Université Mohammed V - Rabat
École Nationale Supérieure d'Informatique et d'Analyse des Systèmes

Filière

Business Intelligence and Analytics (BI&A)

Rapport de Projet de Fin d'Année (1A)

Sujet :

LISAN : Une Conversation Vocale mieux qu'une Lecture

Réalisé par :

Ouahib Salma

Tayi Anouar

Encadré par :

M. Aouragh Si lhoussain

Membre de Jury :

M. Aouragh Si lhoussain

M. Benhiba Lamia

Année Universitaire 2023-2024

Dédicace :

Au Dieu tout-puissant notre créateur

À nos chères familles
Pour leurs sacrifices sans limites

À nos chers enseignants
Pour leurs soutiens et encouragements

À nos chères encadrantes
Pour leur soutien et consignes

À nos amis,
Pour leurs aides et leurs points de vue

Remerciements

Nous laissons notre plume écrire pour remercier le Dieu qui nous a laissé vivre jusqu'à ce moment, et nous a aidé durant notre année d'étude, pour dire un grand merci pour nos chers parents, qui lèvent les mains à chaque prière pour nous souhaiter la réussite et le bonheur, qui ont apporté la torche du sacrifice pour éclaircir notre chemin, qui ont fortifié notre volonté.
Aussi, nous adressons à nos chers professeurs que nous rencontrons dans notre première année, ceux qui sacrifient le temps pour nous aider, et nous donner des conseils, dans notre vie professionnelle, et nous aider à développer nos compétences. Un grand respect de notre part à vous.
Ainsi, nous remercions très chaleureusement notre encadrant ***M. Aouragh Si Lhoussain***, un grand Merci, et un grand respect, pour nous est aidé de réaliser ce projet.
Nous tenons à remercier sincèrement Monsieur ***Pr. Aouragh Si Lhoussain*** et***Pr. Benhiba Lamia*** pour leur temps précieux et leur expertise en tant que membres du jury.

Résumé

Le rapport présenté détaille toutes les étapes réalisées pour la concrétisation de ce projet. Il explore la motivation et les objectifs de notre initiative. De plus, il décrit en détail les outils et les technologies implémentés. Il expose également le processus allant de la recherche des modèles pré-entraînés à la réalisation de l'interface utilisateur.

Notre projet consiste en la création d'un chatbot capable de fournir des réponses vocales basées sur un livre donné. L'utilisateur pose une question vocale, et le chatbot utilise le contenu du livre pour formuler une réponse pertinente.
La motivation derrière ce projet est de faciliter l'accès à l'information contenue dans les livres par le biais de la technologie vocale. L'objectif principal est de créer un outil interactif qui permet aux utilisateurs de poser des questions vocales et de recevoir des réponses précises et informatives, basées sur le contenu d'un livre spécifique.
Pour réaliser ce projet, nous avons utilisé plusieurs outils et technologies. Le chatbot repose sur des modèles de traitement du langage naturel (NLP) avancés, pour comprendre et générer du texte.
La première étape du développement a consisté à rechercher et sélectionner des modèles NLP pré-entraînés capables de comprendre et de répondre à des questions basées sur un livre. Nous avons exploré différents modèles pour trouver celle qui offrait la meilleure précision et pertinence pour notre application spécifique.
Une fois les modèles sélectionnés, nous les avons intégrés dans notre système. Cela a impliqué de former les modèles supplémentaires sur le contenu spécifique du livre pour garantir que les réponses générées soient pertinentes et précises. Ensuite, nous avons développé l'interface utilisateur, qui permet aux utilisateurs de poser des questions vocales et de recevoir des réponses vocales. L'interface a été conçue pour être intuitive et facile à utiliser, offrant une expérience utilisateur fluide.

En conclusion, ce projet de chatbot vocal basé sur un livre combine des technologies avancées de traitement du langage naturel et de reconnaissance vocale pour offrir une nouvelle façon d'interagir avec le contenu des livres. Il représente une avancée significative dans l'accessibilité et l'utilisation de l'information textuelle à travers des interfaces vocales.

Liste des abréviations

IA	*Intelligence Artificielle*
ML	*Machine Learning*
DL	*Deep Learning*
NLP	*Natural Language Processing*
TAL	*Traitement Automatique de la Langue*
BERT	*Bidirectional Encoder Representations from Transformers*
API	*Application Programming Interface*

Table des figures

Table des matières

Introduction générale

Nous vivons dans une époque où l'intelligence artificielle est devenue indispensable dans notre vie quotidienne. Tout le monde utilise l'intelligence artificielle pour écrire des articles, rédiger des lettres, corriger des phrases, etc. Les modèles d'intelligence artificielle répondent à divers besoins humains, mais la plupart de ces modèles traitent principalement le français et l'anglais.

Cependant, les personnes intéressées par la langue arabe, les livres, les romans et les recherches en langue arabe rencontrent des difficultés, car il y a très peu de recherches dans ce domaine. De plus, la nature des personnes change avec les technologies : elles ne veulent plus faire beaucoup d'efforts pour obtenir des informations, mais préfèrent y accéder rapidement.

Notre sujet porte sur le deep learning pour le traitement de la langue arabe. La langue arabe est sous-estimée par rapport au français et à l'anglais. Nous trouvons plusieurs modèles en intelligence artificielle efficaces pour le français et l'anglais, mais il est nécessaire de développer un modèle orienté vers l'arabe.

De plus, il y a peu de personnes intéressées par les livres en arabe. L'un des objectifs de notre modèle est d'interagir avec les utilisateurs sur des livres en arabe et de faciliter les recherches en fournissant rapidement des informations essentielles. Par exemple, les étudiants ou les doctorants ont besoin des points essentiels d'un livre de 1000 pages rapidement et efficacement.

Chapitre 1

Contexte et besoin du projet

1.1 Introduction

Dans ce premier chapitre, nous posons les bases de notre projet en détaillant ses objectifs, en explorant la problématique qu'il vise à résoudre, et en présentant la structure que nous avons adoptée. Notre initiative se situe dans le domaine du traitement de la langue arabe, un domaine en pleine expansion mais encore largement sous-exploré en comparaison avec d'autres langues majeures comme l'anglais ou le chinois. Ce chapitre constitue ainsi une introduction essentielle pour comprendre le cadre et les enjeux de notre initiative.

Le traitement automatique du langage naturel (NLP) a connu des avancées significatives grâce à l'application de techniques de deep learning. Cependant, les ressources et les modèles disponibles pour la langue arabe restent relativement limités. Notre projet vise à combler cette lacune en développant des solutions innovantes et efficaces pour le traitement de la langue arabe, tout en assurant une interaction utilisateur optimisée et intuitive.

1.2 Motivations et problématique du projet

Notre projet est motivé par la reconnaissance d'un besoin crucial dans le domaine du traitement de la langue arabe. Malgré les progrès technologiques réalisés dans le NLP, la langue arabe n'a pas bénéficié de la même attention que d'autres langues. Ce manque de modèles robustes et spécifiques pour l'arabe entraîne des difficultés pour les utilisateurs cherchant à accéder à des informations précises et de qualité. La complexité morphologique et syntaxique de l'arabe ajoute un défi supplémentaire, nécessitant des solutions sur mesure et adaptées.

1.3 Objectifs

Pour répondre à ces défis, notre projet se fixe plusieurs objectifs clés :

1. Amélioration des Modèles de Deep Learning : Développer et optimiser des modèles de deep learning spécialement conçus pour le traitement de la langue arabe. Cela inclut la gestion des particularités linguistiques de l'arabe, telles que la richesse morphologique et les variations dialectales.

