performances des deux méthodologies principales : le pipeline RAG et le fine-tuning du modèle LLM.

2.2.1 Les étapes du RAG

Les différentes étapes du pipeline RAG comprennent :

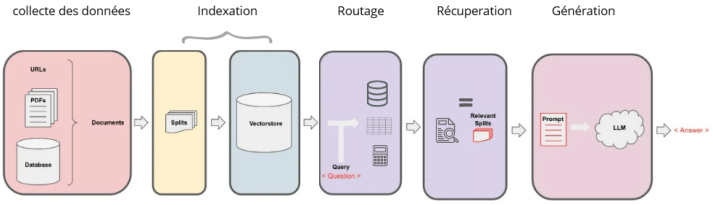

FIGURE 2.8 – Les étapes du RAG

Indexation

- **Chunking sémantique des données :** Fractionnement du contenu des documents en morceaux de texte cohérents et significatifs.
- **Génération de résumés des chunks :** Création de résumés courts pour chaque chunk afin de faciliter la recherche rapide.
- **Transformation en embeddings vectoriels :** Conversion des chunks et de leurs résumés en représentations vectorielles à l'aide de modèles de génération d'embeddings.
- **Stockage dans une base de données vectorielle :** Enregistrement des vecteurs associés aux chunks et à leurs métadonnées dans une base optimisée pour les recherches vectorielles.

Routage

Implémentation d'un mécanisme permettant de diriger les requêtes entrantes vers la source de données appropriée. Le système peut :

- **Recherche dans les documents texte :** Utilisation de la méthode multivector-retriever pour chercher à travers les vecteurs des résumés. La recherche est effectuée avec similarity search pour identifier les chunks les plus pertinents et retourner leurs contenus originaux.

- **Recherche dans les fichiers CSV :** Conversion de la requête utilisateur en une opération sur les données du CSV, comme la recherche d'un employé ou d'une date d'embauche.

Récupération des données

- Rechercher dans la base de données vectorielle pour des informations dans les documents textes.
- Traiter les opérations de recherche dans le fichier CSV contenant les données des employés.

Génération

À partir des données récupérées, le modèle génère une réponse finale précise et cohérente, destinée à être présentée à l'utilisateur.

2.2.2 Les étapes du Fine-Tuning

L'implémentation du fine-tuning du modèle LLM suit les étapes suivantes :

FIGURE 2.9 – Les étapes du Fine-Tuning

Création du jeu de données d'entraînement

Utiliser le modèle LLM pour générer des paires de questions et réponses basées sur le contenu RH.

Évaluation du jeu de données

Analyse et validation des paires questions-réponses pour s'assurer de leur pertinence et précision.

Techniques d'optimisation

Implémentation de méthodes spécifiques pour réduire le temps et la mémoire nécessaires à l'entraînement.

Entraînement du modèle LLM

Adaptation du modèle en utilisant les paires question-réponse validées pour renforcer sa capacité à traiter les requêtes RH.

Évaluation des performances

Mesurer les résultats du modèle fine-tuné sur un ensemble de données de test pour vérifier son amélioration en précision et en pertinence.

2.3 Métriques d'évaluation

2.3.1 Évaluation des questions/réponses générées pour le Fine-Tuning

Pour évaluer la qualité des questions/réponses générées et utilisées lors du Fine-Tuning, nous avons implémenté la métrique de fiabilité en utilisant le framework RAGAS (**Retrieval-Augmented Generation Analysis Suite**).

La fiabilité, dans ce contexte, fait référence à la capacité du système à fournir des réponses précises, cohérentes et pertinentes basées sur les documents de référence disponibles. Elle prend en compte des aspects tels que la fidélité à la source (absence de contenu inventé), la pertinence contextuelle par rapport à la requête de l'utilisateur, et l'exactitude factuelle. Ces évaluations permettent de quantifier dans quelle mesure les réponses générées respectent les informations issues du processus de récupération tout en restant adaptées à la demande initiale.

2.3.2 Évaluation des performances du RAG et du Fine-Tuning

Processus d'évaluation

Pour comparer les performances des systèmes basés sur le RAG et le Fine-Tuning, nous avons suivi les étapes suivantes :

1. **Calcul de la similarité cosinus :** Les réponses sont converties en vecteurs d'embedding à l'aide d'un modèle comme *all-MiniLM-L6-v2*, puis la **cosine similarity** est calculée pour les deux approches par rapport aux réponses attendues :

$$\text{Sin}(r_{true}, r_{model}) = \frac{\vec{r}_{true} \cdot \vec{r}_{model}}{\|\vec{r}_{true}\| \cdot \|\vec{r}_{model}\|} \tag{2.1}$$

   où :

   - $\vec{r}_{true}$: Embedding de la réponse attendue.
   - $\vec{r}_{model}$: Embedding de la réponse générée (RAG ou Fine-Tuning).
   - $\|\vec{r}\|$: Norme euclidienne du vecteur $\vec{r}$.

