République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université des Sciences et de la Technologie Houari Boumediene
Faculté d'Électronique et d'Informatique
Département d'Informatique
Mémoire de Master
Spécialité :
Systèmes Informatiques Intelligents

Thème:
Un système intelligent d'aide à l'audition au profit de la sécurité publique

---

Encadré par:
Col. BOUKERCA ahcène
Col. DJAMA Adel

Présenté par:
YOUSFI Zakaria
Soutenu le:
??/06/2024

Membre du Jurys:

Projet: SII-19 / 2024

Remerciements

Je remercie et je rends grâce à Dieu, qui m'a bénéficié d'une volonté suffisante pour accomplir ce modeste travail.

J'exprime mes remerciements et ma gratitude à mes promoteurs Mr BOUKERCA et Mr DJEMAA, pour m'avoir proposé ce sujet. Leur disponibilité, leur serviabilité et leurs conseils constructifs m'ont énormément aidé tout au long de mon travail, ainsi que pour l'inspiration, l'aide et le temps qu'ils ont bien voulu me consacrer.

Je tiens à remercier sincèrement les membres du jury ***Mme BABA ALI sadjia et Mme KERKAD amira*** *qui me font le grand honneur d'évaluer ce travail.*

Je remercie également ma famille pour ses contributions, son soutien et sa patience. Enfin, j'adresse mes plus sincères remerciements à tous mes proches et amis, qui m'ont toujours soutenu et encouragé au cours de la réalisation de ce mémoire.

Table des matières

Table des figures

Liste des tableaux

Liste d'abréviations

CGN	Commandement de la Gendarmerie Nationale
KB	Knowledge Base
NER	Named Entity Recognition
NLP	Natural Language Processing
$TALN$	Traitement automatique du langage naturelle
IA	Intelligence Artificielle
RE	Relation Extraction
BOW	Bag of words

Introduction Générale

Contexte

Dans les services de maintien de l'ordre, la collecte et l'analyse précises des déclarations des suspects, des témoins et des victimes sont cruciales pour mener à bien les différentes enquêtes judiciaires ou administratives. Les entretiens et les interrogatoires jouent un rôle central dans ces enquêtes, fournissant des informations essentielles qui peuvent déterminer la direction de l'enquête et influencer les décisions judiciaires. Cependant, la gestion de ces informations présente de nombreux défis, notamment la détection des contradictions dans les déclarations et la formulation de questions pertinentes pour obtenir des informations supplémentaires.

Traditionnellement, les enquêteurs prennent des notes manuelles ou enregistrent des conversations, ce qui nécessite un effort considérable pour analyser et comparer les informations recueillies. Cette approche peut être sujette à des erreurs humaines et à des pertes d'informations importantes. De plus, les enquêteurs peuvent rencontrer des difficultés lorsqu'ils sont confrontés à des déclarations ambiguës ou incohérentes, et lorsqu'ils doivent formuler des questions de suivi pour clarifier ou approfondir les réponses obtenues.

Avec l'avènement des technologies d'intelligence artificielle (IA) et du traitement automatique du langage naturel (TALN), il est désormais possible de développer des systèmes automatisés capables d'analyser les conversations de manière plus efficace et précise. Ces technologies peuvent aider à détecter les contradictions dans les déclarations, en comparant les nouvelles informations avec les données existantes dans le système. Elles peuvent également fournir des recommandations de questions basées sur le contexte de l'entretien, aidant ainsi les enquêteurs à obtenir des informations plus complètes et cohérentes.

Des systèmes similaires ont été développés dans divers domaines du sécurité publique, tels que la détection de la fraude, la reconnaissance faciale [1] et l'analyse prédictive du crime [2]. Ces systèmes montrent le potentiel de l'IA à transformer les pratiques d'enquête en automatisant des tâches complexes et en fournissant des analyses avancées. Cependant, un système spécifiquement conçu pour enregistrer les conversations d'enquête, détecter les contradictions et recommander des questions en temps réel représente une avancée significative dans ce domaine.

Contribution

Ce projet vise à combler cette lacune en fournissant un outil intégré et automatisé pour les enquêteurs, qui exploite les capacités des technologies d'IA et de TALN, le système proposé améliorera la précision et l'efficacité des enquêtes, tout en réduisant la charge cognitive des enquêteurs. Cette initiative s'inscrit dans une tendance plus large d'adoption de technologies avancées par les services de sécurité publique pour améliorer leurs capacités d'investigation.

Structure du mémoire

Ce mémoire est structuré en quatre chapitres, abordant tous les aspects liés à notre travail en terme de développement d'un outil intelligent d'aide à améliorer le processus d'audition au sein des services de maintien d'ordre. La structure des chapitres est détaillée comme suit :

Chapitre 1 : Dans ce chapitre, nous présenterons brièvement l'organisme d'accueil, en abordant son historique, ses missions, et son organisation. Nous décrirons également le Service Central Informatique du Commandement de la Gendarmerie Nationale et introduirons la problématique de notre projet ainsi que la feuille de route de notre travail.

Chapitre 2 : Ce chapitre introduise les notions nécessaires et quelques travaux liés à notre problématique. Nous introduirons le Traitement Automatique du Langage Naturel (NLP) et ses différentes tâches, ainsi nous discuterons des représentations textuelles, des modèles de réseaux de neurones pour le NLP. En plus, Nous aborderons également les travaux connexes dans le domaine liés aux systèmes de recommandation .

Chapitre 3 : Dans ce chapitre, nous décrirons en détail la conception de notre contribution. Nous présenterons les diagrammes de cas d'utilisation, le diagramme de classes. Ainsi que, nous détaillerons notre approche pour la détection des contradictions et la recommandation de questions.

Chapitre 4 : Ce chapitre aborde les aspects techniques de la mise en œuvre de notre système. Nous présenterons l'environnement de développement, les modèles utilisés, et nous décrirons en détail l'application développée.

Le mémoire se termine par un résumé des contributions, des discussions sur les limitations et des suggestions pour les orientations futures pour éventuelle amélioration.

Chapitre 1

Mise en Contexte et problématique

1.1 Introduction

Nous allons aborder dans ce chapitre le contexte et la problématique de notre projet fin d'étude. En effet, notre stage s'est déroulé au niveau du Commandement de la Gendarmerie Nationale, plus précisément au niveau du Service Central de l'Informatique de la Direction de la Télématique.

Dans la suite du chapitre, nous présenterons l'organisme d'accueil et le sujet de recherche appliqué qui nous été confié dans la cadre de notre PFE.

1.2 Présentation de l'organisme d'accueil

1.2.1 Historique de la GN

La Gendarmerie Nationale algérienne a traversé trois phases majeures d'évolution depuis sa création en 1962. Durant la première phase (1962-1973), elle a été établie comme force publique chargée du maintien de l'ordre, renforçant ainsi l'autorité de l'État. La deuxième phase (1974-1987) a vu une réorganisation visant à accompagner le développement social du pays, augmentant sa flexibilité et sa capacité d'intervention. Enfin, la troisième phase (après 1988) a connu une expansion significative de ses unités pour faire face à la criminalité organisée. Aujourd'hui, la Gendarmerie Nationale demeure un pilier crucial de la sécurité et de la stabilité nationales, œuvrant pour le bien-être et le progrès du pays [3].

1.2.2 Missions de la GN

La Gendarmerie Nationale participe à la défense nationale, notamment en matière de lutte contre le terrorisme, conformément aux plans arrêtés par le

Ministre de la Défense Nationale. Elle a pour charge l'exercice des missions de police judiciaire, de police administrative et de police militaire.

a) **Police judiciaire :**

La Gendarmerie Nationale lutte contre la criminalité et le crime organisé, raison pour laquelle, elle met en œuvre des moyens d'investigation scientifiques, techniques et d'expertise criminalistique. Elle exerce cette mission conformément aux dispositions des lois et des règlements en vigueur notamment le code des procédures pénales.

b) **Police administrative :**

La Gendarmerie Nationale veille au maintien de l'ordre et de la tranquillité publics, par une action préventive, caractérisée par une surveillance générale et continue. Elle assure la sécurité publique par la protection des personnes et des biens et la liberté de circulation sur les voies de communication. A ce titre, elle veille à l'application des lois et règlements régissant les polices générale et spéciale.

c) **Police militaire :**

La Gendarmerie Nationale assure la police judiciaire militaire conformément aux dispositions du code de justice militaire et la police générale militaire, en vertu des lois et règlements en vigueur régissant le service au sein de l'Armée Nationale Populaire.

1.2.3 Organisation de la GN

La Gendarmerie Nationale est commandée par un officier général, nommé en sa qualité de Commandant de la Gendarmerie Nationale, par un décret présidentiel. Par ailleurs, le CGN se compose des unités et structures suivantes :

— Les unités territoriales.

— Les unités Constituées.

— Les unités spécialisées.

— Les unités de soutient.

— Les organes de formation.

— Institut National de Criminalistique et de Criminologie de la Gendarmerie Nationale.

— Les services et les centres scientifique et technique.

— Service Central des Investigations Criminelles.

— Service Central de cybercriminoligie.

— Détachement Spécial d'Intervention.

Le Commandement de Gendarmerie Nationale contient :

— Etat-major de la Gendarmerie Nationale.

— Département de l'Administration Générale et des Moyens.

— Commandement des Unités des Gardes Frontières.

— nspection Générale.

— Cabinet.

— Service communication.

— Service de prévention et de contrôle .

L'Etat-major de la Gendarmerie Nationale contient :

— Direction de la Sécurité Publique et de l'Emploi.

— Direction des Ressources Humaines.

— Direction des Ecoles.

— Direction des Unités Constituées.

— Direction de la Planification et des Finances.

— Direction de la Télématique.

— Direction de Logistique et d'Infrastructures.

— Centre des Opérations.

1.2.4 Le Service Central Informatique du Commandement de la Gendarmerie Nationale

Le Service Central Informatique est l'un des services Centraux de la Gendarmerie Nationale, assume des missions particulièrement cruciales à l'échelle nationale. Il s'occupe des problèmes informatiques touchant l'ensemble des services, divisions, bureaux, commandements régionaux, groupements territoriaux, ainsi que des unit´ es et brigades.

Organisation du Service Central Informatique :

Le Service Central Informatique, illustré dans la Figure 1.1 par son organigramme, est composé de plusieurs entités œuvrant de concert pour le bon déroulement des missions informatiques au sein de la Gendarmerie Nationale.

1.3 Description et problématique de notre projet

Dans la sécurité publique, le processus d'audition est une étape cruciale pour l'officier de police judiciaire soit pour la résolution des crimes ou pour mener les différentes enquêtes judiciaires ou administratives. L'audition concerne une personne impliquée dans une affaire d'enquête, soit suspecte, plaignante, ou témoins et doit répondre à des finalités juridiques bien précises en consignant les

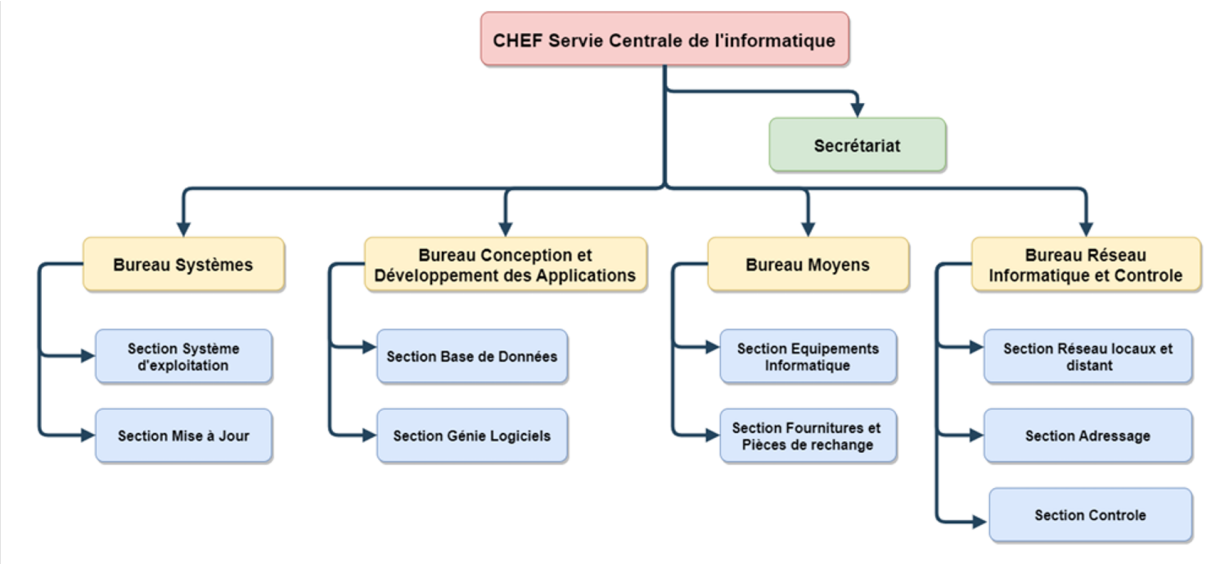

FIGURE 1.1 – Organigramme du Sevice Central Informatique

déclarations spontanées de l'intéressé et les réponses aux différentes questions. Les difficultés liées à l'opération de l'audition sont nombreuses : elles commencent par le choix des questions les plus adéquates (l'art d'interroger), la détection des contradictions dans les déclarations, la transcription précise et fidèle des déclarations pour les mettre à la disposition des magistrats et des avocats ainsi que la sécurisation des déclarations.

Dans le but d'automatiser le processus d'audition durant les enquêtes judiciaires et administratives, ce projet vise à concevoir et à développer un système d'aide à l'audition. Le système d'audition permet à tous les agents chargés de mener les enquêtes judiciaires et administratives, durant la phase d'audition d'enregistrer les déclarations, les questions et les réponses aux questions des personnes impliquées, dans une base de données centralisée et sécurisée. L'ensemble de ces informations enregistrées seront copiées fidèlement dans un procès-verbal, qui sera envoyé à l'autorité judiciaire et/ou administrative compétente. Le système d'audition permet d'assurer la continuité de service en cas de coupure de réseau/dysfonctionnement du système central, en permettant l'enregistrement des données liées à l'audition en local, et la mise à jour se fait automatiquement une fois le service sera rétabli. Le système visé devra avoir un outil intelligent de recommandation pour aider l'enquêteur à choisir les bonnes questions et pour détecter automatiquement d'éventuelles contradictions dans les déclarations d'une ou plusieurs personnes dans une ou plusieurs affaire(s) donnée(s).

1.4 Feuille de route de notre travail

pour répondre au besoin de l'organisme d'accueil, nous avons opté pour le plan suivant.

1. Elaborer un état de l'art sur les techniques traitant la même problématique. similaire à la notre
2. Proposer une approche pour répondre à notre problématique
3. faire proposer une conception adéquate et développement d'un outil prototype pour résoudre notre problème.

1.5 Conclusion

Après avoir pris connaissance de notre problématique, nous avons jugé utile de commencer par présenter l'ensemble des services de l'entreprise d'accueil. Ensuite, nous avons décrit en détail l'ensemble des paramètres à notre problématique.

Chapitre 2

État de l'art

2.1 Traitement automatique du langage naturel

Le traitement automatique du langage naturel (TALN), ou Natural Language Processing (NLP) en anglais, est un domaine essentiel de l'intelligence artificielle qui vise à permettre aux machines de comprendre et d'interpréter le langage humain [4]. Cette discipline repose sur la combinaison de la linguistique informatique, des techniques statistiques, et des algorithmes d'apprentissage automatique et d'apprentissage profond [5].

Les technologies NLP permettent aux ordinateurs de traiter le texte et les données vocales de manière à extraire des informations significatives, comme le contexte, l'intention et les émotions [6]. Grâce à ces capacités, le NLP trouve des applications variées dans de nombreux domaines.