2. Accès Simplifié aux Données : Créer des outils qui facilitent l'accès aux données et aux informations contenues dans les livres et la recherche en langue arabe. Cela inclut la numérisation et l'indexation de ces ressources pour un accès plus rapide et plus intuitif.

3. Intégration de Systèmes Coherents : Concevoir une solution intégrée qui combine plusieurs systèmes pour offrir une interaction fluide et efficace. Nous avons intégré trois systèmes principaux : - Un système de gestion de documents en langue arabe. - Un système de traitement des questions vocales posées par les utilisateurs. - Un système de génération de réponses précises et pertinentes.

4. Qualité et Efficacité des Réponses : Assurer que les réponses générées par notre modèle soient de haute qualité et répondent aux attentes des utilisateurs. Cela inclut l'amélioration continue des algorithmes de traitement pour offrir des réponses de plus en plus précises et pertinentes.

5. Expérience Utilisateur Optimisée : Mettre en place une interface utilisateur intuitive et conviviale qui permet une interaction naturelle et efficace avec le système, que ce soit à travers des questions vocales ou des requêtes textuelles.

1.4 Description du projet

Pour créer un modèle de deep learning capable d'interagir avec les utilisateurs de manière vocale sous forme de question-réponse, vous aurez besoin de plusieurs composants et algorithmes de deep learning. Voici les principaux éléments et algorithmes nécessaires pour réaliser ce projet : 1. Reconnaissance vocale (Speech Recognition) La reconnaissance vocale est la technologie qui permet de convertir la parole en texte. Cette technologie permet à un ordinateur de comprendre et de transcrire les mots prononcés par un utilisateur. Elle est essentielle pour les interactions vocales, car elle constitue la première étape où les données vocales sont transformées en une forme que les systèmes de traitement du langage naturel peuvent analyser.

2. Traitement du langage naturel (Natural Language Processing - NLP) Le traitement du langage naturel (NLP) est un domaine de l'intelligence artificielle qui se concentre sur l'interaction entre les ordinateurs et les langues humaines. Le NLP permet aux machines de lire, comprendre et déduire le sens du texte. Il comprend diverses techniques pour analyser le langage, telles que la tokenization (découpage du texte en unités plus petites), l'analyse syntaxique (compréhension de la structure grammaticale) et l'analyse sémantique (compréhension du sens).

3. Systèmes de question-réponse (Question-Answering Systems) Les systèmes de question-réponse sont des applications du traitement du langage naturel qui prennent une question en langage naturel en entrée et fournissent une réponse pertinente. Ces systèmes sont conçus pour comprendre le contexte et fournir des réponses précises en recherchant des informations dans des bases de données ou en générant des réponses basées sur des modèles linguistiques pré-entraînés.

4. Intégration et Interface Utilisateur L'intégration et l'interface utilisateur concernent la manière dont les différentes technologies et algorithmes sont combinés et présentés à l'utilisateur final. Une interface utilisateur bien conçue permet aux utilisateurs d'interagir facilement avec le système. L'intégration assure que les différents composants, tels que la reconnaissance vocale, le traitement du langage naturel, et la synthèse vocale, fonctionnent ensemble de manière harmonieuse pour offrir une expérience utilisateur fluide et efficace.

1.5 Conclusion

En conclusion, dans ce chapitre introductif, nous avons jeté les bases de notre projet en identifiant nos motivations principales et en décrivant la solution que nous proposons : le développement d'un modèle de deep learning capable d'interagir avec les utilisateurs de manière vocale, sous forme de questions-réponses, en langue arabe. Les technologies de reconnaissance vocale, de traitement du langage naturel, de systèmes de question-réponse et de synthèse vocale sont essentielles pour réaliser ce projet. Nous explorerons chacune de ces technologies plus en détail dans les chapitres suivants, où nous spécifierons les algorithmes et les modèles que nous utiliserons pour les implémenter.

Chapitre 2

Les axes du projet

2.1 Introduction

Dans ce chapitre, nous abordons en détail les différents modèles que nous avons choisis pour la réalisation de notre projet. Ces modèles sont tous des modèles préentraînés disponibles sur la plateforme Hugging Face, une ressource reconnue pour la diversité et la qualité de ses modèles de traitement du langage naturel. Les modèles que nous avons sélectionnés sont le fruit du travail de grandes entreprises technologiques et de laboratoires de recherche renommés. Ils ont été entraînés sur des milliards de données, ce qui garantit leur robustesse et leur capacité à gérer des tâches complexes de traitement de la langue arabe.

L'utilisation de ces modèles préentraînés présente plusieurs avantages. Tout d'abord, elle permet de tirer parti des avancées technologiques récentes sans avoir à investir des ressources considérables dans l'entraînement de nouveaux modèles à partir de zéro. De plus, ces modèles ont déjà prouvé leur efficacité dans divers contextes et applications, ce qui nous assure une base solide sur laquelle construire notre solution. Enfin, l'accès à une large gamme de modèles sur Hugging Face nous offre la flexibilité de tester et de comparer différentes approches pour identifier celles qui répondent le mieux à nos besoins spécifiques.

Dans les sections suivantes, nous présenterons chaque modèle en détail, en expliquant les raisons de notre choix, les caractéristiques techniques de chaque modèle, et comment nous comptons les adapter pour le traitement de la langue arabe dans notre projet. Cette analyse approfondie nous permettra de poser les fondations techniques nécessaires pour le développement de notre système interactif de questions-réponses en langue arabe.

2.2 Axe 1: Traitement Automatique de la Langue Naturelle

Le traitement du langage naturel (NLP) est un domaine de l'intelligence artificielle qui se concentre sur l'interaction entre les ordinateurs et les humains à travers le langage naturel. L'objectif du NLP est de permettre aux ordinateurs de comprendre, interpréter et répondre au langage humain de manière à la fois utile et naturelle. On a différents traitements du NLP :

1. Tokenisation : Le processus de découpage du texte en unités plus petites appelées tokens (mots, phrases, ou symboles). Par exemple, la phrase "Le chat dort" peut être tokenisée en ["Le", "chat", "dort"].

2. Lemmatisation et racinisation : La lemmatisation réduit les mots à leur forme de base ou lemme, par exemple, "manges", "mangé" et "manger" deviennent "manger". La racinisation (stemming) est une méthode plus agressive qui coupe les suffixes pour obtenir la racine du mot.

3. Étiquetage grammatical (Part-of-Speech Tagging) : Attribution d'une étiquette grammaticale à chaque token, par exemple, déterminer si un mot est un nom, un verbe, un adjectif, etc.

4. Reconnaissance d'entités nommées (NER) : Identification et classification des entités nommées dans le texte, comme les noms de personnes, d'organisations, de lieux, etc.

5. Analyse syntaxique (Parsing) : Analyse de la structure grammaticale du texte, construisant un arbre syntaxique qui représente la structure des phrases.

6. Analyse sémantique : Compréhension du sens des mots et des phrases, y compris la désambiguïsation des mots, l'analyse des relations entre les mots, et la compréhension des significations implicites.

7. Analyse des sentiments : Détection et classification des émotions exprimées dans un texte, par exemple, déterminer si un commentaire est positif, négatif ou neutre.