2. **Comparaison des scores moyens :** Les similarités moyennes sont calculées pour chaque système (RAG et Fine-Tuning) sur toutes les réponses :

$$S_{mean,rag} = \frac{1}{n} \sum_{i=1}^{n} \text{Sim}(r_{true}^{i}, r_{rag}^{i}) \tag{2.2}$$

$$S_{mean,ft} = \frac{1}{n} \sum_{i=1}^{n} \text{Sim}(r_{true}^{i}, r_{ft}^{i}) \tag{2.3}$$

où n est le nombre total de questions évaluées.

3. **Analyse des résultats :** Un score de similarité moyen (S_{mean}) plus élevé indique des réponses générées plus proches des attentes.

Conclusion

En conclusion, ce chapitre a présenté les concepts théoriques liés au projet, notamment les modèles LLM utilisés, ainsi que les technologies employées pour implémenter le RAG et le Fine-Tuning. Nous avons également détaillé les différentes étapes d'implémentation du projet, ainsi que les métriques d'évaluation permettant de comparer les deux approches.

CHAPITRE 3

Implémentation

Introduction

Dans ce chapitre, nous détaillerons la mise en œuvre technique de l'implémentation du chatbot des Ressources Humaines. Nous commencerons par présenter l'environnement de travail, incluant les plateformes utilisées et les frameworks choisis. Ensuite, nous aborderons la génération des données, ainsi que l'implémentation du RAG et du Fine-Tuning.

3.1 Environnement de travail

Pour la réalisation de ce projet, nous avons utilisé une combinaison d'outils et de bibliothèques Python afin de développer, tester et évaluer le chatbot. Les principaux outils utilisés sont les suivants :

Plateformes :

- **Google Colab** : une plateforme de développement collaboratif en ligne qui fournit un environnement interactif pour écrire et exécuter des scripts Python, avec un accès facile aux ressources matérielles comme les GPU.
- **Hugging Face** : une plateforme qui offre des modèles de langage pré-entraînés et un environnement pour les fine-tuning ainsi que des outils pour travailler sur le traitement du langage naturel (NLP).
- **Azure OpenAI** : une solution cloud fournissant des modèles de langage développés par OpenAI, permettant d'intégrer facilement des modèles d'IA avancés dans des applications d'entreprise via des API.
- **Weights and Biases** : une plateforme de suivi et de gestion des expériences de machine learning, permettant de surveiller les hyperparamètres, visualiser les métriques, et collaborer sur les projets.

Frameworks :

- **LangChain** : un framework permettant de construire des applications NLP complexes en orchestrant plusieurs modèles, sources de données et outils comme les langages de programmation.
- **Transformers** : un framework développé par Hugging Face qui permet d'utiliser et d'optimiser des modèles pré-entraînés pour des tâches NLP comme la classification, la génération de texte, etc.
- **PyTorch** : un framework de machine learning permettant de définir, entraîner et déployer des modèles de réseaux neuronaux, couramment utilisé pour les projets de Deep Learning.