Par exemple, les systèmes de traduction automatique, les assistants virtuels comme **SIRI** d'appel [1] et **ALEXA** d'AMAZON [2], ainsi que les chatbots (un agent logiciel qui dialogue avec un utilisateur), qui utilisent des techniques de NLP pour interagir avec les utilisateurs de manière fluide et naturelle.

Le NLP joue un rôle crucial dans l'avancement de diverses technologies contemporaines. Voici quelques exemples illustratifs :

— Modèles de langage comme GPT [7]

— Reconnaissances vocales [8]

— Traduction automatique [9], analyse des sentiments [4]

2.2 Représentation du texte

Le domaine du NLP se divise principalement en deux étapes : la première consiste à représenter le texte d'entrée (données brutes) sous forma numérique

1. `https://www.apple.com/fr/siri/`
2. `https://www.alexa.com/`

(vecteurs ou matrices), et la deuxième concerne la conception de modèles pour traiter ces données numériques afin d'atteindre un objectif ou une tâche spécifique. Cette section et la section 2.3 se concentre sur la première étape et montre comment, grâce à l'évolution des méthodes de représentation textuelle, le domaine du NLP est passé de la compréhension de fragments isolés à la prise en compte de tous les aspects du texte.

Pour que les machines puissent comprendre et analyser les modèles linguistiques, il est essentiel de convertir les mots en nombres. Ce processus, appelé représentation textuelle, est fondamental pour la plupart des tâches de NLP. La manière dont le texte est représenté influence grandement les performances des modèles d'apprentissage automatique. On distingue deux grandes catégories de représentations textuelles [10] :

— Représentation textuelle discrète

— Représentation textuelle continue

2.2.1 Représentation textuelle discrète

Dans ce type de représentation, les mots sont initialement représentés par des indices correspondant à leur position dans un dictionnaire dérivé d'un corpus. Le processus de création de cette représentation se déroule en plusieurs étapes.

Tout d'abord, un corpus est une collection de textes utilisés comme base pour analyser et construire des représentations. Il peut s'agir de documents, d'articles, de livres, ou tout autre ensemble de textes pertinents pour le domaine d'application envisagé.

À partir du corpus, un dictionnaire de mots est construit. Le processus commence par la collecte de toutes les données textuelles qui feront partie du corpus. Ensuite, le texte est prétraité par des tâches comme la conversion du texte en minuscules, la suppression de la ponctuation, et l'élimination des mots vides (stop words). Une fois le texte prétraité, il est divisé en unités significatives appelées tokens. Optionnellement, les tokens peuvent être filtrés pour enlever ceux qui sont trop fréquents ou trop rares dans le corpus. Enfin, un dictionnaire est créé où chaque mot unique du corpus est associé à un indice unique. Ce dictionnaire est souvent représenté sous forme de table de correspondance ou de liste triée [6].

Prenons un exemple simple. Considérons le corpus constitué de deux documents suivants :

d1 = "Il fait beau aujourd'hui. Le soleil brille."

d2 = "Aujourd'hui est une belle journée."

Et la phrase a représenter : "Il fait beau aujourd'hui"

Le processus est résumé dans la figure 2.1.

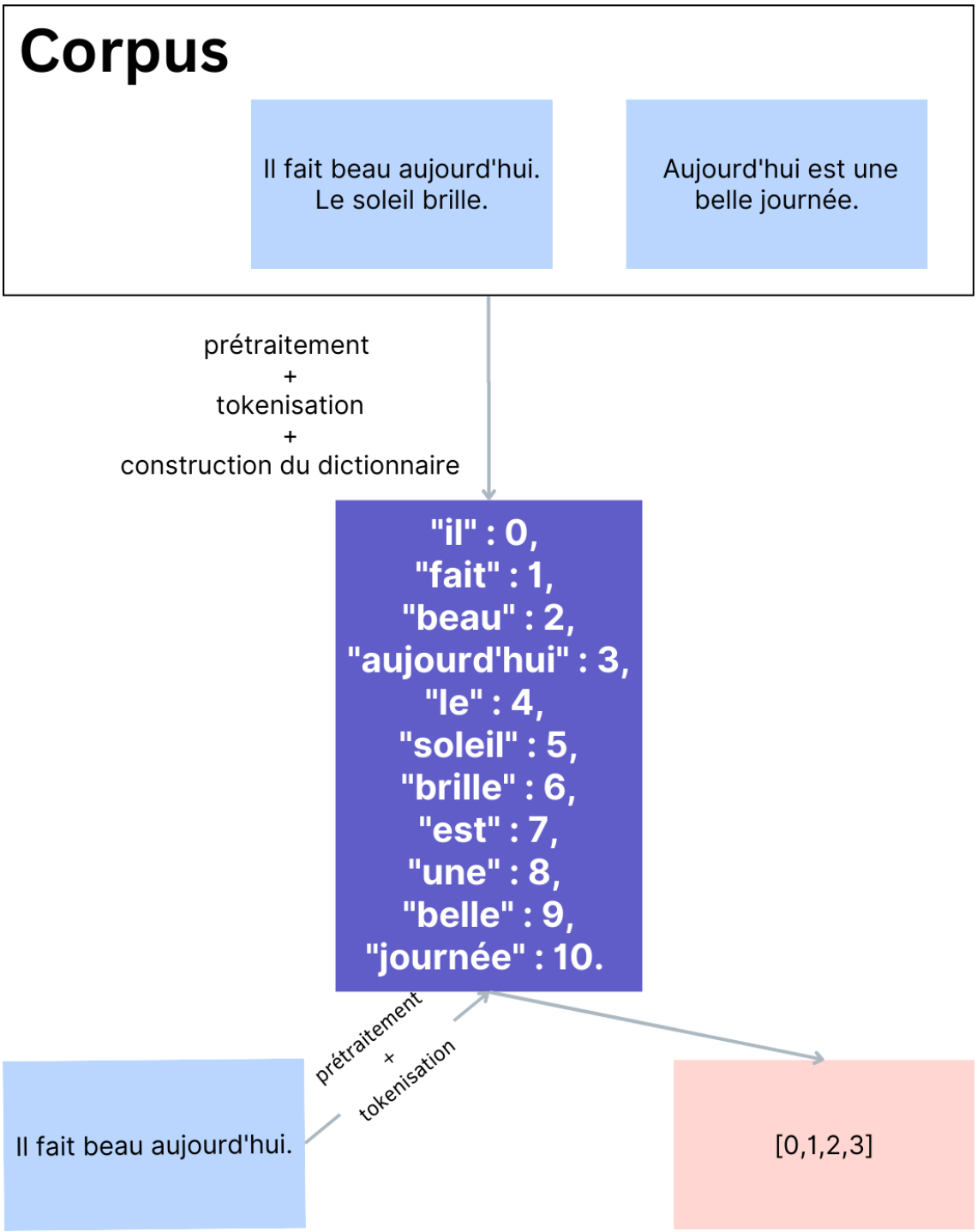

FIGURE 2.1 – Représentation de texte

Les méthodes de représentation textuelle discrètes couramment utilisées sont :

— encodage One-Hot

— Sac de mots

— TF-IDF

— Matrice de co-occurrence

2.2.1.1 encodage One-Hot

Cette méthode [11] représente chaque mot du vocabulaire par un vecteur binaire de la taille du vocabulaire, avec un seul élément à 1 et tous les autres à 0.

Par exemple, la phrase "il fait beau aujourd'hui" est illustrée par la matrice présentée dans le tableau 2.1.

	il	fait	beau	aujourd'hui	le	soleil	brille	est	une	belle	journée
il	1	0	0	0	0	0	0	0	0	0	0
fait	0	1	0	0	0	0	0	0	0	0	0
beau	0	0	1	0	0	0	0	0	0	0	0
aujourd'hui	0	0	0	1	0	0	0	0	0	0	0

Tableau 2.1 – exemple de l'encodage one-hot

Pour les vocabulaires de grande taille cette méthode produit des vecteurs de grande dimension, ce qui peut entraîner des problèmes de stockage et de calcul. De plus, le One-Hot ne capture pas les relations sémantiques entre les mots. Par exemple, les mots "chien" et "loup", qui sont sémantiquement proches, auront des vecteurs orthogonaux et sans aucune similarité . Il faut noter aussi que parmi les représentations textuelles discrète, cette méthode est la seule qui conserve l'ordre des mots.

2.2.1.2 Sac de mots

La représentation par sac de mots (BOW pour *bag of words* en anglais), comme son nom l'indique intuitivement, place les mots dans un « sac » et calcule la fréquence d'apparition de chaque mot. Il existe d'autre variantes de cette représentation qui ne considère pas la fréquence mais la présence des mots seulement.

Par exemple, la phrase "il fait beau aujourd'hui" est présentée par la matrice dans le tableau 2.2.

	il	fait	beau	aujourd'hui	le	soleil	brille	est	une	belle	journée
il fait beau aujourd'hui	1	1	1	1	0	0	0	0	0	0	0

Tableau 2.2 – example de BOW

L'intuition derrière la représentation par sac de mots est que des documents ayant des mots similaires sont similaires, indépendamment de la position des mots.

Cette méthode est simple à implémenter et efficace pour des tâches comme l'analyse de sentiments [12], mais les relations sémantiques ne sont pas capturées. Le nombre des lignes de la matrice de représentation est significativement plus petite que celle de l'encodage one-hot, mais elle reste grande en dimension pour les vocabulaires de grande taille.

2.2.1.3 TF-IDF

Le TF-IDF (Term Frequency-Inverse Document Frequency) est utilisé notamment dans la recherche d'information [13]. Ce approche pondère chaque mot en fonction de sa fréquence dans le document et de sa rareté dans l'ensemble des documents (corpus). Il combine deux mesures pour évaluer l'importance d'un mot dans un document :

$$\text{TFIDF}(m, d) = \text{TF}(m, d) \times \text{IDF}(m) \tag{2.1}$$

où :

— **TF (Term Frequency)** : la fréquence d'apparition d'un mot m dans un document d.

— **IDF (Inverse Document Frequency)** : Cette valeur s'appelle la valeur de discrimination [14], elle mesure de la rareté d'un mot dans l'ensemble des documents, calculée comme suit :

$$\text{IDF}(m) = \log\left(\frac{N}{n_i}\right) \tag{2.2}$$

— **N** : le nombre total de documents

— n_i : le nombre de documents contenant le mot m

Pour notre exemple, la matrice tf-idf construite est présentée dans le tableau 2.3.

Term	**TF-IDF(d1)**	**TF-IDF(d2)**
il	0.301	0.0
fait	0.301	0.0
beau	0.301	0.0
aujourd'hui	0.0	0.0
le	0.301	0.0
soleil	0.301	0.0
brille	0.301	0.0
est	0.0	0.301
une	0.0	0.301
belle	0.0	0.301
journée	0.0	0.301

TABLEAU 2.3 – Représentation TF-IDF des documents d1 et d2

Cette représentation Réduit l'impact des mots courants et met en valeur les mots informatifs (rares) ce qui permet de distinguer entre les documents. Les vecteurs restent de grande dimension pour un grand vocabulaire et petite documents comme des phrases. La méthode ne capture pas les relations contextuelles entre les mots.

2.2.1.4 Matrice de co-occurrence

Cette représentation [15] examine la proximité des entités les unes par rapport aux autres au sein d'un texte. Une entité peut être un mot unique, un bi-gramme (séquence de deux mots), ou même une phrase, bien que l'utilisation d'un seul mot soit la méthode la plus courante pour calculer la matrice. En analysant les co-occurrences dans une fenêtre contextuelle (c'est-à-dire un nombre défini de mots autour d'un mot cible), cette matrice permet de révéler les associations et les relations entre différents mots dans un corpus. Cela aide à comprendre comment les mots sont utilisés ensemble et à identifier les tendances et motifs linguistiques présents dans le texte.

Tableau 2.4 montre la matrice de co-occurrence construite pour l'exemple en considération.

	il	fait	beau	aujourd'hui	le	soleil	brille	est	une	belle	journée
il	0	1	0	0	0	0	0	0	0	0	0
fait	1	0	1	0	0	0	0	0	0	0	0
beau	0	1	0	1	0	0	0	0	0	0	0
aujourd'hui	0	0	1	0	0	0	0	1	0	0	0
le	0	0	0	0	0	1	0	0	0	0	0
soleil	0	0	0	0	1	0	1	0	0	0	0
brille	0	0	0	0	0	1	0	0	0	0	0
est	0	0	0	1	0	0	0	0	1	0	0
une	0	0	0	0	0	0	0	1	0	1	0
belle	0	0	0	0	0	0	0	0	1	0	1
journée	0	0	0	0	0	0	0	0	0	1	0

Tableau 2.4 – Matrice de co-occurrence pour le corpus donné

Dans cette représentation, chaque vecteur de mot est formé en fonction des mots qui l'entourent, de sorte que le sens du mot est déduit de son contexte. Cependant, un problème majeur avec ce approche est que les mots fréquemment utilisés influencent de manière disproportionnée la mesure de similarité. Les mots fréquents dans le contexte peuvent ne pas fournir beaucoup d'informations sémantiques sur les mots cibles mais affectent tout de même considérablement leur score de similarité. Par exemple, "épicerie" et "sports" ne sont pas étroitement liés, mais s'ils partagent des mots de contexte similaires, leurs vecteurs seront projetés plus près les uns des autres, ce qui entraînera un score de similarité élevé.

Il existe d'autres méthodes de représentation discrète qui ne sont pas mentionnées ici. Pour une discussion plus détaillée et une couverture complète de ces techniques, le lecteur est invité à consulter la référence suivante : [10].

2.2.2 Représentation textuelle continue

La représentation continue, qui est basée sur les réseaux de neurones, permet de capturer des structures complexes dans les données. Les représentations de ce type sont souvent appelées des embeddings, qui sont des vecteurs denses contenant des nombres réels. Cela diffère des représentations discrètes où il n'y a que des nombres naturels et où les vecteurs sont clairsemés. Dans ce cadre, la signification d'un mot dépend du contexte fourni par les autres mots, et n'est donc pas indépendante. Les configurations des mots reflètent diverses métriques et concepts présents dans les données. Ainsi, les informations relatives à un mot sont réparties le long du vecteur qui le représente. À l'opposé, dans la représentation discrète, chaque mot est considéré comme unique et indépendant des autres. Ces nouveaux vecteurs de mots sont sensibles au contexte, peuvent identifier des synonymes et des antonymes, et construire des analogies et des catégories de mots, ce qui était impossible avec les approches vu précédemment. Les vecteurs de mots capturent le sens des mots (littéral et implicite) et les représentent en utilisant des valeurs flottantes denses. Ils représentent à la fois les aspects sémantiques et syntaxiques du mot. Leur longueur se situe généralement entre 100 et 500 dimensions. Il existe deux types de représentation continue : les word embeddings statiques (Word2Vec, GloVe, FastText) et dynamiques (générés par des modèles de langage comme BERT) [10].

Les méthodes courantes de représentation continue incluent :

— Word2Vec

— GloVe

— Transformateurs (voir section 2.3.3)

2.2.2.1 Word2Vec

Word2Vec est un algorithme de word embedding statique, qui représente les mots ou phrases d'un texte sous forme de vecteurs de nombres réels dans un modèle vectoriel. Développé par une équipe de recherche de Google sous la direction de Tomas Mikolov [16], cet algorithme est devenu un outil essentiel dans le traitement du langage naturel [17].

Word2Vec propose deux architectures neuronales principales : CBOW (Continuous Bag of Words) et Skip-Gram.