8. Traduction automatique : Conversion automatique du texte d'une langue à une autre.

9. Résumé automatique : Génération d'un résumé concis à partir d'un texte plus long. L'input dans le NLP est généralement un texte brut, qui peut être sous forme de phrases, de paragraphes, de documents, ou même de données conversationnelles. Par exemple, un

utilisateur peut entrer une question, une commande vocale, un document à analyser, etc. L'output dépend de la tâche NLP spécifique. Par exemple : - Pour une tâche de traduction automatique, l'output sera le texte traduit dans la langue cible. - Pour une tâche d'analyse des sentiments, l'output sera une étiquette indiquant le sentiment (positif, négatif, neutre). - Pour une tâche de question/réponse, l'output sera la réponse à la question posée. - Pour une tâche de résumé automatique, l'output sera un résumé du texte fourni. - Pour une tâche de reconnaissance vocale, l'output sera la transcription textuelle du discours.
Les modèles NLP sont utilisés dans une variété de contextes, y compris :

- Chatbots et assistants virtuels : Pour comprendre et répondre aux requêtes des utilisateurs. - Moteurs de recherche : Pour améliorer la pertinence des résultats de recherche. - Réseaux sociaux : Pour modérer le contenu, analyser les tendances et surveiller les sentiments. - Service client : Pour automatiser les réponses aux questions fréquentes et améliorer l'efficacité du support. - Applications de traduction : Pour permettre la communication entre personnes parlant différentes langues. - Analyse de documents : Pour extraire des informations clés et résumer des documents volumineux.
En conclusion, le traitement du langage naturel est un domaine vaste et complexe avec de nombreuses applications pratiques qui améliorent notre interaction avec la technologie et facilitent l'automatisation de nombreuses tâches liées au langage.

2.3 Axe 2: Reconnaissance vocale

2.3.1 Autour de la reconnaissance vocale :

La reconnaissance vocale est une technologie qui permet de convertir la parole humaine en texte ou de comprendre les commandes vocales. Elle repose sur des algorithmes sophistiqués de traitement du signal et de l'apprentissage automatique. Cette technologie a de nombreuses applications, allant des assistants vocaux comme Siri et Alexa à la transcription de réunions ou de conférences.
L'entrée dans un système de reconnaissance vocale est le signal vocal, qui est capturé via un microphone. Ce signal vocal est une onde sonore, qui est ensuite convertie en un format numérique pour être traité par l'ordinateur. Le processus de numérisation implique l'échantillonnage de la voix à des intervalles réguliers et la conversion de ces échantillons en une série de valeurs numériques.
Le signal vocal numérique est ensuite segmenté en fenêtres de temps plus petites, souvent de 20 à 40 millisecondes, pour une analyse plus fine. Chaque segment est transformé en un spectre de fréquences à l'aide de techniques comme la Transformée de Fourier Rapide (FFT). Les caractéristiques acoustiques (comme les coefficients cepstraux en fréquences mél (MFCC)) sont extraites de ces segments pour représenter les informations phonétiques du signal vocal.
La sortie d'un système de reconnaissance vocale est généralement du texte. Une fois les caractéristiques acoustiques extraites, elles sont passées à travers un modèle de reconnaissance vocale qui les mappe sur les phonèmes correspondants. Ce modèle utilise souvent un réseau de neurones ou un modèle statistique pour prédire les séquences de phonèmes.

Les phonèmes sont ensuite assemblés en mots à l'aide d'un modèle de langage, qui prend en compte les probabilités des différentes séquences de mots en fonction de la langue naturelle. Enfin, ces mots sont assemblés pour former des phrases complètes. Le texte final est ensuite fourni en sortie, pouvant être utilisé pour diverses applications comme la transcription automatique ou la réponse à des commandes vocales.

2.3.2 Exemples de modèle

Voici des modèles Célèbres de Reconnaissance Vocale sont :

- ***Hidden Markov Models (HMM) :*** Les modèles de Markov cachés (HMM) ont été parmi les premiers modèles utilisés dans la reconnaissance vocale. Ils reposent sur l'idée que la parole peut être modélisée comme une séquence de symboles discrets (phonèmes) générée par un processus stochastique caché. Les HMM sont efficaces pour capturer les variations temporelles et séquentielles de la parole, mais ils ont des limites en termes de précision et de gestion des longues dépendances.
- ***Deep Neural Networks (DNN) :*** Avec l'avènement des réseaux de neurones profonds (DNN), la reconnaissance vocale a connu une amélioration significative. Les DNN peuvent modéliser des relations complexes et capturer des caractéristiques plus abstraites des données vocales. Les architectures comme les réseaux de neurones récurrents (RNN) et les réseaux de neurones convolutionnels (CNN) sont couramment utilisées pour la reconnaissance vocale.
- ***Transformer Models :*** Les modèles Transformers, comme BERT et GPT, ont révolutionné le traitement du langage naturel, y compris la reconnaissance vocale. Ces modèles utilisent des mécanismes d'attention pour traiter l'information contextuelle de manière plus efficace que les RNN traditionnels. Ils sont capables de gérer des séquences de grande longueur et de fournir des prédictions plus précises et contextuellement pertinentes.
- ***Whisper d'OpenAI :*** Whisper est un modèle de reconnaissance vocale développé par OpenAI, basé sur l'architecture Transformer. Il est conçu pour être particulièrement robuste aux variations de l'environnement acoustique et à la diversité des accents. Whisper utilise une grande quantité de données de parole pour s'entraîner, ce qui lui permet de généraliser efficacement à de nouvelles voix et de nouveaux contextes. Whisper se distingue par sa capacité à traiter des langues multiples et à fournir des transcriptions précises même dans des conditions de bruit ambiant. En utilisant une architecture Transformer, Whisper peut capturer des relations complexes dans les données vocales et fournir des résultats de haute qualité pour diverses applications de reconnaissance vocale.

2.3.3 Le modèle whisper :

Nous avons choisi Whisper, développé par OpenAI, car c'est un modèle de reconnaissance vocale particulièrement performant. Whisper est conçu pour être robuste aux variations acoustiques et à la diversité des accents. Il peut gérer des enregistrements de qualité variable et fonctionner efficacement même dans des environnements bruyants. Cette robustesse est cruciale pour les applications du monde réel, où les conditions d'enregistrement ne sont pas toujours idéales.
Whisper est capable de traiter plusieurs langues, ce qui le rend utile pour des applications internationales. Il peut reconnaître et transcrire la parole dans diverses langues, facilitant ainsi la communication et l'accessibilité globale. Grâce à son architecture avancée basée sur des modèles Transformers, Whisper offre une précision de transcription élevée. Il utilise des mécanismes d'attention pour comprendre le contexte et produire des transcriptions précises, même pour des discours rapides ou des phrases complexes.
Whisper est entraîné sur une grande variété de données vocales, ce qui lui permet de s'adapter à différents contextes et domaines d'application. Que ce soit pour la transcription de réunions, de podcasts, ou d'interviews, Whisper est capable de fournir des résultats cohérents et pertinents.Ainsi qu'il peut traiter les données vocales en temps réel, ce qui est essentiel pour les applications interactives comme les assistants vocaux et les systèmes de commande vocale. Cette capacité permet une interaction fluide et immédiate avec les utilisateurs. Ilutilise des techniques de modélisation du langage pour corriger automatiquement les erreurs potentielles dans les transcriptions. Cela inclut la correction des mots mal transcrits en fonction du contexte et des règles grammaticales, améliorant ainsi la qualité globale des transcriptions.
Le modèle Whisper est conçu pour être facilement intégré dans diverses applications et systèmes. Les développeurs peuvent l'utiliser via des API fournies par OpenAI, ce qui simplifie l'intégration de la reconnaissance vocale avancée dans les produits et services existants. Aussi qu'il traite les données vocales de manière sécurisée, respectant les normes de confidentialité. Cela est particulièrement important pour les applications sensibles où la sécurité des données utilisateur est une priorité.
On peut l'accéder a ce modèle via le lien : https://huggingface.co/openai/whisper-large-v3
Voici un exemple de l'utilisation (API disponible sur hugging face) :