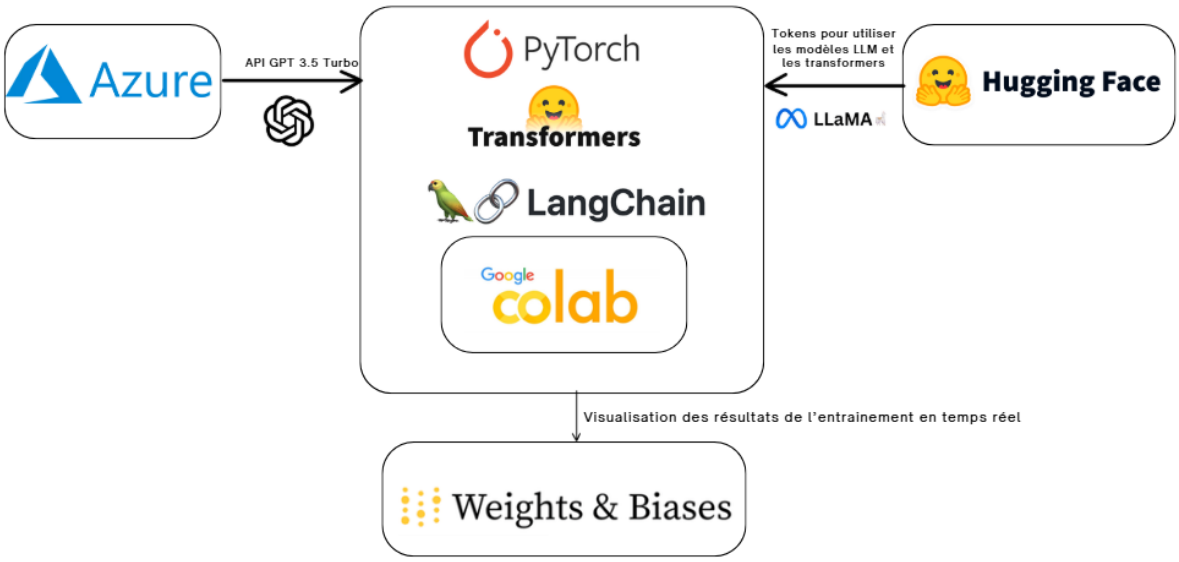

FIGURE 3.1 – Plateformes et Frameworks utilisés

Librairies Python :

- **sentence-transformers** : une bibliothèque pour générer des représentations vectorielles (embeddings) de textes afin de faciliter la comparaison sémantique.
- **Faker** : une bibliothèque Python permettant de créer des données fictives.
- **Streamlit** : une bibliothèque Python permettant de créer rapidement des interfaces utilisateur interactives pour les applications de machine learning.
- **bitsandbytes** : une bibliothèque Python optimisée permettant la quantification et l'entraînement de modèles de deep learning en 4 et 8 bits pour améliorer l'efficacité mémoire et les performances.
- **pandas** : une bibliothèque pour la manipulation et l'analyse de données, offrant des structures de données et des outils pour gérer les données en format tabulaire.

- **scikit-learn** : une bibliothèque fournissant des outils simples et efficaces pour le machine learning et l'analyse des données, utilisée pour l'implémentation et l'évaluation de modèles.
- **numpy** : une bibliothèque pour le calcul scientifique avec des tableaux multidimensionnels et des fonctions mathématiques rapides.
- **matplotlib** : une bibliothèque de visualisation graphique permettant de créer des graphiques et des visualisations statistiques à partir des données.

3.2 Génération de données

En s'inspirant des données de ressources humaines open-source, nous avons généré :

- un document PDF de politiques de ressources humaines, contenant les politiques de congés, les procédures de recrutement, la formation et le développement, la gestion des performances, la conduite des employés, la santé et sécurité au travail, le télétravail et la flexibilité, ainsi que les relations sociales.
- Un document CSV contenant des données sur 983 employés fictifs, comprenant leurs coordonnées personnelles, département, poste, date d'embauche, date de départ, ancienneté, type de contrat, salaire, congés disponibles et le nom du manager. Pour générer ces données de manière automatisée et réaliste, nous avons utilisé la bibliothéque Faker pour créer des données fictives.