— **CBOW (Continuous Bag of Words)** : Cette méthode tente de prédire un mot en se basant sur son contexte, c'est-à-dire les termes qui l'entourent dans une phrase. Elle est particulièrement efficace avec des ensembles de données plus petits et offre un temps d'entraînement rapide comparé à Skip-Gram.

— **Skip-Gram** : Cette méthode, à l'inverse de CBOW, prédit le contexte à partir du mot cible. Elle tend à mieux fonctionner avec des ensembles de données plus larges, bien qu'elle nécessite un temps d'entraînement plus long.

2.2.2.2 GloVe

GloVe [18] (Global Vectors for Word Representation) est un algorithme d'embedding statique qui combine les avantages des méthodes basées sur le comptage (comme les matrices de cooccurrence) et des méthodes prédictives (comme Word2Vec), le qualifiant ainsi de méthode hybride. L'objectif principal de GloVe est de trouver des vecteurs de mots qui capturent les relations statistiques globales des cooccurrences de mots. L'algorithme s'appuie sur la fonction d'objectif suivante :

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T w_j + b_i + b_j - \log(X_{ij}))^2$$

où :

— X_{ij} représente le nombre de cooccurrences entre le mot i et le mot j

— w_i w_j sont les vecteurs de mots,

— b_i et b_j sont les biais associés à chaque mot,

— f est une fonction de pondération.

Ainsi, GloVe construit des vecteurs de mots w_i et w_j qui respectent les cooccurrences observées X_{ij}, une statistique calculée globalement à partir de la matrice de cooccurrence.

2.3 Modèles de réseaux de neurones pour le NLP

2.3.1 Réseaux de Neurones Récurrents

2.3.1.1 Introduction

Les réseaux de neurones récurrents (RNN pour *recurrent neural network* en anglais) sont une architecture de réseau de neurones spécialement conçue pour traiter des données séquentielles telles que le texte, l'audio, et plus encore. Contrairement aux réseaux de neurones classiques, qui traitent chaque donnée indépendamment, les RNN conservent une "mémoire" des états précédents. À chaque nouvelle entrée, les RNN concatènent cette entrée avec l'état précédent, ce qui permet au réseau d'apprendre et de comprendre le contexte global, comme le contexte d'une phrase ou d'un extrait audio, et ainsi de prédire le mot suivant [19].

En pratique, un réseau de neurones récurrent se présente comme dans la figure suivante [20] :

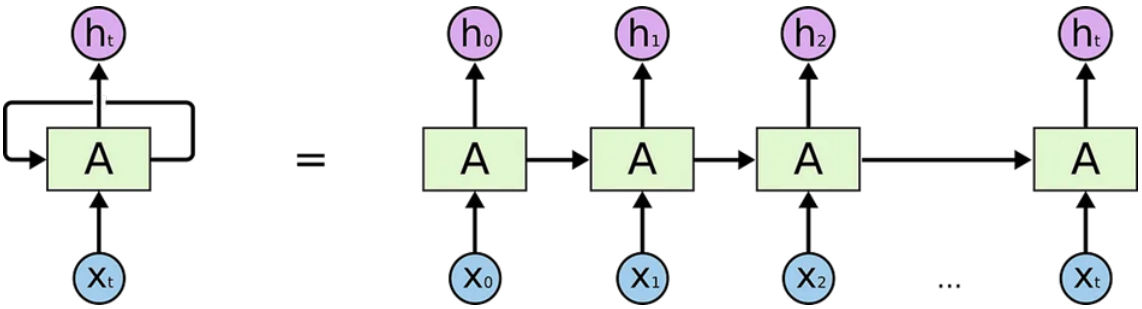

FIGURE 2.2 – structure d'un réseau de neurones récurrent standard

2.3.1.2 Limites des RNN Classiques

Problème de Mémoire à Long Terme

Un des principaux défis des RNN est la gestion de la mémoire à long terme. Prenons l'exemple d'une phrase complexe :

"Le football est un sport très populaire, il est joué dans presque tous les pays du monde. Cristiano Ronaldo et Lionel Messi, deux des plus grands joueurs de tous les temps, sont de véritables légendes dans ce domaine. Le basketball, un autre sport apprécié par des millions de personnes à travers le monde, est basé sur des règles similaires à ...."

Pour prédire "le football" à la fin, le modèle doit se souvenir que "Le football est un sport" mentionné au début, une tâche difficile pour un RNN classique.

Problème de Parallélisation

Les RNN traitant les données de manière séquentielle, leur entraînement ne peut pas tirer pleinement parti des optimisations matérielles et logicielles pour la parallélisation, ce qui ralentit considérablement leur processus d'apprentissage.

Pour surmonter ces limitations, des unités récurrentes plus complexes, telles que les cellules "gated" comme les LSTM (Long Short-Term Memory) et les GRU (Gated Recurrent Units), ont été développées. Ces cellules améliorent la gestion des flux de données et filtrent les informations pour ne conserver que les plus pertinentes.

2.3.2 LSTM : Long-Short-Term-Memory

2.3.2.1 Introduction

Les LSTM (Long Short-Term Memory ou mémoire à long terme en français) ont été créés pour résoudre le problème de la mémoire à long terme rencontré par les RNN classiques [19]. Ces réseaux utilisent des cellules mémoire plus complexes, appelées "gated cells", qui contrôlent le flux de données. Une cellule LSTM se compose de trois états d'entrée/sortie et de trois portes internes, comme illustré dans la figure 2.3 [21].

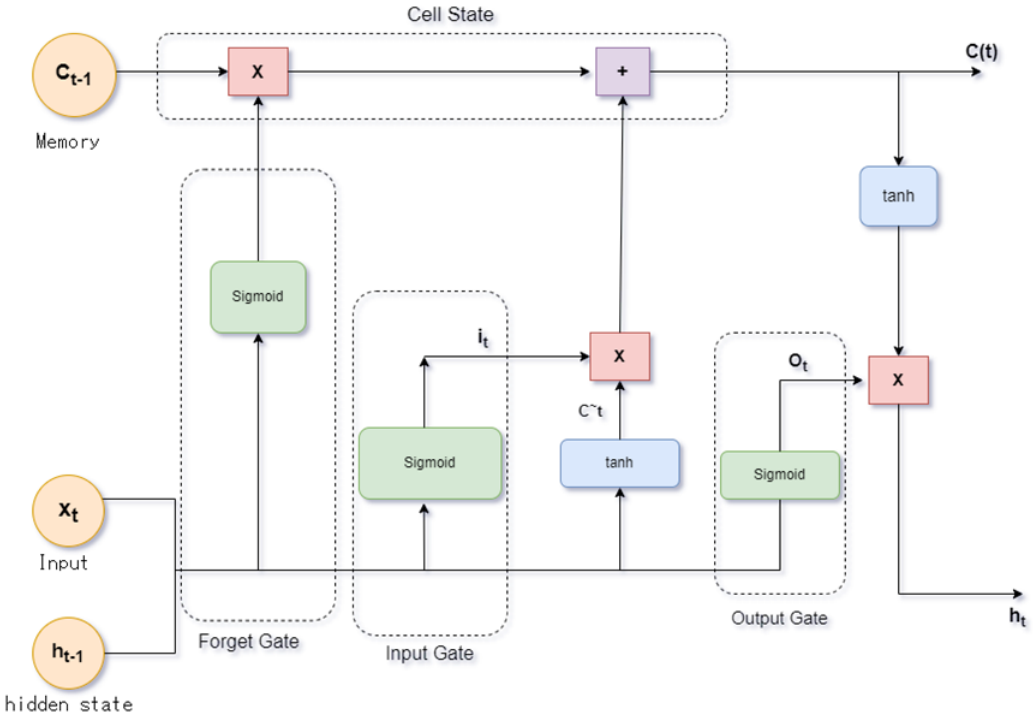

FIGURE 2.3 – Structure d'une cellule de LSTM

2.3.2.2 Structure des Cellules LSTM

Les cellules LSTM se distinguent par leur capacité à retenir et à oublier des informations de manière sélective à travers trois types de portes : la porte d'oubli (forget gate), la porte d'entrée (input gate) et la porte de sortie (Output gate). Chaque porte joue un rôle crucial dans la gestion des données à chaque étape du traitement séquentiel.

États de la Cellule

— **Cell state :**

Il s'agit de l'état de la cellule qui agrège les données de tous les états précédents. C'est ici que se trouve la "mémoire" à long terme du LSTM.

— **Hidden state :**

Cet état encode la caractérisation de l'entrée précédente. Il est utilisé pour la sortie actuelle et également passé à la prochaine étape temporelle.

— **Current input :**

C'est l'entrée actuelle, par exemple, un mot dans une phrase pour le traitement de texte.

Portes des Cellules

— **Porte d'Oubli (Forget Gate) :**

La porte d'oubli décide quelles informations de l'état de la cellule doivent être

oubliées. Cela se fait via une fonction sigmoïde suivie d'une multiplication pointwise.

— **Porte d'Entrée (Input Gate) :**

La porte d'entrée sélectionne les nouvelles informations à ajouter à l'état de la cellule. Elle utilise une combinaison de fonction sigmoïde et tanh pour ajouter les informations pertinentes à l'état de la cellule.

— **Porte de Sortie (Output Gate) :**

La porte de sortie décide quelles informations de l'état de la cellule vont influencer la sortie à la prochaine étape. Elle combine les informations des portes précédentes pour former la sortie "hidden", utilisée soit comme prochaine entrée, soit pour effectuer une prédiction.

Les LSTM permettent de retenir des informations pertinentes sur de longues séquences de données grâce à leurs portes intelligentes. Elles filtrent les informations et maintiennent celles qui sont cruciales pour des prédictions précises. La combinaison des états de la cellule et des portes permet aux LSTM de surmonter les limitations des RNN classiques, particulièrement en ce qui concerne la mémoire à long terme et la parallélisation des données.

2.3.2.3 Limites des LSTM

Complexité et ressources computationnelles

Les LSTM sont plus complexes que les RNN traditionnels en raison de leur architecture avec des portes d'entrée, de sortie et d'oubli. Cette complexité accrue entraîne une augmentation des ressources computationnelles nécessaires pour l'entraînement et l'inférence. De plus, la présence de nombreuses hyperparamètres à régler peut compliquer le processus de formation et nécessiter des efforts supplémentaires pour trouver les configurations optimales.

Problèmes de parallélisation

Les LSTM, comme tous les réseaux de neurones récurrents (RNN), traitent les données de manière séquentielle. Cette caractéristique séquentielle limite leur capacité à tirer parti des optimisations matérielles et logicielles pour la parallélisation, ce qui ralentit considérablement le processus d'apprentissage et d'inférence. Cette limitation rend les LSTM moins efficaces pour les applications nécessitant une grande vitesse de traitement.

2.3.3 Transformateurs

2.3.3.1 Introduction

Le transformateur (transformer) en anglais, une innovation majeure dans le domaine de l'apprentissage profond, a été dévoilé en 2017 dans l'essai "Attention is

all you need" [22]. Conçu par une équipe de chercheurs de Google Brain et Google Research, ce modèle révolutionnaire a été conçu pour pallier les lacunes observées dans les architectures précédentes, telles que les RNN et les LSTM.

Au cœur du Transformer réside son mécanisme d'attention, qui permet au modèle de prendre en compte le contexte global de l'entrée. Contrairement aux RNN, qui traitent les données de manière séquentielle et donc lente, le Transformer peut exploiter la parallélisation pour accélérer considérablement le processus d'entraînement. Ci-dessous l'architecture du transformer :

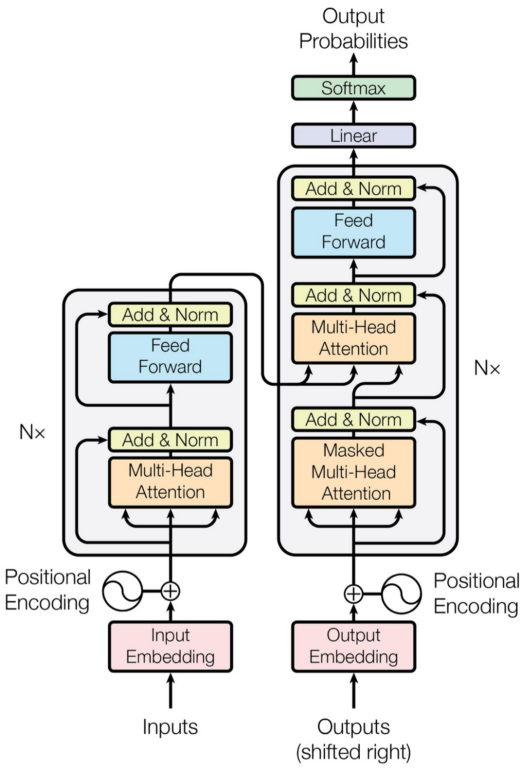

FIGURE 2.4 – L'architecture du transformer

Nous allons maintenant expliquer les mécanismes clé qui forme le transformer :

2.3.3.2 Mécanisme d'attention

Le concept fondamental du mécanisme d'attention, connu sous le nom de self-attention, est au cœur de l'architecture Transformer. Ce mécanisme analyse les relations entre les différents mots d'une séquence afin de produire une représentation contextuelle pertinente, enrichissant ainsi les informations présentes dans notre entrée.

Le processus de self-attention repose sur trois vecteurs clés :

— Vecteur de requête (q).

— Vecteur de clé (k).

— Vecteur de valeur (v).

Chacun de ces vecteurs est indexé en fonction de la position du mot dans la séquence (par exemple, q1, k1 et v1 pour le premier mot). Ces vecteurs sont calculés en multipliant l'embedding de la séquence d'entrée par trois matrices, qui sont ajustées pendant le processus d'entraînement du Transformer.

2.3.3.3 Encodeur et Décodeur

En ce qui concerne l'architecture spécifique du Transformer, l'encodeur et le décodeur jouent des rôles essentiels. L'encodeur est constitué d'un empilement de N = 6 couches identiques, chaque couche comprenant deux sous-couches : un mécanisme de multi-têtes self-attention et un réseau de feed-forward. L'entrée de chaque encodeur est la sortie du précédent, avec le premier encodeur recevant un vecteur d'embedding et la sortie du dernier encodeur étant utilisée dans le décodeur.

Le décodeur, similaire à l'encodeur, est également composé de 6 couches identiques, intégrant une sous-couche de self-attention et un réseau feed-forward. De plus, le décodeur comprend une couche d'attention encodeur-décodeur, qui permet au décodeur de se concentrer sur les parties pertinentes de la séquence d'entrée pendant le processus de décodage. Finalement, le dernier décodeur est connecté à un bloc de réseau neuronal linéaire + Soft-max, qui identifie les correspondances dans le vocabulaire pour les sorties du dernier encodeur.

2.3.4 BERT

2.3.4.1 Introduction

BERT (Bidirectional Encoder Representations from Transformers) se présente comme un modèle transformateur révolutionnaire conçu par une équipe de Google en 2018 [23]. Il réinvente la pré-entraînement de représentations profondes bidirectionnelles à partir de texte non étiqueté en conditionnant conjointement les contextes gauche et droit, dans le but d'acquérir une compréhension approfondie du contexte linguistique.