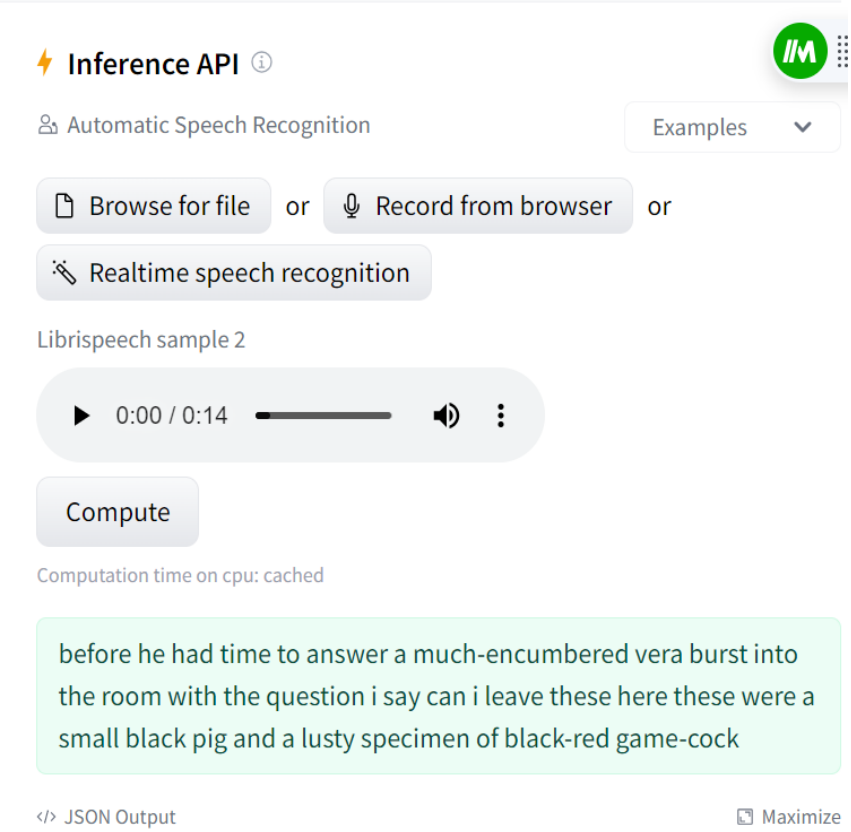

FIG. 2.1 : API de whisper disponibe sur hugging face

2.4 Axe 3: Système Question/Réponse :

2.4.1 Atour de système Q/R :

Les systèmes question/réponse (Q/R) en intelligence artificielle (IA) sont des technologies avancées conçues pour répondre aux questions posées en langage naturel par les utilisateurs. Ces systèmes sont devenus un élément essentiel dans divers domaines tels que le service client, l'éducation, la santé et la recherche d'informations.
Leur objectif principal est de comprendre la requête de l'utilisateur et de fournir une réponse pertinente et précise. Les systèmes Q/R reposent sur plusieurs technologies et techniques d'intelligence artificielle, parmi lesquelles : - Traitement du Langage Naturel (NLP) : Le NLP permet aux machines de comprendre, interpréter et générer le langage humain. Cela inclut la reconnaissance des mots et des phrases, l'analyse de la syntaxe et de la sémantique, ainsi que la compréhension contextuelle.

- Apprentissage Automatique (Machine Learning) : Les algorithmes d'apprentissage automatique permettent aux systèmes Q/R d'améliorer leurs performances au fil du temps. Ils utilisent des techniques de supervision et non-supervision pour apprendre à partir de grandes quantités de données textuelles.

- Bases de Connaissances et Graphes de Connaissances : Ces systèmes exploitent des bases de données structurées et des graphes de connaissances pour rechercher des infor-

mations pertinentes et établir des relations entre différents concepts.

- Réseaux de Neurones Profonds (Deep Learning) : Les modèles de deep learning, comme les réseaux de neurones convolutifs et récurrents, sont utilisés pour traiter des tâches complexes de compréhension et de génération de texte. Les transformateurs, tels que BERT et GPT, sont particulièrement efficaces pour les tâches Q/R.
L'input dans un système Q/R est généralement une question posée par un utilisateur. Les questions sont formulées en langage naturel, ce qui signifie qu'elles peuvent varier en termes de structure, de syntaxe et de vocabulaire. Il peut contenir des informations contextuelles implicites ou explicites. Les questions peuvent varier en complexité, allant de simples requêtes factuelles à des questions nécessitant une compréhension approfondie et une analyse contextuelle.
Dans certains cas, l'input peut également inclure des données multimodales, telles que des images, des sons ou des vidéos, qui nécessitent une analyse supplémentaire. Les questions peuvent varier en complexité, allant de simples requêtes factuelles à des questions nécessitant une compréhension approfondie et une analyse contextuelle.
L'output d'un système Q/R est la réponse générée en réponse à la question posée. Les caractéristiques principales de cet output incluent : - La précision et Pertinence :la réponse doit être précise et pertinente par rapport à la question posée. - Forme de la Réponse : Les réponses peuvent être formulées de différentes manières, en fonction de la question et du contexte. - Contextualisation : Pour les questions nécessitant un contexte, la réponse doit être contextualisée correctement.

- Richesse de l'Information : Les réponses peuvent inclure des informations additionnelles pour fournir une réponse plus complète.

- Format de la Réponse : Les réponses peuvent être sous différents formats, comme du texte, des listes, des graphiques, etc.

Le processus entre l'input et l'output dans un système Q/R implique plusieurs étapes :
Etape 1 : Compréhension de la Question en utilisant des techniques de traitement du langage naturel (NLP) pour analyser et comprendre la question.
Etape 2 : Recherche d'Informations : Consultation de bases de données, graphes de connaissances ou ressources externes pour trouver les informations pertinentes.
Etape 3 : Génération de la Réponse : Synthèse des informations trouvées et génération d'une réponse cohérente et appropriée.
Etape 4 : Affinement de la Réponse : Utilisation de modèles de deep learning et de feedback pour affiner et améliorer la précision des réponses fournies.

2.4.2 Exemples de modèles :

Les systèmes question/réponse (Q/R) sont conçus pour répondre à des questions posées en langage naturel. Voici quelques exemples notables de tels systèmes :

- ***IBM Watson :*** plateforme d'IA avancée capable de répondre à des questions complexes en utilisant des techniques de traitement du langage naturel et d'apprentissage automatique.Il est utilisé dans la médecine pour aider à diagnostiquer

des maladies et recommander des traitements en analysant des millions d'articles médicaux et des dossiers de patients.

- ***Google Assistant :*** Google Assistant : est un assistant virtuel qui répond à des questions posées par les utilisateurs et exécute des commandes vocales. Il fournit des informations sur la météo, la navigation, les événements, et bien plus, en utilisant les vastes ressources de données de Google.

- ***Siri d'Apple :*** assistant vocal intégré aux appareils Apple, capable de répondre à des questions et de réaliser des tâches basées sur des commandes vocales. Il envoie des messages, passe des appels, règle des alarmes, et répond à des questions factuelles ou contextuelles.