ID Employé	Nom	Prénom	Genre	Date de naiss	Âge	Adresse Emai	Numéro de té	Adresse post	Département	Date d'embauche	Date de dépar	Ancienneté (a	Poste	Type de contr	Salaire	Bonus	Congés disp	Manager
d4713d60-c6	Green	William	M	19/06/1992	32	william.green	475-693-824	578 Michael	IT	11/04/2023	12/05/2023	0	Manager	CDD	36421	452	24	NONE
d3290a4c-b5	Moon	Christopher	M	31/01/1968	56	christopher.m	+1-909-375-2	714 Mann	Marketing	20/03/2022		2	Manager	CDD	22804	342	2	NONE
42930b33-a8	Lopez	Jacob	F	09/08/1983	41	jacob.lopez@	511.822.018	5159 Tom	IT	22/01/2022	03/11/2022	0	Manager	CDI	27983	3788	11	NONE
0b9475b1-3f	Brown	Krista	M	04/11/1992	32	krista.brown	001-389-810	90321 Clark	Qualité	30/10/2022		2	Manager	CDD	47506	1580	4	NONE
80ee526e-0f	Jones	Kevin	F	10/08/1968	56	kevin.jones@	209.416.345	419 Wade	R&D	18/03/2022	24/02/2023	1	Manager	CDI	16558	1238	7	NONE
0bb2c3f0-bd	Vazquez	Stephen	F	15/04/1974	50	stephen.vazq	-7023	109 Holly	Production	18/02/2022	07/07/2024	2	Manager	CDD	57216	4213	30	NONE
7cbd7025-e2	Brown	Jacqueline	M	13/02/1975	49	jacqueline.br	-6866	206 Stewart	Qualité	25/05/2020		4	Manager	CDD	38411	119	26	NONE
2ea60b99-fa	Banks	Morgan	M	09/12/1971	53	morgan.banks	476-710-471	0003 Grant	Finance et Co	14/08/2021	16/03/2024	3	Manager	CDD	26510	2016	2	NONE
0247145f-4a	Savage	Gregory	F	23/06/1993	31	gregory.savag	(224)255-51	456 Kelly	Service Client	26/11/2023		1	Manager	CDI	41925	3500	8	NONE
e89dc815-8f	Houston	Kevin	F	10/06/1984	40	kevin.houstor	475.851.717	13306 Corey	Marketing	08/08/2020	26/04/2022	2	Manager	CDI	37389	3023	29	NONE
2aa50f4e-c6	Richardson	Vanessa	F	19/08/1992	32	vanessa.richa	(235)511-08	76582	R&D	05/11/2021		3	Directeur de c	CDI	21795	3456	26	
a2939b3b-7f	Stark	Marcus	M	23/07/1997	27	marcus.starki	+1-438-893-(	18013 Billy	R&D	07/06/2024		0	Directeur de c	CDD	28872	1980	10	
138c3460-fd	Gregory	Linda	M	18/02/1966	58	linda.gregory	001-263-448	1744 Cruz	Marketing	09/04/2024	14/05/2024	0	Directeur de c	CDI	24583	1382	18	
b2ddc481-ac	Vasquez	Michelle	M	13/04/1992	32	michelle.vasq	(356)909-76	99251 Lee	Production	04/02/2024		0	Directeur de c	CDI	18450	3557	19	
d960af85-c9	Martinez	Felicia	M	02/11/1986	38	felicia.martin	(764)518-30	07408 Jerry	Finance et Co	22/12/2023	30/08/2024	1	Directeur de c	CDD	40101	1050	24	
27f9e728-c6	Walls	Robert	F	26/07/1992	32	robert.walls@	001-707-266	35192 Aaron	Service Client	24/10/2024		0	Directeur de c	CDI	22974	3020	3	
215203c7-42	Gibson	Thomas	F	02/07/1973	51	thomas.gibsc	4,354E+09	62235 Joel	R&D	24/12/2021		3	Directeur de c	CDD	51623	616	22	
6c148fc6-97	Evans	Robert	F	30/10/1972	52	robert.evans@	951-537-580	21418 Laura	IT	15/08/2020	29/05/2022	2	Chef de proje	CDD	29706	2875	17	

FIGURE 3.2 – Les données des employés

3.3 Implémentation du RAG

3.3.1 Indexation

Chunking sémantique des données

Cette étape a été appliquée au document PDF des politiques de ressources humaines. Nous avons procédé comme suit :

- Divisé le document en phrases, en utilisant des séparateurs comme le point (”.”), le point d'interrogation (” ?”) ou le point d'exclamation (” !”).

- Transformé les phrases en représentations vectorielles.
- Comparé chaque phrase avec ses phrases voisines en calculant la distance de similarité via la *cosine similarity*.

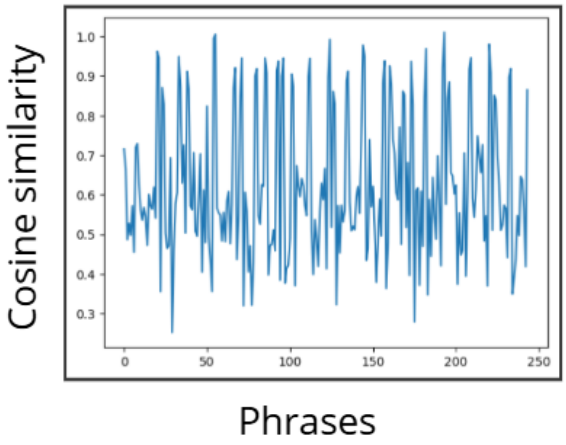

FIGURE 3.3 – visualisation des distances entre les phrases

- Regroupé les phrases similaires ayant une distance faible ensemble dans un même chunk.
- Observé des sections avec des distances plus petites, suivies de zones avec des distances plus grandes, correspondant aux valeurs aberrantes (*outliers*). Les *outliers* marquent la frontière des chunks. Toute distance supérieure au 95 percentile a été considérée comme un point de rupture pour former les chunks.