Grâce au réglage fin (fine-tuning), réalisé en ajoutant une seule couche de sortie supplémentaire, BERT peut produire des résultats de pointe. Cette avancée est rendue possible grâce à la technique de Masked LM, permettant un entraînement bidirectionnel dans des modèles jusque-là inatteignables. Ci-dessous, nous décrivons les procédures générales de pré-entraînement et de réglage fin pour BERT :

2.3.4.2 Aperçu Architectural de BERT

BERT se compose d'un encodeur de transformateur multicouche bidirectionnel, avec deux variantes : le modèle de base et le modèle large. La distinction réside dans le nombre de couches, la taille cachée et le nombre de têtes d'auto-attention. La

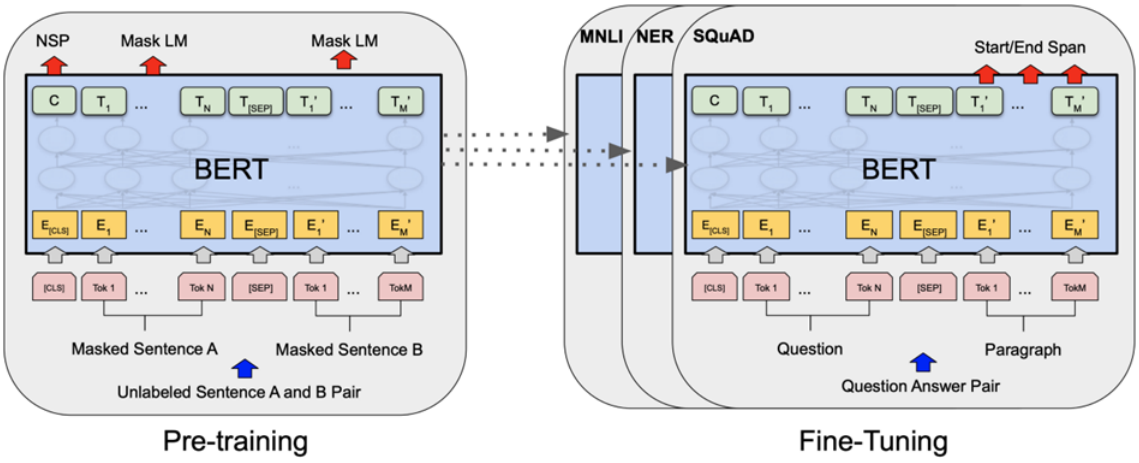

FIGURE 2.5 – L'Architecture de BERT pour le Pré-entraînement et le Réglage Fin

représentation des données d'entrée peut englober à la fois des phrases simples et des paires de phrases au sein d'une seule séquence de tokens, facilitant ainsi la mise en œuvre de tâches de TALN sur des données conversationnelles.

2.3.4.3 Pré-entraînement

Deux concepts clés sous-tendent l'ascension de BERT en tant que modèle standard pour l'apprentissage par transfert :

• Masked LM : Pour chaque séquence, un pourcentage de mots est remplacé par le token [MASK], incitant le modèle à les prédire en fonction du contexte fourni par les autres mots non masqués de la séquence.

• Prédiction de la Phrase Suivante (NSP) : Au cours de l'entraînement de BERT, le modèle reçoit des paires de phrases en entrée et apprend à prédire si la deuxième phrase de la paire est la phrase suivante dans le document d'origine. Tout au long de l'entraînement, 50% des entrées sont une paire dans laquelle la deuxième phrase est la phrase suivante dans le document d'origine, tandis que dans les 50% restants, une phrase aléatoire du corpus est choisie comme deuxième phrase. Ce concept est essentiel pour comprendre la relation entre deux phrases, cruciale pour des tâches de TALN telles que la réponse aux questions et l'inférence en langage naturel.

2.3.4.4 Réglage Fin de BERT

Le réglage fin permet à BERT de modéliser de nombreuses tâches de TALN, à un coût relativement faible, en ajoutant simplement une couche au modèle de base :

• Les tâches de classification, telles que l'analyse des sentiments, consistent à ajouter une couche de classification au-dessus de la sortie du transformateur pour le token [CLS].

• Pour la Reconnaissance d'Entités Nommées (NER), où le modèle reçoit une séquence de texte et doit annoter les différents types d'entités qu'elle contient, un

modèle NER peut être entraîné en alimentant le vecteur de sortie de chaque token dans une couche de classification qui prédit l'étiquette.

2.3.5 Transformateurs de phrases

2.3.5.1 Introduction

Les transformers de phrases (*sentence transformers* en anglais) sont un type de modèle transformer conçu pour générer des embeddings de phrases de haute qualité, qui sont des vecteurs de taille fixe capturant la signification sémantique des phrases. Contrairement aux modèles transformers traditionnels comme BERT, qui génèrent des embeddings pour des tokens individuels, les transformers de phrases produisent des embeddings pour des phrases entières. Cela les rend particulièrement utiles pour des tâches impliquant la comparaison ou la compréhension de phrases complètes, telles que la similarité textuelle sémantique et le regroupement de phrases.

2.3.5.2 Développement et Évolution des Transformers de Phrases

Le développement des transformers de phrases a été marqué par plusieurs innovations clés. L'un des travaux fondamentaux est le Sentence-BERT (SBERT) [24], qui adapte l'architecture BERT pour des tâches au niveau des phrases. SBERT a introduit une structure de réseau siamois qui ajuste BERT sur une combinaison de tâches de classification et de régression pour produire des embeddings de phrases significatifs. Les modèles ultérieurs ont construit sur cette approche, améliorant l'efficacité et la qualité des embeddings [25].

2.3.5.3 Modèle All-MiniLM-L6-v2

Le modèle all-MiniLM-L6-v2 [26] est une version compacte et efficace du MiniLM (Miniature Language Model) [25], conçu par Hugging Face. Il utilise une architecture plus petite (6 couches) par rapport aux 12 ou 24 couches de BERT, le rendant ainsi beaucoup plus rapide tout en maintenant des performances compétitives. Ce modèle a été pré-entraîné sur un corpus large et diversifié et affiné en utilisant une variété de tâches au niveau des phrases pour garantir des performances robustes dans différents domaines.

2.4 Les différentes tâches de NLP

Les subtilités du langage humain rendent extrêmement complexe la tâche de développer des logiciels capables de comprendre avec précision le sens voulu des textes ou d'autres données. Des phénomènes tels que les homonymes, les homophones, le sarcasme, les métaphores et les variations de la structure des phrases ne sont que quelques exemples des défis que présente le langage humain.

Ces subtilités, qui peuvent prendre des années à être maîtrisées par les humains, doivent être apprises par les programmeurs dès le début pour que les applications basées sur le langage naturel puissent reconnaître et comprendre correctement ces nuances, et ainsi être véritablement utiles.

Plusieurs tâches sont définies pour aider les ordinateurs à interpréter les données textuelles. Voici quelques exemples de ces tâches qui sont pertinentes pour la réalisation de notre projet :

2.4.1 Reconnaissance d'entités nommées

2.4.1.1 Définition

La reconnaissance d'entités nommées (Named Entity Recognition ou NER en anglais) est une sous-tâche de l'extraction d'informations visant à identifier et classer les mots clés, appelés entités, présents dans un document. Cette technologie permet de regrouper ces entités en catégories prédéfinies. Par exemple, dans un texte, la NER peut détecter et distinguer des mentions de personnes et de lieux, qui appartiennent à des catégories distinctes.

2.4.1.2 Les approches du NER

Comme évoqué précédemment, la reconnaissance d'entités nommées a pour but de repérer et de classer des mots dans un document textuel. Pour réaliser cette tâche, trois méthodes principales sont utilisées : l'approche statistique, l'approche basée sur des règles et l'approche hybride, qui combine les deux premières [27].

— **Approche basé sur les règles**

Cette méthode repose sur la définition d'un ensemble de règles grammaticales, syntaxiques, etc., établies par des linguistes, ainsi que sur l'utilisation de dictionnaires. Elle implique l'analyse du texte fourni en entrée pour identifier les entités et leurs catégories en appliquant les règles prédéfinies. Bien que cette approche offre une grande précision, elle exige un investissement considérable en termes de travail humain.

— **Approche basée statistique**

Cette méthode, contrairement à la précédente, s'appuie sur des règles statistiques et logiques pour atteindre le même objectif, en utilisant diverses techniques d'apprentissage automatique.

— **Apprentissage supervisé**

Cette méthode nécessite un corpus annoté pour entraîner le modèle. Le principal inconvénient est qu'elle ne se généralise pas bien, en raison du manque de grands ensembles de données pour ce type de tâche, ce qui nous pousse à envisager d'autres approches.

— **Apprentissage semi-supervisé**

Cette approche commence avec un corpus où seules quelques données sont étiquetées. Le modèle est d'abord entraîné sur ces données, puis le processus est itéré pour détecter d'autres entités similaires aux premières. Ce processus est répété en utilisant les résultats précédents pour affiner le modèle.

— **Apprentissage non supervisé**

Cette technique utilise le clustering, qui consiste à regrouper des éléments apparaissant dans des contextes similaires. Pour un texte donné, le modèle tente de trouver le groupe le plus similaire.

Malgré l'efficacité des méthodes d'apprentissage automatique et profond, le manque de grands ensembles de données les empêche d'atteindre la précision des approches basées sur des règles dans des domaines spécifiques. Cependant, ces méthodes offrent de meilleures performances en termes de généralisation.

— **Approche Hybride**

Dans cette approche, l'objectif est de fusionner les techniques basées sur des règles avec celles basées sur des statistiques pour tirer le meilleur parti des deux méthodes. L'idée est de parvenir à un compromis pour obtenir des résultats optimaux.

2.4.2 Classification de texte

La classification de texte est l'une des tâches de NLP les plus largement utilisées en raison de ses nombreuses utilisations dans le monde réel telles que l'analyse des sentiments, la recherche d'informations, la classification des actualités, ou étiquetage de sujets.

Cette tâche consiste à attribuer des catégories ou des étiquettes à des documents textuels en fonction de leur contenu. Elle permet de rendre les informations non structurées plus accessibles et exploitables. En utilisant des algorithmes sophistiqués, la classification de texte aide à automatiser et à accélérer le processus de tri et d'organisation de grandes quantités de données textuelles, ce qui est essentiel dans de nombreux domaines, de la recherche académique à l'industrie [28].

2.4.3 Extraction d'informations

L'extraction d'informations est la tâche de traitement du langage naturel (NLP) qui extrait des informations sémantiques structurées à partir de texte. Ces informations incluent des relations binaires - par exemple, des interactions biochimiques entre deux protéines [29] ou des événements n-aires - c'est-à-dire des événements avec plus de deux arguments tels que des attaques terroristes, où

chaque attaque est associée à plusieurs arguments, y compris l'emplacement de l'attaque, l'identité de l'attaquant, le nombre de victimes, le montant des dommages matériels, et ainsi de suite [30]. L'extraction d'informations permet de nombreuses applications réelles importantes telles que la découverte de traitements potentiels pour les maladies ou la surveillance des attaques terroristes à partir de documents de presse.

2.4.4 Extraction de relations

2.4.4.1 Introduction

L'extraction de relations est une sous tache de l'extraction d'informations (IE) qui vise à identifier et à extraire les liens sémantiques entre différents éléments dans un texte. Ces éléments peuvent être des entités telles que des personnes, des lieux ou des événements, et les relations entre eux peuvent être diverses, allant des simples associations binaires aux structures plus complexes impliquant plusieurs entités. L'objectif principal de l'extraction de relations est de transformer le texte non structuré en données exploitables, facilitant ainsi la compréhension automatique des informations contenues dans les documents textuels. Cette tâche est cruciale dans de nombreux domaines, notamment la réponse aux questions et l'extraction de biotexte [31].

2.4.4.2 Les approches du RE

Plusieurs approches existent pour l'extraction de relations [32] :

— **Approche basé sur les règles**

Ces méthodes sont également appelées méthodes basées sur des patterns (motifs) construits manuellement. Ces types de méthodes définissent un ensemble de patterns d'extraction pour un ensemble prédéfini de relations. Ensuite, ces patterns d'extraction sont comparés au texte. Si un pattern correspond, une relation correspondant à ce pattern est trouvée dans le texte.

Par exemple un pattern pour les hyponymes comme 'such X as Y' avec le texte 'such actors as angelina' donne hyponyme(actor,angelina).

Les méthodes basées sur des règles nécessitent une expertise du domaine et des connaissances linguistiques pour définir des modèles d'extraction. Ces méthodes sont spécifiques à un domaine, reposant sur une structure de document fixe et des relations cibles prédéfinies. Lorsqu'on passe d'un domaine à un autre, il devient nécessaire de redéfinir les relations cibles et les modèles d'extraction. Par conséquent, les méthodes basées sur des règles demandent un effort manuel considérable et ne conviennent pas pour des corpus hétérogènes.

— **Approche basé sur l'apprentissage non supervisé**

Les méthodes non supervisées ne nécessitent pas de données annotées. La plupart des méthodes non supervisées d'extraction de relations utilisent une approche basée sur le clustering. L'une des premières approches non supervisées d'extraction de relations basée sur le clustering ont utilisé un étiqueteur d'entités nommées pour extraire les entités afin de se concentrer uniquement sur les relations avec les entités nommées mentionnées. Les étapes principales de la méthode d'apprentissage non supervisée sont :

1. Identification des entités nommées dans le corpus de texte
2. Identification des entités nommées co-occurrentes et de leur contexte
3. Regroupement des paires d'entités en fonction de la similarité de leur contexte
4. Attribution d'un nom de relation sémantique à chaque groupe.

— **Approche basé sur l'apprentissage supervisé**

Les méthodes supervisées nécessitent une grande quantité de données d'entraînement, annotées avec un ensemble d'entités et de relations. Elles utilisent ces données d'entraînement pour former un classificateur, qui extraira ensuite les relations des données de test. Il existe deux types de méthodes supervisées : les méthodes basées sur les caractéristiques et les méthodes basées sur les noyaux.

— Les méthodes basées sur les caractéristiques (features) : Un ensemble de caractéristiques est généré pour chaque relation dans les données d'entraînement, puis un classificateur est entraîné à extraire une nouvelle instance de relation.

— Les méthodes basées sur les noyaux (kernal) : Dans les approches basées sur les noyaux, des fonctions de noyau sont utilisées pour déterminer la similarité entre deux représentations d'instances de relation.

— **Approche basé sur l'apprentissage semi-supervisé**

La création de données pour les méthodes supervisées d'extraction de relations implique des coûts, des efforts et du temps. Cependant, les méthodes supervisées peuvent automatiser le processus de génération de données étiquetées grâce à des algorithmes de bootstrap. Cette approche offre deux avantages clés :

— Elle réduit l'effort nécessaire pour créer des données étiquetées

— Elle tire parti des données non étiquetées disponibles gratuitement.

L'algorithme de bootstrap repose sur une grande quantité de données non étiquetées et un petit ensemble d'instances de départ représentant le type de

relation souhaité. Par exemple, pour extraire la relation "CapitaleDe", des exemples de départ comme (New Delhi, Inde), (Canberra, Australie) et (Londres, Angleterre) peuvent être utilisés pour développer un modèle d'extraction. Avec ces exemples de départ en entrée, l'algorithme de bootstrap est conçu pour identifier des relations similaires impliquant des paires d'entités telles que (Paris, France). La Figure 7 illustre le modèle pour extraire des motifs et des tuples de départ en utilisant une approche d'apprentissage semi-supervisé.