- ***Alexa d'Amazon :*** un assistant virtuel qui répond aux questions et contrôle les appareils domestiques intelligents. Il fournit des réponses sur les actualités, la météo, les sports, et contrôle des dispositifs domestiques comme les lumières et les thermostats.

- ***ChatGPT d'OpenAI :*** modèle de langage avancé capable de tenir des conversations en langage naturel et de répondre à une variété de questions. Il est utilisé pour l'assistance client, la création de contenu, et l'éducation, offrant des réponses détaillées et personnalisées aux questions posées par les utilisateurs.

- ***Llama-ChatQA :*** modèle de langage développé par Meta, conçu pour offrir des réponses précises et contextuelles en langage naturel. Il est utilisé pour diverses applications nécessitant des réponses conversationnelles sophistiquées, comme l'assistance client, l'éducation et les services interactifs en ligne. Llama-ChatQA peut répondre à des questions complexes, fournir des explications détaillées, et tenir des conversations prolongées avec un haut degré de compréhension contextuelle.

2.4.3 Le modèle Llama-ChatQA :

Llama-ChatQA est un modèle de langage avancé développé par Meta, conçu pour répondre à des questions et tenir des conversations en langage naturel.
Llama-ChatQA utilise des techniques de NLP de pointe pour comprendre et générer du langage humain de manière naturelle et contextuelle. Il Construit sur une architecture de réseau de neurones profonds avec des milliards de paramètres, permettant une compréhension fine des nuances du langage et une génération de texte cohérente. Il est entraîné sur un large éventail de données textuelles provenant de diverses sources, ce qui améliore sa capacité à répondre à des questions sur une multitude de sujets. Il a la capacité d'apprendre et de s'adapter continuellement à partir de nouvelles données, améliorant ainsi ses performances au fil du temps. Il a une excellente capacité à comprendre le contexte des questions posées, permettant de fournir des réponses précises et pertinentes. Ainsi qu'il est capable de générer des réponses détaillées et nuancées, adaptées à des questions complexes et spécifiques.
Nous avons choisis ce modèle car il a plusieurs avantages : 1. Précision et Pertinence : - Grâce à sa formation sur des données diversifiées et son architecture avancée, Llama-

ChatQA offre des réponses précises et pertinentes à un large éventail de questions.

2. Polyvalence : - Adapté à divers domaines d'application, allant de l'assistance client à l'éducation en passant par la recherche et le divertissement.

3. Interaction Naturelle : - Capacité à tenir des conversations prolongées avec les utilisateurs, offrant une expérience d'interaction plus naturelle et fluide.

4. Efficacité : - Réduit le besoin de rechercher manuellement des informations, en fournissant des réponses rapides et directes aux questions posées.

5. Amélioration Continue : - Les mises à jour régulières et l'apprentissage en continu permettent à Llama-ChatQA de rester à jour et d'améliorer ses performances.
On peut l'accéder a ce modèle via le lien : nvidia/Llama3-ChatQA-1.5-8B

2.5 Axe 4: Embeddings :

2.5.1 Autour de l'embeddings :

Les embeddings sont des représentations vectorielles continues de données discrètes, généralement des mots, des phrases ou des documents. Ils transforment des éléments textuels en vecteurs de nombres réels dans un espace vectoriel de dimension fixe, permettant ainsi de capturer des relations sémantiques et syntaxiques entre ces éléments.
L'input pour les embeddings est généralement du texte brut. Voici les étapes typiques pour préparer cet input :

1. Texte Brut : Le texte d'entrée peut être une phrase, un paragraphe ou un document entier. - Exemple : "Le chat noir dort sur le canapé."

2. Prétraitement : Le texte est souvent prétraité pour normaliser les données, ce qui peut inclure : - Tokenisation : Division du texte en unités de mots (tokens). - Exemple : ["Le", "chat", "noir", "dort", "sur", "le", "canapé"] - Lemmatisation ou Stemming : Réduction des mots à leur forme de base ou radicale. - Exemple : "dormir" au lieu de "dort" - Suppression des Stop Words : Élimination des mots courants sans signification importante (ex : "le", "sur").

3. Encodage : Transformation des tokens en un format numérique que le modèle peut traiter. Cela peut inclure l'utilisation de vocabulaire où chaque mot est associé à un identifiant unique.
L'output des embeddings est un vecteur de nombres réels qui représente chaque élément textuel dans un espace vectoriel de dimension fixe. Les caractéristiques de cet output incluent :

1. Vecteur de Dimension Fixe : Chaque mot ou token est représenté par un vecteur de dimensions fixées par le modèle d'embedding utilisé (ex : 100, 300, 768 dimensions). - Exemple : - "chat" -> [0.21, -0.14, 0.33, ..., 0.56] - "noir" -> [0.45, 0.22, -0.11, ..., 0.78]

2. Proximité Sémantique : Les vecteurs de mots sémantiquement similaires sont proches les uns des autres dans l'espace vectoriel. - Exemple : Les vecteurs pour "chat" et "chien" seront plus proches l'un de l'autre que les vecteurs pour "chat" et "voiture".

3. Contextualisation (pour les embeddings contextuels) : Dans les modèles contextuels comme BERT, chaque mot est représenté différemment selon le contexte dans lequel il apparaît. - Exemple : - ”bark” dans ”The dog barks” -> [0.34, 0.12, -0.45, ..., 0.67] - ”bark” dans ”The tree’s bark” -> [-0.22, 0.44, 0.11, ..., -0.31]

2.5.2 Exemples de modèles :

Voici des modèles Célèbres de L’embeddings qui sont utilisés partout dans le monde :

- ***Word2Vec :*** créer par Google, il utilise deux architectures, CBOW (Continuous Bag of Words) et Skip-Gram, pour apprendre des représentations vectorielles de mots en fonction de leur contexte dans des phrases., et il est utilisé pour Capturer les relations sémantiques entre les mots, utilisée dans la classification de texte, la recherche d’information et la recommandation de contenu.
- ***GloVe (Global Vectors for Word Representation) :*** créer par Stanford University, il est un modèle basé sur des statistiques globales des occurrences de mots dans un corpus, produisant des embeddings où les mots similaires sont proches dans l’espace vectoriel., et il est utilisé pour Analyse de texte, détection de sentiments, et toute application nécessitant une compréhension sémantique des mots.
- ***FastText :*** créer par Facebook AI Research, il améliore Word2Vec en prenant en compte les sous-mots (n-grams), ce qui permet de gérer les mots rares et les variantes morphologiques., et il est utilisé pour Traitement de langues avec une riche morphologie, comme l’allemand ou le turc, et applications nécessitant des performances rapides et efficaces.
- ***ELMo (Embeddings from Language Models) :*** créer par Allen Institute for AI, il produit des embeddings contextuels dynamiques en utilisant des réseaux de neurones bidirectionnels (BiLSTM), capturant la signification des mots en fonction de leur contexte., et il est utilisé pour Analyse de sentiments, classification de texte, et autres tâches de NLP nécessitant une compréhension contextuelle des mots.
- ***BERT (Bidirectional Encoder Representations from Transformers) :*** créer par Google, il utilise des Transformers bidirectionnels pour produire des embeddings contextuels riches, prenant en compte le contexte à gauche et à droite de chaque mot dans une phrase., et il est utilisé pour Question/réponse, classification de texte, et traduction automatique.
- ***all-MiniLM-L6-v2 :*** créer par Microsoft. C’est une version compacte de BERT, optimisée pour une performance élevée et une faible latence.