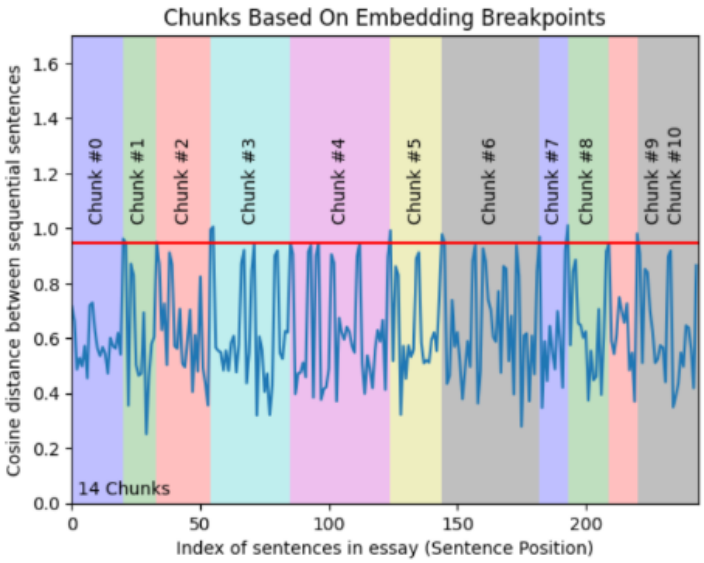

FIGURE 3.4 – visualisation des chunks obtenus

Génération de résumés des chunks

Nous avons généré des résumés pour chaque chunk en utilisant le modèle **LLAMA 3.2**. Ces résumés condensent les informations principales des chunks tout en conservant leur sémantique.

Transformation en embeddings vectoriels

Les chunks ainsi que leurs résumés ont été transformés en embeddings vectoriels à l'aide du modèle **all-MiniLM-L6-v2**. Ce modèle permet de capturer efficacement les relations sémantiques au sein des textes.

Stockage dans une base de données vectorielle

Les embeddings vectoriels, leurs résumés, ainsi que leurs métadonnées associées ont été stockés dans une base de données vectorielle, **ChromaDB**. Chaque chunk a été lié à son résumé et structuré de manière optimale pour les recherches ultérieures.

3.3.2 Routage

Recherche dans ChromaDB

La méthode **multivector-retriever** a été utilisée pour rechercher dans les vecteurs des résumés. Une recherche par similarité (*similarity search*) a permis d'identifier les chunks les plus pertinents et de retourner leurs contenus originaux.

Recherche dans le fichier CSV

Les requêtes des utilisateurs ont été converties en opérations sur les données du fichier CSV. Cela inclut des actions comme la recherche d'un employé ou la récupération d'une date d'embauche.

3.3.3 Récupération des données

- Recherche dans la base de données vectorielle pour extraire les informations pertinentes des documents textuels à l'aide de la *similarity search*.
- Traitement des requêtes sur les données du fichier CSV, contenant les informations des employés.

3.3.4 Génération

Pour générer la réponse finale à la question de l'utilisateur, nous avons utilisé le modèle **GPT-3.5 Turbo** via l'API **Azure OpenAI**. Ce modèle est capable de pro-

duire des réponses précises et contextuelles, en intégrant les résultats issus de l'étape de récupération des données.

3.3.5 Développement du chatbot

Le chatbot a été développé avec **Streamlit**, Cette solution a simplifié la mise en œuvre de l'interface utilisateur tout en offrant une expérience utilisateur intuitive.