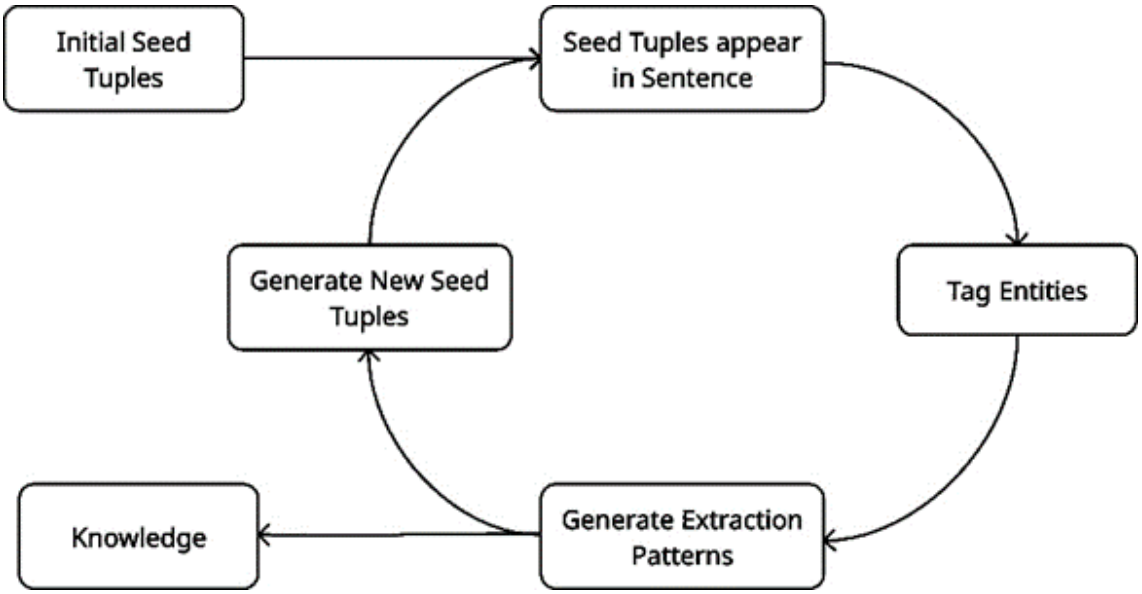

FIGURE 2.6 – Etapes pour l'apprentissage semi-supervisé de RE

— **Approche de supervision distante**

Ces méthodes sont également appelées méthodes supervisées faiblement ou basées sur la connaissance. Les chercheurs ont proposé une méthode dans laquelle les données d'entraînement sont automatiquement générées en alignant le texte avec une base de connaissances (KB), ce qui élimine le besoin d'étiquetage manuel. La supervision distante repose sur l'idée que si deux entités partagent une relation dans une KB, toutes les phrases mentionnant ces entités pourraient exprimer cette relation.

La supervision distante utilise une base de connaissances comme Freebase pour extraire les relations entre entités. Lorsqu'une paire d'entités apparaît à la fois dans une phrase et dans la KB, la phrase est liée de manière heuristique à la relation correspondante de la KB. Par exemple, dans la phrase "Bill Gates est le fondateur de Microsoft", si "Bill Gates" et "Microsoft" sont répertoriés comme un triplet (entité1 : Bill Gates, entité2 : Microsoft, relation : fondateur_de) dans Freebase, alors ces entités représentent la relation "fondateur_de".

— **Approche basé apprentissage profond**

L'utilisation de l'apprentissage profond dans l'extraction de relations a

considérablement révolutionné la façon dont nous abordons cette tâche. Les réseaux de neurones profonds permettent une représentation complexe et hiérarchique des données textuelles, ce qui les rend particulièrement adaptés à la capture de nuances et de contextes subtils présents dans les relations entre entités. Par exemple, les architectures telles que les réseaux de neurones récurrents (RNN) ou les transformers peuvent prendre en compte la séquentialité du langage naturel, permettant ainsi une compréhension plus profonde des dépendances contextuelles. De plus, les modèles de langue pré-entraînés, comme BERT, ont démontré leur capacité à capturer des informations sémantiques riches, ce qui est crucial pour l'extraction précise des relations.

2.4.4.3 Langues à faible ressource

Dans les langues riches en ressources telles que l'anglais, une grande quantité de corpus étiquetés est disponible pour entraîner des modèles pour différentes tâches de traitement du langage naturel (TALN). Les langues à faible ressource comme l'arabe, l'hindi, etc. disposent de corpus étiquetés très limités. Les progrès récents des modèles d'apprentissage profond pour l'extraction de relations ont obtenu des résultats prometteurs ; cependant, la performance des modèles diminue considérablement lorsque le nombre d'exemples d'entraînement diminue [32].

2.4.5 Similarité des phrases

2.4.5.1 Introduction

La similarité des phrases (sentence similarity en anglais) est la tâche qui consiste à déterminer à quel point deux textes sont similaires. Les modèles de similarité des phrases convertissent les textes d'entrée en vecteurs (embeddings) qui capturent les informations sémantiques et calculent à quel point ils sont proches (similaires) entre eux. Cette tâche est particulièrement utile pour la recherche d'informations et le clustering [33].

2.4.5.2 Approches pour la similarité des phrases

Plusieurs méthodes existent :

— **Approches basé vecteurs clairsemé**

Ces approches utilisant les représentations discutées précédemment, Bag of words et tf-idf. L'utilisation des représentations en sac de mots (Bag of Words) et du TF-IDF (Term Frequency-Inverse Document Frequency) pour mesurer la similarité des phrases est une méthode couramment employée en traitement automatique des langues. Le modèle sac de mots transforme un

texte en une matrice de termes, où chaque phrase est représentée par la fréquence des mots qu'elle contient. Bien que simple, cette approche ignore l'ordre des mots, ce qui peut limiter sa capacité à saisir le sens contextuel. Pour améliorer cette représentation, le TF-IDF pondère les fréquences des termes en tenant compte de leur importance dans le corpus, réduisant l'impact des mots courants et accentuant ceux plus significatifs. En utilisant ces représentations, les phrases peuvent être comparées à l'aide de mesures de similarité telles que le cosinus, permettant d'évaluer la proximité sémantique entre elles. Bien que ces méthodes soient moins sophistiquées que les modèles d'apprentissage profond modernes, elles restent efficaces et largement utilisées pour des tâches telles que la recherche d'informations, le clustering de textes et la classification de documents.

L'utilisation des représentations en sac de mots (Bag of Words) et du TF-IDF pour mesurer la similarité des phrases présente plusieurs inconvénients, notamment la création de vecteurs clairsemés (sparse). Ces méthodes transforment chaque phrase en un vecteur de grande dimension où chaque dimension correspond à un mot unique du corpus. Étant donné que chaque phrase ne contient qu'un sous-ensemble des mots possibles, la plupart des dimensions de ces vecteurs sont nulles, ce qui conduit à des vecteurs extrêmement clairsemés.

Cette sparsité pose plusieurs problèmes. Premièrement, elle rend difficile la capture des relations sémantiques entre les mots, car les représentations ne prennent pas en compte l'ordre des mots ni les contextes dans lesquels ils apparaissent. Deuxièmement, la haute dimensionnalité des vecteurs peut entraîner des inefficacités en termes de stockage et de calcul, rendant les opérations de similarité, comme la mesure de la similarité cosinus, plus coûteuses et moins robustes. Enfin, les vecteurs clairsemés sont sensibles au bruit et peuvent manquer de généralisation, ce qui limite leur efficacité pour des tâches complexes nécessitant une compréhension fine des nuances linguistiques. Ces limitations ont conduit à l'adoption de méthodes plus avancées, telles que les embeddings de mots et les modèles de langue pré-entraînés, qui offrent des représentations denses et contextuelles mieux adaptées à la tâche de similarité de phrases.

— **Approches basé vecteurs dense**

L'utilisation des embeddings a transformé la manière dont nous abordons la similarité des phrases en offrant des représentations vectorielles denses et contextuelles des mots et des documents. Les word embeddings, tels que Word2Vec et GloVe, capturent des relations sémantiques entre les mots en les plaçant dans un espace vectoriel continu où des mots similaires sont proches les uns des autres. Ces représentations surpassent les modèles

traditionnels comme le sac de mots et TF-IDF en considérant le contexte dans lequel les mots apparaissent. Cependant, les word embeddings présentent des limitations. Ils ne capturent que les relations de mots isolés et ignorent les nuances contextuelles des phrases. De plus, les embeddings statiques ne changent pas en fonction du contexte, ce qui peut conduire à des ambiguïtés pour des mots ayant plusieurs sens.

Pour pallier ces problèmes, des méthodes comme Doc2Vec ont été développées pour générer des embeddings de documents en prenant en compte l'ensemble de la phrase ou du texte. Doc2Vec produit des vecteurs denses pour des phrases, des paragraphes ou des documents entiers, offrant ainsi une meilleure capture des contextes larges et des relations sémantiques plus globales. Néanmoins, Doc2Vec peut être complexe à former et nécessite une quantité substantielle de données pour produire des représentations de haute qualité.

Des avancées plus récentes, telles que les modèles de langage contextuels comme BERT et SBERT, utilisent des techniques de pré-entraînement sur de vastes corpus textuels pour créer des embeddings qui changent en fonction du contexte. Ces modèles offrent des représentations contextuelles riches, capturant les nuances fines des phrases et permettant une compréhension approfondie du langage naturel. Malgré leurs performances impressionnantes, ces modèles nécessitent une grande puissance de calcul pour l'entraînement et l'inférence, ce qui peut être un obstacle pour certaines applications.

2.4.6 Détection de contradictions

La détection de contradictions est une tâche difficile dans le domaine du traitement du langage naturel en raison de la variété des façons dont les contradictions apparaissent dans les textes [34]. Elle implique l'identification et le traitement des énoncés qui sont mutuellement exclusifs ou en conflit de signification.

Une approche courante de la détection de contradiction consiste à utiliser des modèles d'apprentissage automatique, en particulier des réseaux neuronaux. Ces modèles sont entraînés sur des ensembles de données annotés contenant des paires de phrases étiquetées comme contradictoires ou non-contradictoires [35]. Des techniques telles que les réseaux siamois [36] ou les architectures basées sur les transformateurs comme BERT sont utilisées à cette fin. En apprenant des représentations contextuelles des phrases, ces modèles peuvent capturer les relations sémantiques et identifier les contradictions [37].

2.5 Systèmes de recommandation

2.5.1 Introduction

Les systèmes de recommandation sont des outils essentiels dans divers domaines, allant du commerce en ligne aux plateformes de streaming, en passant par les réseaux sociaux. Une definition des Systèmes de recommandation est la suivante : "Outils, logiciels, et techniques qui fournissent des suggestions personnalisées, guidant l'utilisateur, dans un grand espace de données, vers des ressources susceptibles de l'intéresser" [38]. Ils utilisent des techniques d'apprentissage automatique et de traitement du langage naturel pour analyser les préférences et les comportements des utilisateurs afin de proposer des contenus ou des produits pertinents [39]. Par exemple, Amazon et Netflix emploient des algorithmes de filtrage collaboratif et de filtrage basé sur le contenu pour améliorer l'expérience utilisateur [40]. En combinant des méthodes telles que les réseaux de neurones et les modèles de graphes de connaissances, ces systèmes peuvent fournir des recommandations personnalisées et contextuellement adaptées [41]. De plus, les avancées récentes dans l'utilisation des modèles de transformateurs, comme BERT et GPT, ont permis d'améliorer considérablement la précision des recommandations en comprenant mieux le contexte et les intentions des utilisateurs [42].

2.5.2 Les types des systèmes de recommandation

— **Filtrage basé sur le contenu (Content-based filtering)**

Le filtrage basé sur le contenu est une technique clé dans les systèmes de recommandation, utilisée pour proposer des éléments similaires à ceux que l'utilisateur a déjà appréciés. Contrairement au filtrage collaboratif, qui repose sur les préférences des autres utilisateurs, le filtrage basé sur le contenu analyse les caractéristiques des éléments eux-mêmes pour fournir des recommandations. Par exemple, dans le domaine de la musique ou des films, les systèmes de recommandation peuvent utiliser des informations telles que les genres, les acteurs, ou les artistes pour suggérer des contenus similaires [43]. Cette méthode utilise souvent des techniques de traitement du langage naturel pour extraire et analyser les caractéristiques des textes, telles que les descriptions de produits ou les résumés de films [44]. De plus, l'intégration des réseaux de neurones et des modèles de représentation de texte comme TF-IDF ou les embeddings de mots a permis d'améliorer la précision et la pertinence des recommandations basées sur le contenu [45].

— **Filtrage collaboratif (Collaborative filtering)**

La mise en œuvre la plus simple et la plus originale de cette approche

consiste à recommander à un utilisateur actif les items que d'autres utilisateurs ayant des goûts similaires ont aimés dans le passé. La similarité de goût entre deux utilisateurs est calculée en fonction de la similarité de leur historique de notation. Le filtrage collaboratif est la technique la plus populaire et la plus répandue dans les systèmes de recommandation [46].

2.5.3 Recommandation des questions

La recherche spécifique dédiée uniquement aux systèmes de recommandation de questions dans l'analyse des entretiens des forces de l'ordre est limitée, mais le domaine le plus large des systèmes de recommandation et de recherche d'informations fournit des méthodologies pertinentes applicables à ce domaine. les communautés de questions comme Quora utilisent les techniques de NLP et d'apprentissage automatique pour analyser les données textuelles et suggérer des questions pertinentes en fonction du contexte, des données historiques et des connaissances du domaine [47].

2.6 Travaux Connexes

Dans le domaine de l'analyse des entretiens des forces de l'ordre, en particulier en ce qui concerne la détection des contradictions et la recommandation de questions, il existe une absence notable de systèmes ou de projets existants directement comparables à la portée et aux objectifs de notre travail. Alors que les technologies de l'intelligence artificielle (IA) ont été largement adoptées dans divers secteurs et applications, leur intégration dans le contexte spécialisé de l'analyse des entretiens des forces de l'ordre reste relativement peu explorée.

Malgré la reconnaissance croissante du potentiel de l'IA pour améliorer les processus d'investigation, notamment dans des tâches telles que l'examen de l'écriture manuscrite [48], la littérature et les systèmes existants ignorent largement les exigences nuancées de l'analyse des entretiens des forces de l'ordre. L'absence de systèmes dédiés à la détection des contradictions et à la recommandation de questions dans ce domaine souligne la nécessité d'approches innovantes adaptées aux défis spécifiques et aux objectifs des agences d'investigation.

De plus, bien que les systèmes pilotés par l'IA pour l'analyse textuelle et les applications de traitement automatique du langage naturel (TALN) aient été largement étudiés dans d'autres domaines, tels que le service client ou les soins de santé, leur adaptation aux subtilités de l'analyse des entretiens des forces de l'ordre est limitée. Les recherches et projets existants dans des domaines adjacents ne parviennent pas à répondre aux exigences uniques de l'analyse des entretiens dans les contextes d'enquête, laissant un écart significatif dans les capacités requises pour des pratiques d'investigation complètes et efficaces.

2.7 Conclusion

Chapitre 3

Conception

3.1 Introduction

Dans ce chapitre, nous détaillerons la conception de notre système. Nous commençons par présenter les diagrammes de cas d'utilisations et de classes pour avoir une vue globale du système. Puis nous passons à nos modules de détection de la contradiction et de recommandation des questions.