2.5.3 le modèle all-MiniLM-L6-v2:

all-MiniLM-L6-v2 est un modèle de type Transformer particulièrement optimisé pour produire des embeddings de phrases, développé par Microsoft. Voici quelques détails sur ce modèle :

- Architecture : Transformer
- Dimensions des Embeddings : 384
- Nombre de Couches : 6
- Taille du Modèle : MiniLM est une version compacte de BERT, optimisée pour une performance élevée et une faible latence.
Nous avons choisi d'utiliser ce modèle pour ces caractéristiques :
1.Compact et Efficace : Le modèle est conçu pour être plus léger et rapide, permettant une utilisation dans des environnements contraints en ressources, comme les appareils mobiles ou les applications en temps réel.
2.Embeddings de Phrases : Spécialisé dans la production d'embeddings pour des phrases entières, capturant le contexte global de la phrase, plutôt que des mots isolés.
3.Précision : Malgré sa petite taille, il maintient une haute précision dans les tâches de similarité de phrases et de classification de texte.

Il a plusieurs avantages :
1.Rapidité : Sa conception légère permet des calculs plus rapides, ce qui est crucial pour les applications nécessitant des réponses en temps réel.
2.Efficacité en Ressources : Adapté pour être utilisé dans des environnements avec des ressources limitées, comme les appareils mobiles ou les applications web en temps réel.
3.Polyvalence : Peut être utilisé dans diverses applications de NLP, telles que la recherche sémantique, le regroupement de phrases similaires, et l'appariement de paires de phrases.
On peut l'accéder a ce modèle via le lien :

2.6 Conclusion

En résumant tout ce que nous avons fait dans ce chapitre, nous avons abordé tous les éléments nécessaires ainsi que toutes les recherches effectuées avant d'entamer la réalisation de notre projet. Nous avons exploré le traitement du langage naturel (NLP), les systèmes d'embeddings, les systèmes de question/réponse, le traitement de la reconnaissance vocale, y compris l'expérimentation avec divers modèles en les utilisant dans différents contextes. Ce chapitre regorgeait d'informations et de recherches menées par des experts depuis des années, ce qui nous a permis d'identifier des modèles pertinents. C'était vraiment enrichissant et informatif.

Chapitre 3

Réalisation

3.1 Introduction :

Dans ce chapitre, nous abordons la mise en œuvre pratique de notre projet en fournissant une description détaillée des outils et des technologies employés, ainsi que du processus d'implémentation et d'évaluation de notre chatbot. Nous commencerons par présenter les différentes plateformes et bibliothèques que nous avons utilisées pour développer notre chatbot, en expliquant pourquoi ces choix technologiques sont appropriés pour notre projet. Ensuite, nous décrirons les étapes d'implémentation, depuis la conception de l'architecture du chatbot jusqu'à son déploiement, en soulignant les défis rencontrés et les solutions apportées. Enfin, nous exposerons les méthodes d'évaluation utilisées pour mesurer les performances de notre chatbot, y compris les critères d'évaluation, les tests effectués, et les résultats obtenus. Ce chapitre vise à fournir une compréhension complète et pratique du développement de notre chatbot, en mettant en lumière les aspects techniques et méthodologiques clés de notre travail.

3.2 Outils et technologies utilisés :

3.2.1 Outils utilisées :

Plateforme HuggingFace :

Hugging Face est une plateforme spécialisée dans l'intelligence artificielle et le traitement du langage naturel, offrant une vaste gamme d'outils et de bibliothèques open source pour les développeurs et les chercheurs. Parmi ses principaux avantages figurent l'accès à une multitude de modèles pré-entraînés, ce qui facilite le développement rapide de solutions basées sur l'IA, et une communauté active qui partage des ressources et des conseils. La plateforme propose également une interface conviviale pour héberger et

tester des modèles, ainsi que des API performantes pour intégrer facilement ces modèles dans diverses applications. De plus, Hugging Face promeut une culture de collaboration ouverte, permettant aux utilisateurs de contribuer et d'améliorer continuellement les modèles et les outils disponibles.

FIG. 3.1 : Hugging Face

Google Collab :

Google Colab, ou Google Collaboratory, est un service gratuit proposé par Google qui permet d'écrire et d'exécuter du code Python directement dans un navigateur, avec un accès facile à des ressources de calcul puissantes, notamment des GPU et des TPU. Conçu pour les chercheurs en machine learning, les scientifiques des données et les éducateurs, Google Colab simplifie le partage et la collaboration sur des projets de code, grâce à une intégration transparente avec Google Drive.
Il offre un environnement de développement interactif où les utilisateurs peuvent créer, partager et modifier des notebooks Jupyter, facilitant ainsi l'expérimentation et la visualisation des résultats. En outre, Colab préinstalle de nombreuses bibliothèques populaires, réduisant le temps nécessaire pour configurer des environnements de développement complexes, et permet d'importer des données à partir de diverses sources, y compris des API et des services de stockage cloud.

FIG. 3.2 : Google Colab

3.2.2 Technologies :

Python

Python est un langage de programmation de haut niveau, interprété et orienté objet, reconnu pour sa syntaxe simple et lisible, ce qui le rend accessible aux débutants tout en étant puissant pour les développeurs expérimentés. Créé par Guido van Rossum et publié en 1991, Python permet une écriture de code claire et concise, facilitant ainsi le développement rapide et la maintenance des logiciels. Il possède un écosystème riche de bibliothèques et de frameworks, comme NumPy et pandas pour la science des données, TensorFlow et PyTorch pour le machine learning, et Django pour le développement web.
Python est également apprécié pour sa grande communauté de développeurs et son utilisation dans des domaines variés, tels que l'automatisation, le développement web, la science des données, et plus encore.

Fig. 3.3 : Python

Les bibliothèques de python :

Nous avons installer plusieurs bibliothèques Python spécialisées dans le traitement du langage naturel, l'analyse de documents, et les modèles de machine learning, entre autres. Voici un aperçu des bibliothèques installées :

1. Langchain : Une bibliothèque conçue pour faciliter la création et la gestion de chaînes de traitement du langage naturel, permettant de combiner divers modèles et composants NLP de manière fluide.

2. pypdf : Une bibliothèque pour manipuler et traiter les fichiers PDF. Elle permet de lire, extraire du texte, et modifier des documents PDF

facilement.

3. unstructured : Cette bibliothèque aide à extraire et à organiser des données à partir de formats de documents non structurés comme des PDF, des HTML, et des fichiers texte, en transformant ces données en structures exploitables.

4. sentence_transformers : Utilisée pour créer et utiliser des modèles de transformation de phrases, cette bibliothèque permet d'obtenir des représentations vectorielles de phrases ou de textes, facilitant des tâches comme la recherche sémantique ou la classification de texte.

5. faisscpu : Un outil développé par Facebook AI Research pour la recherche et le clustering de vecteurs denses, FAISS est particulièrement efficace pour des tâches impliquant des grandes quantités de données vectorielles, comme la recherche de similarités.

6. tiktoken : Une bibliothèque spécialisée dans la tokenisation, importante pour le traitement des textes en unités plus petites comme des mots ou des sous-mots, souvent utilisée en préparation des données pour les modèles NLP.

7. auto_gptq : Un paquet pour l'utilisation d'AutoGPT-Q, un modèle de langage basé sur GPT qui peut être utilisé pour diverses tâches de génération de texte, souvent optimisé pour des performances spécifiques.

8. langchaincommunity : Une extension de Langchain, qui inclut des contributions et des outils supplémentaires développés par la communauté, enrichissant les capacités de traitement du langage naturel.