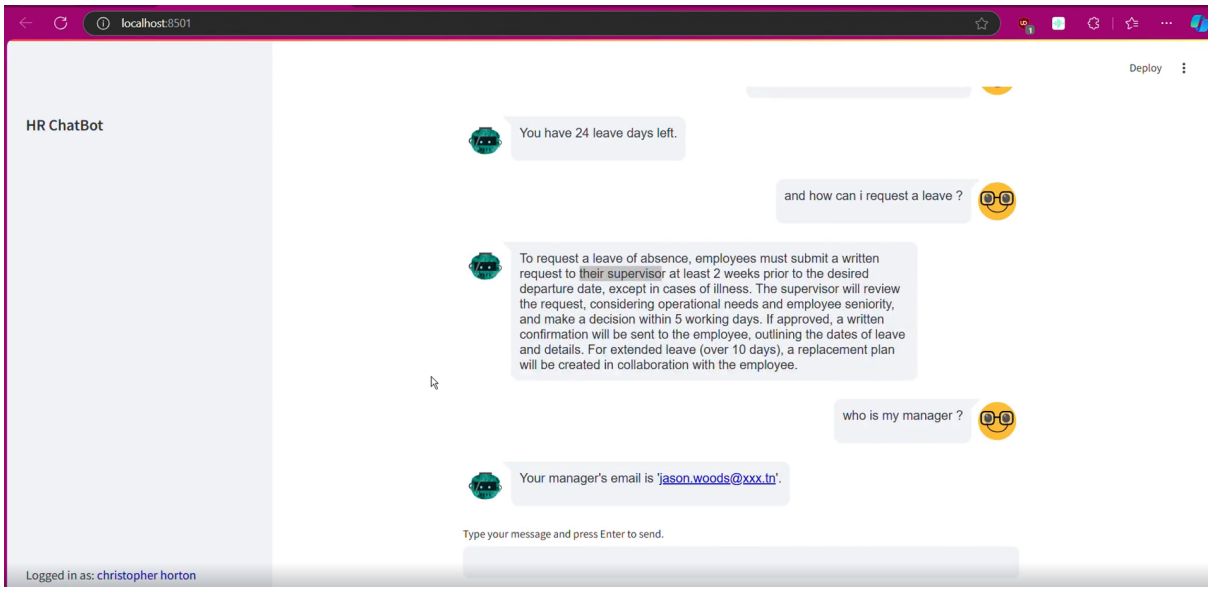

FIGURE 3.5 – Interface du chatbot

3.4 Implémentation du Fine-Tuning

3.4.1 Création du jeu de données d'entraînement

À l'aide du modèle **GPT-3.5 Turbo**, nous avons généré des paires de questions/réponses à partir des chunks obtenus lors de l'étape de *chunking sémantique*. Ces paires ont ensuite été transformées au format **DatasetDict**, adapté à l'entraînement du modèle.

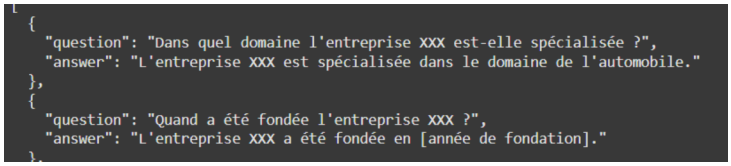

FIGURE 3.6 – Exemple des questions/réponses

3.4.2 Implémentation des techniques d'optimisation du Fine-Tuning

Nous avons utilisé la méthode **LoRA** (*Low-Rank Adaptation*) pour réduire la consommation des ressources pendant l'entraînement. En complément, nous avons appliqué une

quantification en **4 bits** grâce à la bibliothèque *BitsAndBytes*, optimisant ainsi les performances sans sacrifier la précision.

3.4.3 Entraînement du modèle LLM

Le fine-tuning a été effectué sur le modèle **Falcon 7B** en utilisant les hyperparamètres suivants :

- `per_device_train_batch_size` = 4
- `gradient_accumulation_steps` = 4
- `save_steps` = 10
- `learning_rate` = 2e-4
- `max_steps` = 200
- `fp16` = `True`