3.2 Diagramme de cas d'utilisation

3.2.1 Diagramme de cas d'utilisation de l'acteur « Agent »

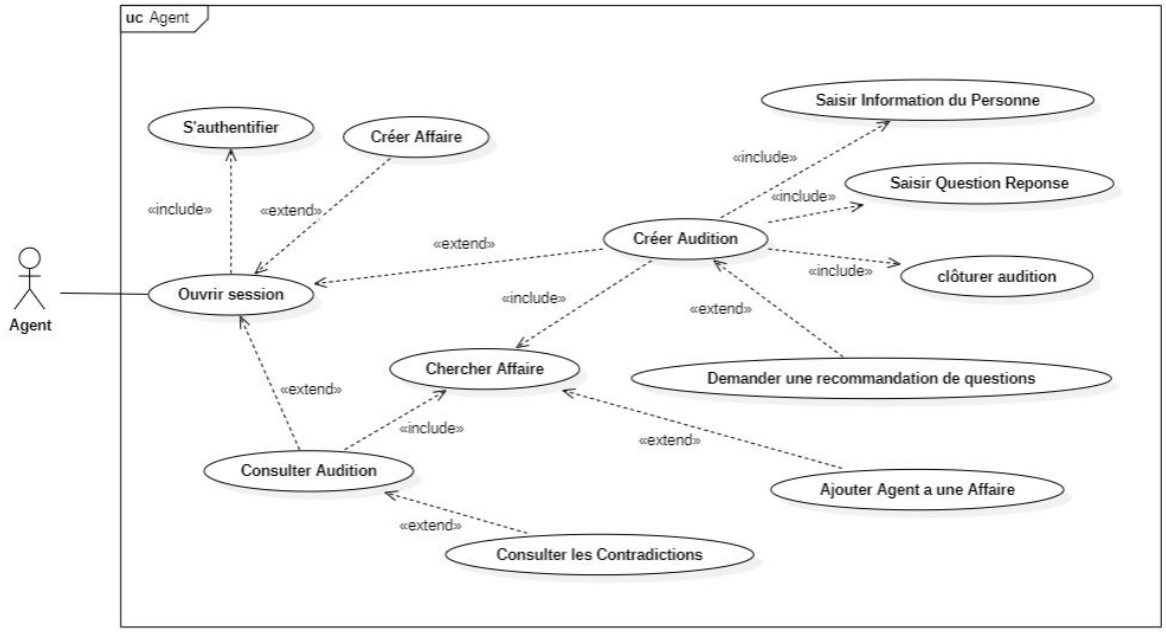

FIGURE 3.1 – Diagramme de cas d'utilisation de l'acteur « Agent »

3.2.2 Diagramme de cas d'utilisation de l'acteur « Administrateur »

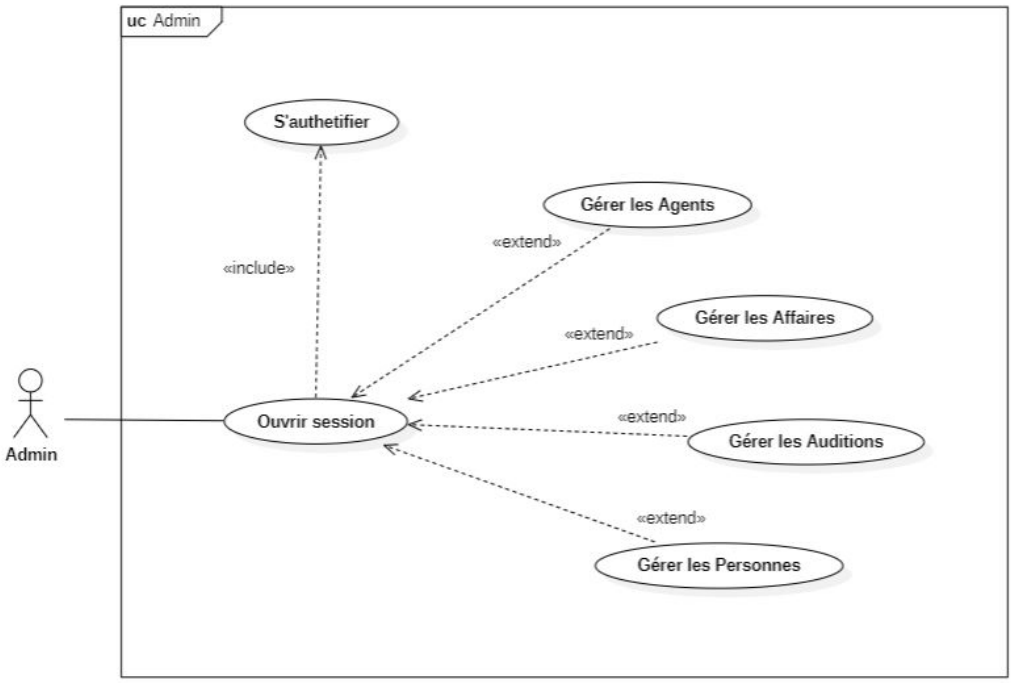

FIGURE 3.2 – Diagramme de cas d'utilisation de l'acteur « Administrateur »

3.3 Diagramme de classes

3.4 Approche de détection de la contradiction

3.4.1 Introduction

La détection des contradictions dans les déclarations est une étape cruciale pour assurer la fiabilité et la cohérence des informations recueillies durant les auditions. Cette section décrit la conception de notre module de détection des contradictions, en se concentrant sur les relations entre les personnes et les relations entre une personne et un lieu.

3.4.2 Description de l'approche

Dans notre approche nous nous concentrons spécifiquement sur la détection des contradictions dans les relations entre les personnes et d'autres entités. Cette décision découle de l'observation que les personnes sont intrinsèquement au centre de l'intérêt dans de telles enquêtes. Par conséquent, les contradictions qui

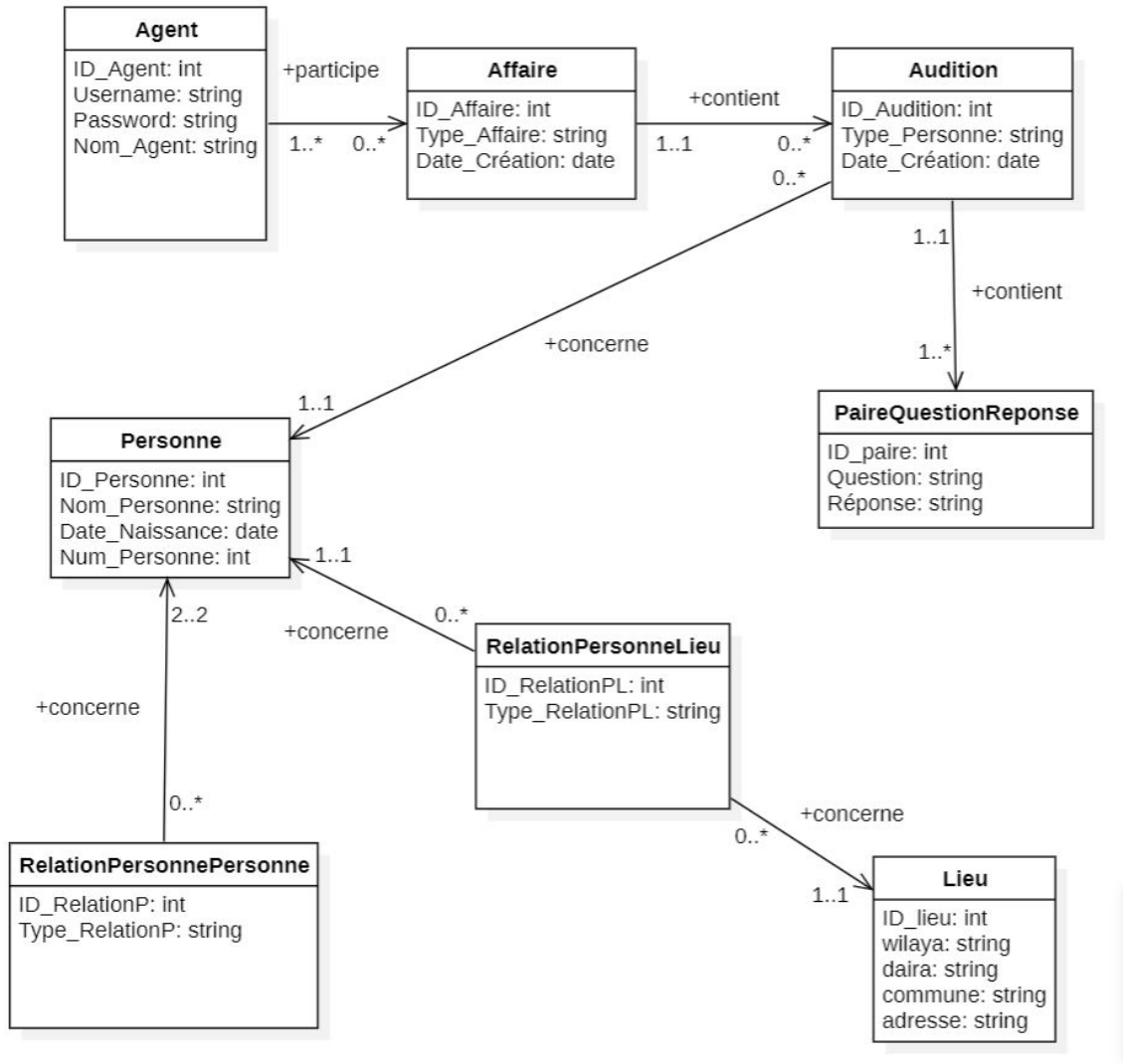

FIGURE 3.3 – Diagramme de Classe

n'impliquent pas les personnes sont souvent moins pertinentes ou significatives dans ce contexte. Ainsi, nous ne visons pas à résoudre la détection des contradictions de manière générale, mais nous nous concentrons plutôt sur les relations impliquant des individus.

Pour extraire et analyser efficacement ces relations, nous fournissons aux agents des questions prédéfinies conçues pour obtenir des réponses structurées. Ces questions sont élaborées pour recueillir systématiquement des informations sur les interactions, les associations et les événements impliquant les sujets des auditions. En utilisant cette méthode, nous veillons à ce que les données collectées soient à la fois complètes et cohérentes, facilitant ainsi une détection des contradictions plus précise, centrée sur les éléments essentiels de l'enquête.

3.4.3 Relations entre Personnes

Pour les relations entre personnes, nous avons mis en place un ensemble de questions prédéfinies que l'agent peut utiliser. Par exemple, une question pourrait

être "Quelle est votre relation avec... ?" L'agent peut ensuite entrer le nom de la personne concernée pour compléter la question. La réponse de la personne est prise telle quelle et classifiée en fonction d'un ensemble de relations prédéfinies telles que "ami", "collègue", "parent", etc.

Nous utilisons des embeddings de mots pour la classification des réponses. Les embeddings de mots permettent de représenter les mots dans un espace vectoriel continu, facilitant ainsi la comparaison et la classification des réponses basées sur leur similarité sémantique.

Processus :

1. L'agent pose une question prédéfinie concernant la relation entre deux personnes.
2. La personne fournit une réponse.
3. La réponse est transformée en vecteur à l'aide des embeddings de mots.
4. Le vecteur est comparé aux vecteurs de relations prédéfinies pour classifier la réponse.
5. La relation identifiée est stockée dans le système.

3.4.4 Relations entre une Personne et un Lieu

Pour les relations entre une personne et un lieu, nous avons également des questions prédéfinies comme "Où étais-tu le ... ?" L'agent peut spécifier l'année, le mois, le jour, l'heure et la durée. La réponse de la personne est formatée selon des catégories de localisation telles que l'état, la ville, etc.

Processus :

1. L'agent pose une question prédéfinie concernant la présence de la personne à un lieu spécifique.
2. L'interviewé fournit une réponse en précisant les détails temporels et géographiques.
3. La réponse est stockée dans le système sous forme de relation entre la personne et le lieu, avec les détails temporels.

3.4.5 Détection de Contradiction

Une fois les relations stockées, chaque nouvelle paire question-réponse est analysée pour extraire la relation qu'elle contient. Cette relation est ensuite comparée aux relations existantes dans le système pour détecter d'éventuelles contradictions.

Processus de Détection :

— Extraction de la relation à partir de la nouvelle paire question-réponse.

— Comparaison de cette relation avec les relations existantes dans le système.

— Identification des contradictions si une nouvelle relation contredit une relation déjà enregistrée.

— Signalement des contradictions détectées pour une révision par l'agent.

3.4.6 Pourquoi notre approche ?

La détection des contradictions est une tâche complexe, surtout dans le contexte de données limitées et spécifiques comme les auditions en dialecte algérien. Dans cette section, nous expliquons les raisons qui nous ont conduits à adopter notre approche particulière pour la détection des contradictions.

— **Données en Arabe et en Dialecte Algérien**

Le choix de notre approche est largement influencé par la nature des données disponibles. Les données en arabe, et plus particulièrement en dialecte algérien, sont rares. La plupart des techniques de traitement du langage naturel sont développées et optimisées pour l'anglais ou d'autres langues avec une abondance de données annotées [32]. En raison de cette rareté, il est difficile d'appliquer des méthodes sophistiquées de traitement des relations qui nécessitent de grandes quantités de données pour l'entraînement.

— **Absence de Données d'audition**

Une autre raison pour laquelle nous avons choisi notre approche est l'absence de données d'audition existantes. La plupart des approches de détection des relations reposent sur des corpus larges et annotés, tels que Wikipedia, qui ne sont pas disponibles dans notre contexte. Notre méthode permet de contourner ce problème en utilisant des questions et réponses prédéfinies pour contrôler et classifier les relations.

— **Précision des Informations**

Nous avons besoin de stocker des informations très précises, ce que les autres méthodes ne permettent pas toujours. Les approches générales de détection de relations peuvent manquer de granularité et ne pas capturer les nuances spécifiques de chaque relation. En utilisant des questions et réponses prédéfinies, nous pouvons garantir une classification précise et cohérente des relations.

— **Nature des Données**

Les autres méthodes de détection de relations sont souvent conçues pour des données structurées comme celles de Wikipedia, qui décrivent des faits de manière descriptive et à la troisième personne. En revanche, nos données consistent en des questions et réponses interrogatives, ce qui change la nature du texte et pourrait réduire l'efficacité des méthodes traditionnelles. Notre

approche est spécialement conçue pour ce type de données, assurant une meilleure performance.

— **Collecte de Données Initiale**

Enfin, l'objectif principal de notre système est de collecter des données initiales. Une fois que nous aurons suffisamment de données, nous pourrons envisager d'autres approches de détection de relations. En attendant, notre méthode permet de construire une base de données solide et fiable, essentielle pour toute analyse future.

Notre approche de la détection des contradictions est adaptée à notre contexte spécifique de données rares en arabe, absence de données d'audition existantes, et besoin de précision dans les informations collectées. En utilisant des questions et réponses prédéfinies, nous pouvons garantir la qualité et la fiabilité des relations extraites, tout en posant les bases pour l'application future de méthodes plus avancées lorsque suffisamment de données seront disponibles.

3.4.7 Conclusion

La conception de détection des contradictions repose sur l'utilisation de questions et réponses prédéfinies pour contrôler les entrées, et l'application des embeddings de mots pour classifier les relations entre personnes. Pour les relations entre une personne et un lieu, nous utilisons des formats standardisés pour les réponses afin de faciliter le stockage et la comparaison. En détectant les contradictions, notre système assure une vérification rigoureuse des informations collectées, augmentant ainsi la fiabilité des données dans les enquêtes.

3.5 Approche de recommandation des questions

3.5.1 Introduction

Le module de recommandation des questions vise à assister les agents de la force publique pendant les interviews en générant des questions pertinentes et contextuelles. Ce module est crucial pour maintenir le flux de l'audition, surtout lorsque l'agent ne sait plus quelles questions poser pour obtenir des informations complètes et cohérentes de la part de l'interviewé.

3.5.2 Description de l'approche

Pour la recommandation de questions, nous adoptons une approche en deux étapes : le filtrage basé sur le contenu et la similarité de phrases.

1. **Filtrage basé sur le contenu**

Nous sélectionnons les auditions ayant des caractéristiques communes avec l'audition en cours. Les caractéristiques utilisées pour le filtrage incluent le type d'affaire (type d'enquête) et le type de personne (victime, suspect, témoin).

2. **Similarité de phrases**

   Pour chaque audition résultante du filtrage, nous comparons les paires question-réponse avec la dernière paire de l'audition en cours en utilisant un modèle de transformateur de phrases. Cela nous permet d'identifier les paires les plus similaires, et donc de recommander les questions suivantes à poser.