9. transformers : Une bibliothèque développée par Hugging Face qui fournit des implémentations prêtes à l'emploi de divers modèles de transformer pour des tâches NLP telles que la classification de texte, la génération de texte, et la traduction.

10. accelerate : Une bibliothèque pour faciliter le développement et l'exécution de modèles de machine learning, optimisant l'utilisation des ressources matérielles comme les GPU.

11. nvidiasmi : Un outil de ligne de commande de NVIDIA qui fournit des informations sur les GPU, y compris leur utilisation et leur état, ce qui est essentiel pour le suivi des performances dans les tâches de machine learning.

Les commandes supplémentaires installent deux bibliothèques Python utiles pour la génération et la manipulation de fichiers audio. Voici un aperçu de ces bibliothèques :

12. gtts (Google TexttoSpeech) : Cette bibliothèque permet de convertir du texte en discours en utilisant l'API Google Text-to-Speech. Elle est simple à utiliser et permet de générer des fichiers audio à partir de n'importe quel texte, ce qui est particulièrement utile pour les applications nécessitant une synthèse vocale, comme les chatbots vocaux, les livres audio, et les aides à l'accessibilité.

13. pydub : Une bibliothèque puissante pour manipuler des fichiers audio. Elle permet de lire, écrire, découper, fusionner, et convertir des fichiers audio entre différents formats. Pydub facilite également l'application d'effets audio comme le volume, les fondus enchaînés, et la gestion des segments audio, rendant le traitement audio en Python plus accessible et flexible.

Ces bibliothèques sont combinés pour créer, déployer et optimiser des applications complexes de traitement du langage naturel, en fournissant des fonctionnalités avancées pour l'extraction de texte, la transformation de données, et la gestion des modèles.

Gradio :

Gradio est une bibliothèque Python open-source qui permet de créer des interfaces utilisateur interactives pour les applications de machine learning de manière simple et rapide. Elle est compatible avec des bibliothèques populaires comme TensorFlow et PyTorch, et permet de transformer des fonctions Python en applications web en quelques lignes de code. Les interfaces sont hautement personnalisables et peuvent être partagées facilement via des liens. Gradio est idéal pour démontrer des modèles, tester des applications avec des utilisateurs finaux, et créer des outils éducatifs interactifs.

Points clés :

- Interfaces interactives pour ML
- Compatibilité avec TensorFlow, PyTorch, etc.
- Création simple et rapide
- Partage facile via liens

- Applications pour démonstrations, tests utilisateurs, éducation

FIG. 3.4 : Gradio

3.3 Réalisation :

3.3.1 Interface d'utilisateur : Application Web

3.3.1.1 L'acceuil : La page par defaut

Lorsque l'utilisateur visite notre application web pour la première fois, il est accueilli par une page lui demandant de téléverser un fichier à traiter et d'enregistrer un audio. (figure 3.5).

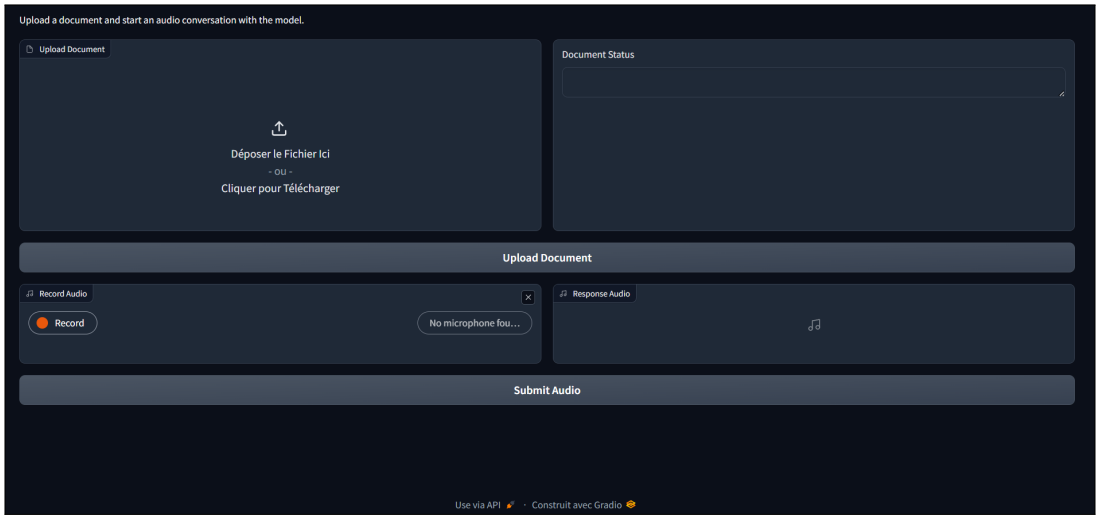

FIG. 3.5 : interface d'utilisateur

3.3.1.2 L'upload de documents

L'upload débute par un clic sur "Déposer le fichier ici", ensuite l'utilisateur choisit un fichier. Après la fin de l'importation, l'utilisateur clique sur "Upload Document" . Si le fichier est bien soumis il informe l'utilisateur sur l'état du fichier et maintenant le fichier est prêt à être utilisé. (figure 3.6 et 3.7).