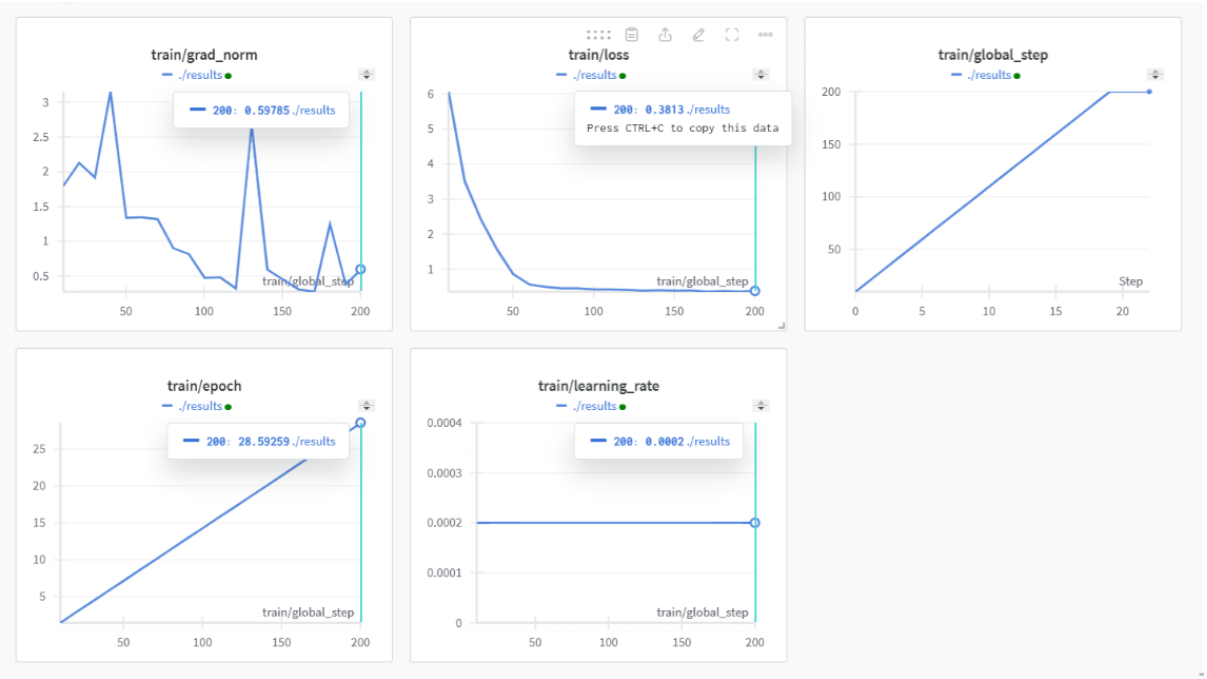

FIGURE 3.7 – Historique d'entrainement visualisé avec Weights and Biases

3.4.4 Résultats

Voici quelques exemples de réponses générées par le modèle après le fine-tuning :

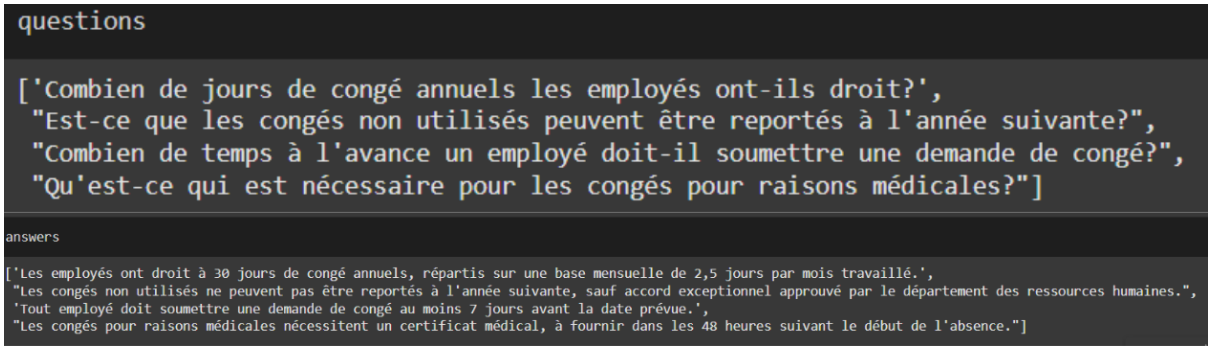

FIGURE 3.8 – Les résultats du Fine-Tuning

Conclusion

Dans ce chapitre, nous avons présenté en détail les outils utilisés ainsi que les différentes étapes nécessaires à la réalisation de ce projet. Nous avons abordé les aspects suivants : la génération des données, les étapes de la mise en œuvre du Retrieval-Augmented Generation (RAG) et le processus de *fine-tuning* du modèle. Ces étapes ont permis d'assurer une construction structurée et performante de notre solution, répondant aux objectifs définis initialement.

CHAPITRE 4

Évaluation et Comparaison

Introduction

Dans ce chapitre, nous allons analyser et comparer en détail les deux approches mises en œuvre dans ce projet : le Retrieval-Augmented Generation (RAG) et le *fine-tuning*. Ces deux méthodologies, bien qu'ayant des objectifs communs, diffèrent par leur architecture, leurs exigences en termes de ressources, et leurs comportements face à différents types de requêtes.

4.1 L'approche de la comparaison

Pour effectuer une évaluation comparative rigoureuse, nous avons structuré les données nécessaires en organisant des fichiers CSV contenant :

- Les questions posées,
- Les réponses générées par le modèle GPT 3.5 Turbo (référence),
- Les réponses produites par les approches RAG et *fine-tuning*.

L'évaluation a été réalisée en calculant la distance sémantique entre chaque réponse générée par les deux approches (RAG et *fine-tuning*) et la réponse de référence issue du modèle GPT 3.5 Turbo. Cette distance permet de mesurer la similitude entre les réponses et d'estimer leur pertinence et leur exactitude par rapport à la réponse idéale.

Nous avons utilisé la distance sémantique qui assure une évaluation objective et cohérente, indépendamment de la variation textuelle entre les réponses.

Enfin, les résultats de cette comparaison seront présentés sous forme de tableaux, permettant de comprendre les performances relatives des deux approches dans différents scénarios liés au service des ressources humaines.

4.2 Résultats et Interprétation

Les résultats des deux approches sont présentés dans les tableaux ci-dessous, qui comparent les scores de similarité cosinus entre les réponses générées par les approches RAG et fine-tuning et les réponses attendues.

4.2.1 Tableau des résultats

Le tableau suivant montre une comparaison des scores moyens de similarité cosinus pour chaque approche, ainsi que les scores extrêmes (le meilleur et le pire score obtenus).