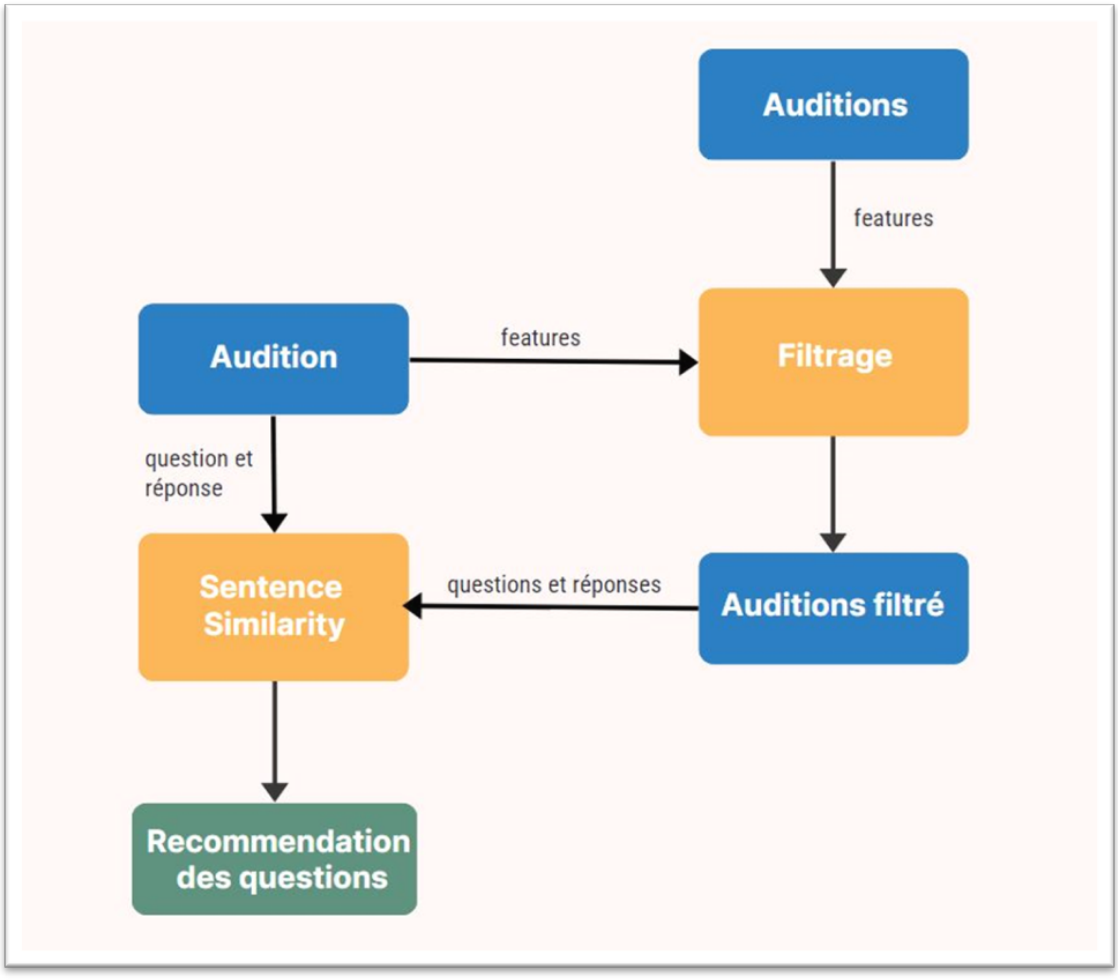

FIGURE 3.4 – Schéma global de la recommendation des questions

3.5.3 Filtrage

Le filtrage est la première étape cruciale pour réduire l'ensemble des auditions potentielles à un sous-ensemble plus gérable et pertinent. Nous utilisons les caractéristiques suivantes pour ce filtrage :

— **Type d'affaire :**

Cette caractéristique permet de s'assurer que les auditions sélectionnées concernent des affaires de même nature, ce qui augmente la pertinence des

questions recommandées.

— **Type de personne :**

Il est essentiel de filtrer selon le type de personne interrogée (victime, suspect, témoin) afin d'aligner le contexte des questions recommandées avec le rôle de la personne dans l'enquête.

Une fois le filtrage effectué, nous obtenons un ensemble d'auditions pertinentes qui serviront de base pour la phase suivante de similarité de phrases.

3.5.4 Similarité de phrases

Après avoir identifié les auditions pertinentes, nous procédons à une comparaison entre la dernière paire question-réponse de notre audition courante et les paires contenues dans ces auditions filtrées. Pour ce faire, nous utilisons un modèle de transformateur de phrases.

Modèle de transformateur de phrases :

Les modèles de transformateurs de phrases, tels que BERT (Bidirectional Encoder Representations from Transformers), sont des réseaux neuronaux pré-entraînés sur de grandes quantités de données textuelles. Ils sont capables de capturer les nuances contextuelles des mots et des phrases, rendant la comparaison sémantique beaucoup plus précise.

1. **Encodage des phrases :**

   Chaque phrase (question et réponse) est encodée en un vecteur dense de dimensions fixes, représentant son contenu sémantique.

2. **Calcul de similarité :**

   Les vecteurs de la dernière paire question-réponse de l'audition courante sont comparés avec les vecteurs des paires question-réponse des auditions filtrées en utilisant des mesures de similarité cosinus. La similarité retournée est un score entre 0 et 1, où 0 indique aucune relation sémantique et 1 indique des phrases identiques.

Le processus de similarité de phrases permet de recommander les questions qui sont les plus pertinentes et contextuellement appropriées pour poursuivre l'audition de manière efficace.

3.5.5 Pourquoi notre approche ?

— **Pertinence contextuelle :**

Notre méthode commence par un filtrage basé sur des caractéristiques telles que le type d'affaire et le type de personne (victime, suspect, témoin). Ce filtrage assure que seules les auditions pertinentes et contextuellement appropriées sont considérées pour la recommandation de questions. Cette

approche garantit que les questions proposées sont non seulement pertinentes mais aussi adaptées à la situation spécifique de l'audition.

— **Élimination du bruit :**

En filtrant les données avant d'appliquer des techniques de similarité, nous réduisons la quantité de données à traiter et éliminons les informations non pertinentes, ce qui améliore la performance globale du système et la qualité des recommandations.

— **Approche sémantique :**

Utiliser la similarité de phrases permet de comparer les paires question-réponse sur une base sémantique plutôt que simplement syntaxique. Cela signifie que nous pouvons identifier des questions qui sont similaires en termes de sens, même si elles sont formulées différemment, ce qui est crucial pour capturer la diversité des expressions linguistiques dans les interviews.

— **Adaptation aux dialogues interrogatifs :**

Contrairement à d'autres approches qui peuvent être basées sur des textes narratifs, la similarité de phrases est particulièrement adaptée pour les données de type question-réponse, typiques des interviews policières. Cela assure une meilleure performance dans notre contexte spécifique.

— **Performance avancée :**

Les modèles de transformers de phrases, tels que les modèles Sentence Transformers, offrent des performances de pointe pour la tâche de similarité de phrases. Ils sont capables de capturer des relations complexes entre les phrases grâce à leur architecture basée sur l'attention.

— **Généralisation :**

Ces modèles sont pré-entraînés sur de vastes corpus de données et peuvent généraliser efficacement à de nouveaux types de données, y compris les dialogues interrogatifs. Cela les rend particulièrement utiles dans notre contexte où les données d'entraînement spécifiques peuvent être limitées.

— **Flexibilité et précision :**

En utilisant des Sentence Transformers, nous pouvons obtenir des vecteurs d'embeddings de haute qualité qui représentent précisément les significations des phrases. Cela permet de comparer efficacement les questions et réponses en termes de leur contenu sémantique, améliorant ainsi la pertinence des recommandations.

Notre approche combine le filtrage basé sur le contenu et la similarité de phrases avec des transformers de phrases pour maximiser la pertinence et la précision des recommandations de questions dans les auditions policières. Cette méthodologie est spécialement conçue pour répondre aux défis uniques posés par les données de

dialogue interrogatif en arabe, tout en assurant une évolutivité et une adaptabilité pour de futures améliorations.

3.5.6 Conclusion

Ce processus en deux étapes assure que les questions recommandées sont non seulement pertinentes par rapport au contexte de l'affaire, mais aussi adaptées à la dynamique actuelle de l'audition.

3.6 Conclusion

Conclusion

Chapitre 4

Implémentation

4.1 Introduction

Ce dernier chapitre est dédié à la réalisation de notre système. L'objectif de ce chapitre est d'exposer les différentes techniques : langages, environnement et les outils de développement utilisés ainsi que les plates-formes de développement choisies. Puis on présente l'architecture du système et à la fin un ensemble de captures d'écrans pour illustrer ses différentes fonctionnalités.

4.2 Présentation de l'environnement de développement

4.2.1 Interface

— **React**

React est une bibliothèque JavaScript open source, développée par Facebook, pour la création d'interfaces utilisateur interactives et réactives. Elle permet de construire des composants d'interface utilisateur réutilisables et de gérer efficacement l'état et le rendu de ces composants.

Dans notre projet, nous avons utilisé React v18.2.0 pour développer l'interface utilisateur de notre application. React nous a permis de créer une interface utilisateur dynamique et réactive, en facilitant la gestion des états et des interactions utilisateur. En utilisant des composants réutilisables, nous avons pu développer rapidement et maintenir facilement notre code front-end.

— **Vite**

Vite est un outil de build et de développement rapide pour les projets front-end, conçu pour offrir une expérience de développement moderne et performante. Il supporte les frameworks populaires comme React, Vue, et

Svelte, et se distingue par sa capacité à fournir un rechargement à chaud ultra-rapide et des temps de compilation réduits.

Dans notre projet, nous avons utilisé Vite v5.2.0 pour gérer le processus de développement et de build de notre application React. Vite nous a permis de bénéficier d'un rechargement à chaud rapide, ce qui a considérablement amélioré notre productivité en offrant un retour immédiat sur les modifications de code. De plus, les temps de build optimisés ont réduit le temps nécessaire pour compiler et déployer notre application.

— **Redux Toolkit**

Redux Toolkit est une bibliothèque officielle pour la gestion de l'état dans les applications Redux, conçue pour simplifier l'écriture du code Redux et améliorer l'expérience des développeurs. Elle offre des abstractions et des utilitaires puissants pour réduire le boilerplate, faciliter la création des slices de l'état et gérer les effets secondaires.

Dans notre projet, nous avons utilisé Redux Toolkit v2.2.5 et react-redux v9.1.2 pour gérer l'état global de notre application React. Redux Toolkit nous a permis de structurer notre état de manière claire et maintenable, en définissant des slices pour différentes parties de notre application.

— **RadixUI et TailwindCSS**

RadixUI est une collection de composants d'interface utilisateur non stylisés et accessibles pour React, conçus pour offrir une base robuste et flexible à partir de laquelle construire des interfaces utilisateur personnalisées. TailwindCSS, quant à lui, est un framework CSS utilitaire qui permet de styliser des applications rapidement et efficacement en utilisant des classes prédéfinies.

Dans notre projet, nous avons utilisé Radix v1.0.1 pour fournir les composants de base de notre interface utilisateur React, en tirant parti de ses fonctionnalités d'accessibilité et de ses comportements par défaut. Nous avons ensuite utilisé TailwindCSS v3.4.3 pour styliser ces composants, en appliquant des classes utilitaires directement dans notre JSX. Cette combinaison nous a permis de créer une interface utilisateur cohérente et esthétique, tout en conservant une grande flexibilité dans le design.

— **NodeJS**

Node.js est un environnement d'exécution JavaScript open source, construit sur le moteur V8 de Chrome. Il permet d'exécuter du code JavaScript en dehors d'un navigateur, ce qui est particulièrement utile pour le développement d'outils de build et de gestion de projets front-end. Node.js est connu pour sa capacité à gérer des opérations d'E/S non bloquantes, et il est largement utilisé pour automatiser les tâches de développement front-end.

Dans notre projet, nous avons utilisé Node.js v20.11.1 et npm v9.5.0 pour gérer notre environnement de développement front-end. Node.js a servi de base pour exécuter des outils comme Vite et des gestionnaires de paquets tels que npm. Cela nous a permis de compiler et de packager notre code, de gérer les dépendances, et de lancer des serveurs de développement pour prévisualiser notre application en temps réel.

— **React-admin**

React Admin est un framework open source pour la création d'interfaces d'administration dans les applications React. Il est basé sur Material-UI et permet de développer rapidement des back-offices fonctionnels en fournissant des composants prêts à l'emploi pour les vues CRUD (Create, Read, Update, Delete), l'authentification, la gestion des permissions, et bien plus encore.

Dans notre projet, nous avons utilisé React Admin v4.16.18 pour construire l'interface d'administration de notre application. React Admin nous a permis de configurer rapidement des tableaux de bord, des formulaires et des listes de données, tout en intégrant des fonctionnalités complexes telles que la pagination, le tri, et la recherche. Grâce à ses abstractions puissantes, nous avons pu gérer les interactions avec notre API backend de manière fluide et structurée.

4.2.2 Back-end

— **Python**

Pour développer la partie Back-end, nous avons opté pour le langage de programmation Python v3.10 64-bit.

Python est un langage de programmation orienté objet, comparable à Perl, Ruby ou Java. Il est apprécié pour sa syntaxe simple, facilitant la lecture et l'écriture des programmes. Python est open source et peut être modifié et redistribué librement. Sa popularité a conduit à la création de nombreuses bibliothèques pour des tâches courantes telles que la connexion à des serveurs Web, la manipulation de textes, le traitement de données, et la gestion de fichiers. Notamment, il dispose de bibliothèques et frameworks d'apprentissage automatique largement utilisés.

— **Flask**

Flask est un micro-framework web en Python, léger et flexible, idéal pour le développement d'applications web et d'API. Il est basé sur Werkzeug et Jinja2, offrant une structure minimale mais extensible pour la création d'applications web. Flask permet aux développeurs de choisir les composants et les bibliothèques qu'ils souhaitent utiliser, favorisant ainsi une grande flexibilité et personnalisation.

Dans notre projet, nous avons utilisé Flask v3.0.3 et Flask-Cors v4.0.1 pour connecter le backend à l'interface utilisateur. Flask a été essentiel pour gérer les requêtes HTTP, permettre la communication entre le serveur et le frontend, et fournir les données nécessaires à l'interface.

— **gensim**

Gensim est une bibliothèque open source en Python, spécialisée dans le traitement du langage naturel (NLP) et l'apprentissage automatique. Elle est particulièrement reconnue pour ses capacités à traiter des modèles de représentation vectorielle de mots, tels que Word2Vec, Doc2Vec et FastText.

Dans notre projet, nous avons principalement utilisé Gensim v4.3.2 pour exploiter un modèle Word2Vec pré-entraîné, afin de classifier les relations entre différents termes. Gensim nous a permis de charger et de manipuler facilement le modèle Word2Vec, transformant les mots en vecteurs de haute dimension.

— **Sentence-Transformers**

Sentence-Transformers est une bibliothèque en Python qui permet de générer des représentations vectorielles pour des phrases ou des textes complets. Elle est basée sur les modèles de transformers, tels que BERT, RoBERTa et autres, et est spécialement optimisée pour calculer la similarité entre des phrases.

Dans notre projet, nous avons utilisé Sentence-Transformers v2.6.1 pour mettre en œuvre un modèle de similarité de phrases. Cela nous a permis de comparer des phrases en termes de similarité sémantique, en transformant les phrases en vecteurs denses et de haute dimension. Ces vecteurs nous ont aidés à évaluer les relations entre différentes phrases de manière précise et efficace.