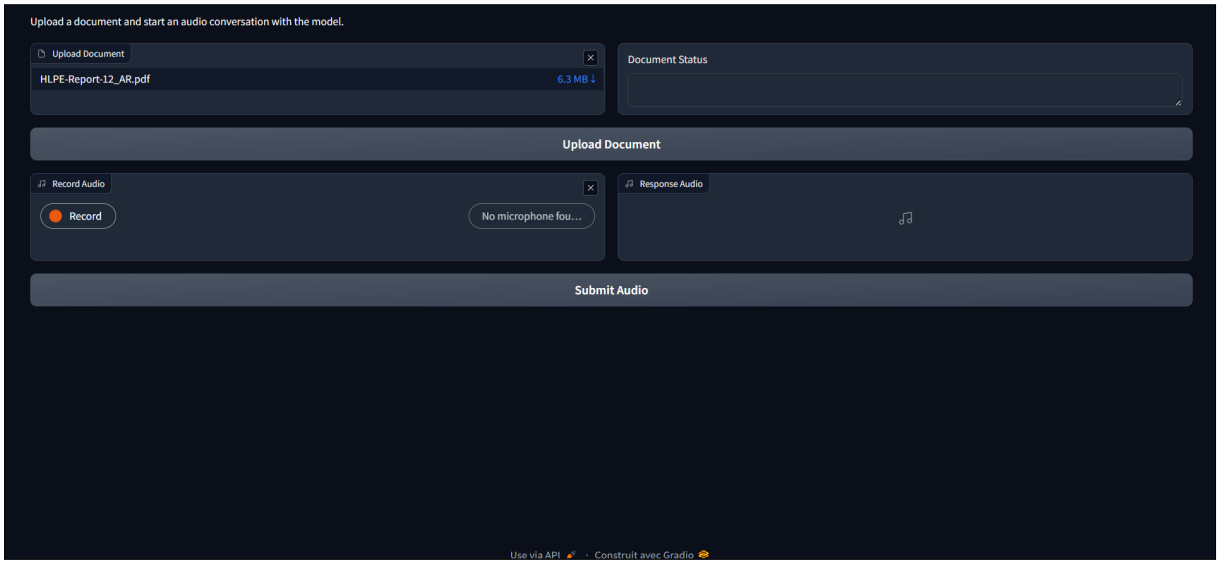

Fig. 3.6 : étapes de l'upload

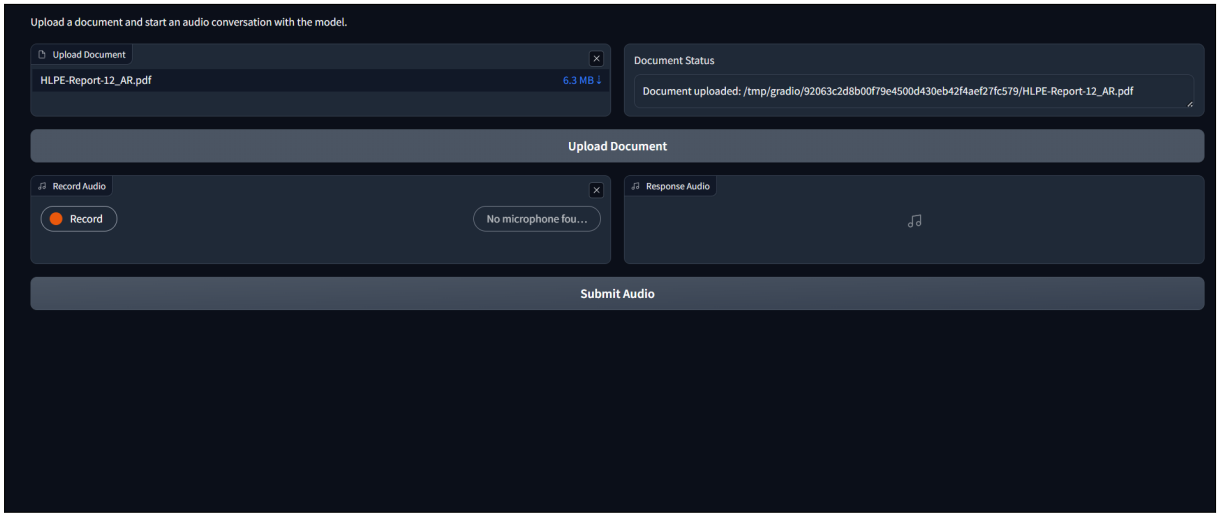

Fig. 3.7 : étapes de l'upload

3.3.1.3 Gestion du compte

Après avoir téléversé les fichiers, l'utilisateur commence sa conversation vocale. Il lance l'enregistrement en cliquant sur "Record", puis, une fois

terminé, il appuie sur ”Stop”. Enfin, il clique sur ”Submit Audio” et attend la réponse de notre modèle. Pour écouter l'audio envoyé par le modèle, il clique sur ”Play”.

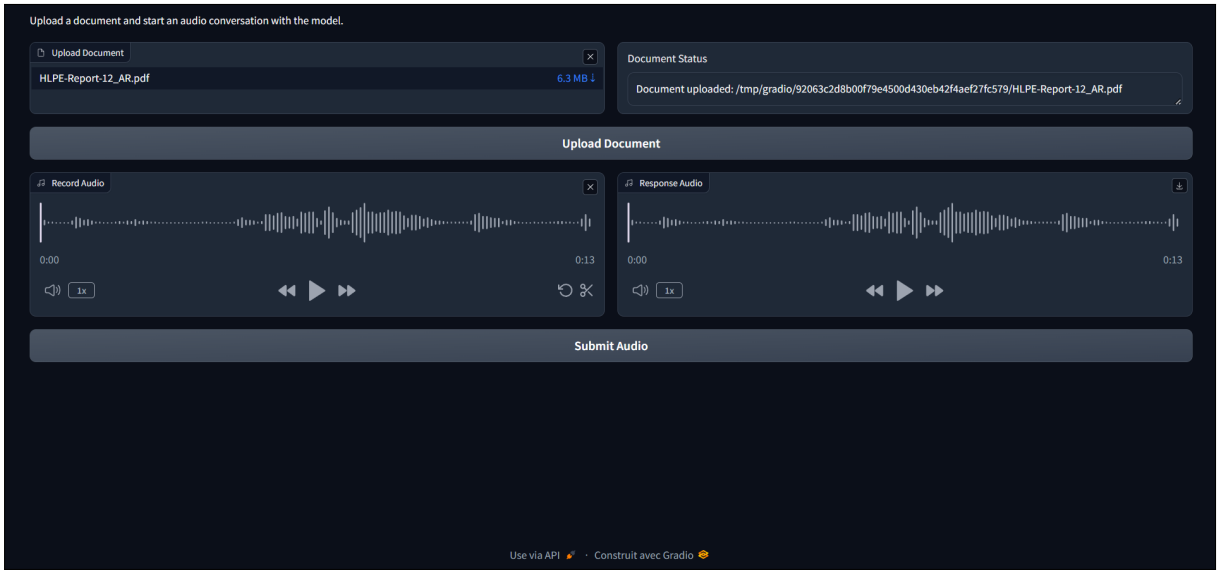

FIG. 3.8 : La conversation vocale.

3.4 Conclusion

Ce chapitre marque une avancée significative dans notre projet, où nous avons transformé notre vision en une réalité fonctionnelle et conviviale. Grâce à une analyse approfondie et à une planification rigoureuse, nous avons réussi à mettre en place les outils nécessaires et à concevoir des interfaces graphiques adaptées pour répondre aux besoins des utilisateurs. Cette réalisation nous rapproche de notre objectif final : offrir une solution innovante et efficace pour faciliter l'utilisation des documents sans perdre beaucoup de temps dans leur lecture.

Conclusion et perspectives

Au terme de notre étude, nous avons scruté avec attention les défis et les opportunités qui émaillent le champ du traitement de la langue arabe, mettant en lumière une nécessité impérieuse d'optimisation des solutions actuelles. Notre projet s'inscrit dans cette dynamique, s'élevant comme une proposition innovante et pertinente fondée sur les avancées du deep learning pour pallier les lacunes spécifiques de cette langue.

En fusionnant habilement des technologies de pointe telles que la reconnaissance vocale, le traitement du langage naturel et les systèmes de question-réponse, notre ambition se dessine : concevoir un modèle interactif et performant, offrant aux utilisateurs une porte d'accès aisée à des informations précises en arabe. Dans un paysage où l'arabe reste relativement sous-représenté dans les avancées technologiques, notre projet cherche à inverser cette tendance, en aspirant à combler un vide significatif dans le domaine du NLP.

Au-delà de l'aspect purement technique, notre engagement transcende vers une mission plus large : celle de favoriser l'accessibilité et la qualité de l'information dans cette langue, contribuant ainsi à son rayonnement et à son épanouissement dans le paysage numérique contemporain. Dans cette perspective, notre projet se dresse comme un jalon crucial dans la promotion et la préservation de la richesse linguistique et culturelle de l'arabe, ouvrant ainsi la voie à de nouvelles perspectives d'exploration et d'innovation dans un monde en constante évolution.

Une vision pour le futur inclut l'intégration de la reconnaissance optique de caractères (OCR) pour les fichiers PDF, nous permettant ainsi de travailler avec une variété de formats de documents et de garantir une expérience utilisateur encore plus enrichie et complète.

Ressources

[1] "Stack Overflow" https://stackoverflow.com/.

[2] "Langchain Documentation" https://python.langchain.com/v0.1/docs/.

[3] "Gradio Documentation" https://www.gradio.app/docs/gradio/interface.

[4] "Hugging Face Models" https://huggingface.co/models.

[5] "Google Colab" https://colab.research.google.com/.