Méthode	Nombre de Questions	Score Moyen	Meilleur Score	Pire Score
RAG	29	0.90	0.99	0.76
Fine-tuning	29	0.93	0.98	0.89

TABLE 4.1 – Comparaison des scores entre les approches RAG et Fine-tuning

4.2.2 Analyse des résultats

À partir des résultats obtenus, il apparaît que l'approche fine-tuning a surpassé l'approche RAG en termes de performance globale, avec un score moyen de 0.93 contre 0.90 pour le RAG. Cette supériorité du fine-tuning indique qu'il a été plus efficace pour produire des réponses proches des réponses de référence, notamment grâce à son adaptation spécifique au domaine ciblé lors de l'entraînement.

Cependant, l'approche RAG a également montré des performances intéressantes, avec des scores comparables au fine-tuning dans certains scénarios. En particulier, RAG s'est avéré performant dans les cas où les réponses nécessitaient une recherche contextuelle ou une consultation directe des documents indexés, démontrant ainsi son utilité pour des applications nécessitant un accès dynamique à une base documentaire. Ces résultats soulignent la complémentarité potentielle des deux approches selon les besoins spécifiques de l'application.

4.2.3 Exemples de cas extrêmes

Le tableau ci-dessous présente quelques exemples de questions avec leurs réponses générées par les deux approches, ainsi que les scores associés.

Question	Réponse Attendue	Réponse RAG	Réponse Fine-tuning	Score RAG	Score Fine-tuning
Comment les programmes de formation sont-ils ajustés ?	Basés sur les retours et l'évolution du marché.	pour les programmes de formations, ils sont ajustés grâce aux retours et enquêtes régulières.	les programmes de formation sont Basés sur les retours et l'évolution du marché.	0.90	0.92

TABLE 4.2 – Exemple de questions/réponses générées par les deux approches, et leurs scores

Comme on peut le voir, l'approche fine-tuning a produit une réponse presque identique à la réponse attendue pour la question "Comment les programmes de formation sont-ils ajustés ?", ce qui a donné un score élevé de 0.92.

Dans le cas de l'approche RAG, bien que la réponse à "Comment demander un congé ?" soit plus pertinente et correct, elle reste moins exacte que celle produite par fine-tuning, avec un score de 0.90.

4.2.4 Conclusion de l'analyse

L'analyse des résultats montre que, dans l'ensemble, l'approche fine-tuning surpasse légèrement l'approche RAG en termes de similarité sémantique avec les réponses attendues. Cependant, cette différence de performance dépend largement du type de question et du contexte d'application. L'approche fine-tuning peut offrir des réponses satisfaisantes dans des scénarios où le contexte documentaire est moins crucial.

Conclusion

Ce dernier chapitre a été consacré à l'évaluation du système développé dans le cadre de ce projet, en mettant en avant les deux approches implémentées : la méthode basée sur le RAG et celle utilisant le *fine-tuning*. Une comparaison approfondie a été réalisée entre ces deux approches, accompagnée d'interprétations des résultats obtenus. Cette analyse a permis de mieux comprendre les performances de chaque méthode dans le contexte des besoins spécifiques du service des ressources humaines.

Conclusion

Ce travail s'inscrit dans le cadre d'un projet tutoré réalisé en collaboration avec ACTIA Engineering Services. L'objectif principal était de développer un chatbot RH intelligent capable d'automatiser les processus des ressources humaines. Ce projet a été structuré en plusieurs phases clés, incluant la conception, l'implémentation et l'évaluation du système.

Pour concrétiser ce projet, nous avons commencé par une analyse approfondie des besoins spécifiques de l'entreprise en matière de gestion des ressources humaines. Nous avons ensuite généré des données simulées, comprenant des politiques de ressources humaines et des informations fictives sur les employés, nécessaires pour le développement et le test de notre solution.

Nous avons mis en œuvre les différentes étapes du système RAG (Retrieval-Augmented Generation), notamment l'indexation des données, le routage, la récupération de l'information, ainsi que la génération de réponses. Parallèlement, nous avons procédé au *fine-tuning* du modèle Falcon 7B en utilisant des données questions-réponses générées avec le modèle GPT 3.5 Turbo. Enfin, nous avons collecté et comparé les réponses produites par les deux approches, ce qui nous a permis d'évaluer leurs performances respectives.

Ce projet a démontré l'importance des techniques avancées d'intelligence artificielle et de traitement du langage naturel dans l'optimisation des processus RH. L'intégration de ces technologies permet non seulement de répondre rapidement et efficacement aux questions des employés, mais également de réduire considérablement la charge de travail des équipes RH. Les résultats obtenus offrent des perspectives intéressantes pour l'amélioration et l'élargissement de l'usage de ce chatbot dans des environnements professionnels variés.