— **PostgreSQL**

PostgreSQL est un système de gestion de base de données relationnelle open source, reconnu pour sa robustesse, sa fiabilité et ses nombreuses fonctionnalités avancées. Il prend en charge les transactions ACID, l'intégrité référentielle, les vues, les déclencheurs, les procédures stockées, et offre une extensibilité via des plugins.

Dans notre projet, nous avons utilisé PostgreSQL v16.3 pour stocker et gérer les données nécessaires à notre application. PostgreSQL a servi de backend de base de données, permettant un stockage structuré et une récupération efficace des données. Nous avons utilisé ses capacités avancées pour gérer des requêtes complexes et assurer l'intégrité des données.

— **pgvector**

pgvector est une extension pour PostgreSQL qui ajoute la prise en charge des vecteurs, permettant ainsi de stocker et de manipuler des données vectorielles directement dans la base de données. Elle est particulièrement utile pour les

applications nécessitant des recherches de similarité, telles que la recherche d'images, la recommandation de produits ou le traitement du langage naturel.

Dans notre projet, nous avons utilisé pgvector v0.7.2 pour stocker les représentations vectorielles générées par nos modèles de traitement du langage naturel et d'apprentissage automatique. Cette extension nous a permis d'effectuer des comparaisons et des recherches de similarité directement au niveau de la base de données, améliorant ainsi l'efficacité et la rapidité de nos opérations.

— **Psycopg2**

Psycopg2 est un adaptateur PostgreSQL pour le langage de programmation Python, permettant une interface fiable et performante pour interagir avec des bases de données PostgreSQL. Il est conçu pour être entièrement compatible avec les spécifications DB-API 2.0 de Python et prend en charge les fonctionnalités avancées de PostgreSQL.

Dans notre projet, nous avons utilisé Psycopg2 v2.9.9 pour connecter notre application Python à notre base de données PostgreSQL. Psycopg2 nous a permis d'exécuter des requêtes SQL, de récupérer des données, et de gérer les transactions de manière efficace et sécurisée. Grâce à ses capacités de gestion de la concurrence et de support des opérations asynchrones, nous avons pu optimiser les performances de notre application.

4.2.3 Logiciels

— **Visual Studio Code**

Visual Studio Code (VSCode) est un éditeur de code source puissant et léger développé par Microsoft. Il est conçu pour supporter plusieurs langages de programmation et offre une large gamme de fonctionnalités adaptées aux développeurs. Parmi ses principales caractéristiques, on trouve la coloration syntaxique, l'auto-complétion intelligente, et un débogueur intégré.

VSCode est extrêmement personnalisable grâce à ses nombreuses extensions disponibles sur le Marketplace, permettant d'ajouter des fonctionnalités spécifiques comme la gestion de versions, l'intégration de terminal, ou encore des outils de développement pour des frameworks et bibliothèques particuliers. Cet éditeur est open source, ce qui permet à la communauté de contribuer à son amélioration continue.

— **DBeaver**

DBeaver est un outil open source de gestion de bases de données, compatible avec de nombreux systèmes de gestion de bases de données relationnelles et non relationnelles tels que PostgreSQL, MySQL,

Oracle, SQLite, et bien d'autres. Il offre une interface utilisateur graphique intuitive pour administrer, développer et maintenir des bases de données.

Dans notre projet, nous avons utilisé DBeaver v24.1.0 pour administrer notre base de données PostgreSQL. DBeaver nous a permis de visualiser et de gérer les schémas, d'exécuter des requêtes SQL, de superviser les performances de la base de données et de manipuler les données de manière efficace. Grâce à ses fonctionnalités avancées et à son interface conviviale, nous avons pu effectuer des opérations complexes de manière plus rapide et plus intuitive.

— **Postman**

Postman est un outil de collaboration et de test pour les API, largement utilisé par les développeurs pour la conception, le test, et la documentation des API. Il offre une interface utilisateur conviviale qui permet de créer et d'envoyer des requêtes HTTP de manière simple et efficace, tout en visualisant les réponses reçues.

Dans le cadre de notre projet, nous avons utilisé Postman pour tester et valider nos endpoints API. Grâce à ses capacités de simulation de requêtes et de visualisation des réponses, nous avons pu identifier et corriger rapidement les erreurs dans notre backend. L'utilisation de collections et d'environnements dans Postman nous a permis de standardiser nos tests et de faciliter la transition entre les différentes phases de développement. De plus, les fonctionnalités de documentation intégrées nous ont aidés à maintenir une documentation claire et à jour pour nos API, facilitant ainsi la compréhension et l'utilisation par les autres membres de l'équipe.

— **Chrome DevTools**

Chrome DevTools est un ensemble d'outils de développement intégrés au navigateur Google Chrome. Ces outils sont essentiels pour les développeurs web, leur permettant de déboguer, tester et optimiser leurs applications web directement dans le navigateur.

Dans le cadre de notre projet, nous avons utilisé Chrome DevTools pour déboguer et optimiser notre interface utilisateur. Grâce à l'inspection du DOM et aux outils de console comme redux devtools, nous avons pu identifier et corriger rapidement les erreurs de rendu et les problèmes de script. Les outils de réseau nous ont permis d'analyser les requêtes API et de garantir des temps de réponse optimaux. Enfin, les outils de performance et de profilage nous ont aidés à améliorer la vitesse de chargement et l'efficacité de notre application, garantissant une expérience utilisateur fluide et réactive.

4.3 Les modèles utilisés

4.3.1 Word2vec CBOW

Pour la détection des relations personne personne, nous avons utilisé un modèle Word2Vec CBOW (Continuous Bag of Words) provenant d'un projet appelé AraVec, qui a développé plusieurs modèles de word embeddings pour l'arabe [49]. Le modèle que nous avons choisi est un modèle unigram de 300 dimensions, construit en utilisant des corpus de données provenant de Twitter.

La sélection de 300 dimensions pour notre modèle Word2Vec se justifie par plusieurs raisons :

— **Équilibre entre Performance et Efficacité**

Un modèle de 300 dimensions offre un bon compromis entre la capacité à capturer les nuances sémantiques et la complexité computationnelle. Des dimensions plus élevées pourraient capturer davantage de détails sémantiques, mais au coût d'une augmentation significative des ressources de calcul et de la mémoire nécessaire.

— **Standard de l'Industrie**

De nombreux travaux de recherche et applications pratiques utilisent des vecteurs de 300 dimensions, ce qui en fait un choix éprouvé pour de nombreux cas d'utilisation dans le traitement du langage naturel [50].

— **Qualité des Embeddings**

Des études ont montré que 300 dimensions sont souvent suffisantes pour représenter de manière efficace les relations sémantiques et syntaxiques entre les mots.

L'utilisation de corpus de données provenant de Twitter pour entraîner le modèle présente plusieurs avantages spécifiques à notre projet :

— **Langage Naturel et Informel**

Les données de Twitter sont caractérisées par un langage naturel et souvent informel, avec de nombreuses variations linguistiques, abréviations et expressions idiomatiques. Cela permet au modèle d'apprendre des représentations riches et diversifiées qui peuvent être utiles pour traiter des textes provenant de sources similaires.

— **Richesse du Corpus**

Twitter génère un volume massif de données textuelles, offrant une large couverture lexicale et contextuelle. Entraîner le modèle sur ces données permet de capturer une grande variété de contextes et de relations sémantiques.

— **Actualité et Variabilité**

Les tweets reflètent souvent des événements actuels, des tendances et des discussions populaires, ce qui rend les embeddings particulièrement adaptés pour des applications nécessitant une compréhension des contextes modernes et variés.

4.3.2 Transformateur de phrase

Pour la similarité des phrases, nous avons utilisé un modèle de transformateur de phrases appelé all-MiniLM-L6-v2 [26].

Nous avons choisi d'utiliser ce modèle car il offre un excellent compromis entre performance et efficacité. Le all-MiniLM-L6-v2 est un modèle compact et rapide qui permet de calculer les similarités entre phrases avec une grande précision, tout en réduisant les besoins en ressources computationnelles. Son architecture légère le rend particulièrement adapté pour des applications nécessitant une évaluation rapide des similarités de texte dans des environnements limités en ressources comme notre.

4.4 Présentation de l'application

Dans cette partie, nous présentons un ensemble d'interface qui représentent notre application.

L'interface de connexion est présentée comme le montre la figure suivante :

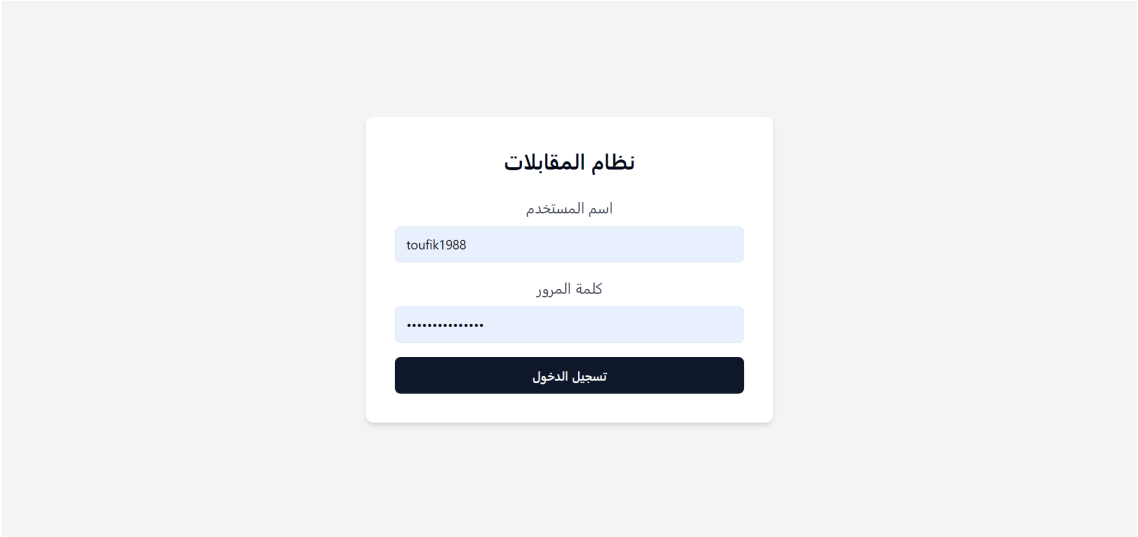

FIGURE 4.1 – Interface d'authentification pour l'agent

Après s'être authentifié, l'agent accède à la page d'accueil des affaires où il peut voir toutes les affaires disponibles pour lui. Comme illustré dans la figure suivante, il peut choisir de créer une nouvelle affaire, de consulter les affaires déjà créées, ou de se déconnecter.

L'agent peut créer de nouvelles affaires en entrant les informations de l'affaire (type d'affaire, description de l'affaire) et en cliquant sur "Créer une nouvelle affaire",

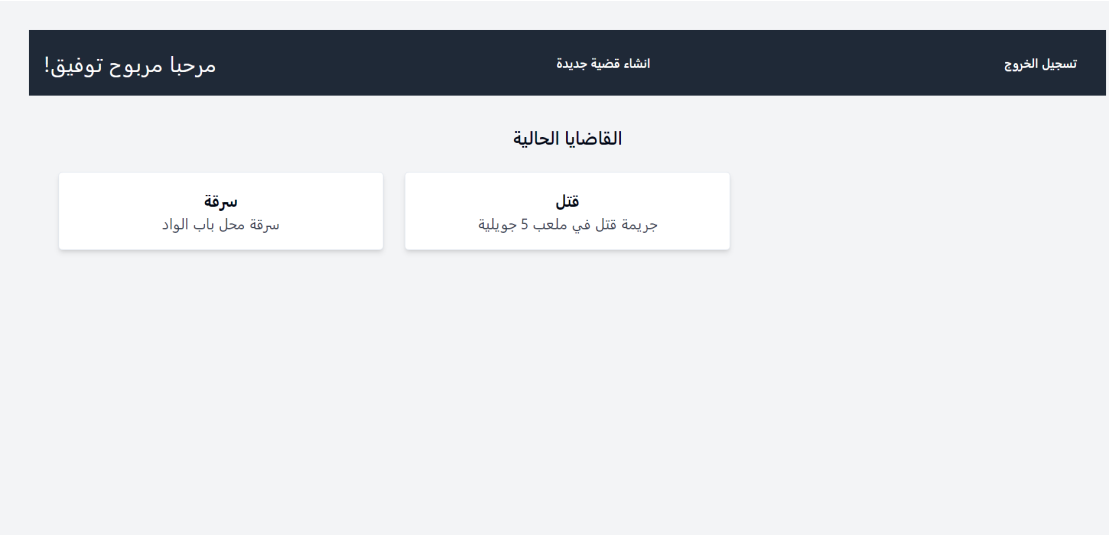

FIGURE 4.2 – Interface d'accueil pour l'agent

comme illustré dans la figure suivante.

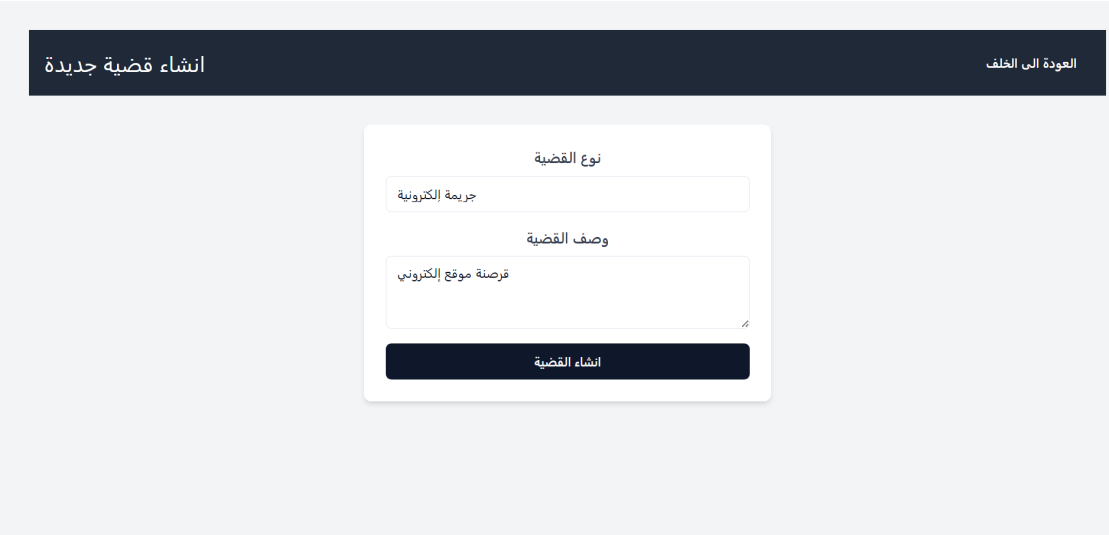

FIGURE 4.3 – Interface pour la création des affaires

Après avoir cliqué sur une affaire, l'agent est transféré à la page d'accueil de l'affaire où il peut choisir de voir les anciennes auditions de l'affaire, d'ajouter d'autres agents à l'affaire, de démarrer une nouvelle audition, ou de revenir simplement à la page d'accueil des affaires, comme montré dans la figure suivante.

Un agent peut ajouter d'autres agents à l'affaire en entrant leur nom d'utilisateur et en cliquant sur "Ajouter un agent".

Après avoir cliqué sur "Voir les anciennes auditions", l'agent accède à une page qui montre toutes les auditions réalisées auparavant concernant l'affaire sélectionnée.

Après avoir cliqué sur "Créer une nouvelle audition", l'agent voit une page contenant un formulaire où il saisit les informations de l'audition : type de personne (suspect, témoin, victime), nom complet, date de naissance et numéro. Il doit ensuite cliquer sur "Confirmer" pour ouvrir une nouvelle audition.

FIGURE 4.4 – Interface d'accueil d'une affaire

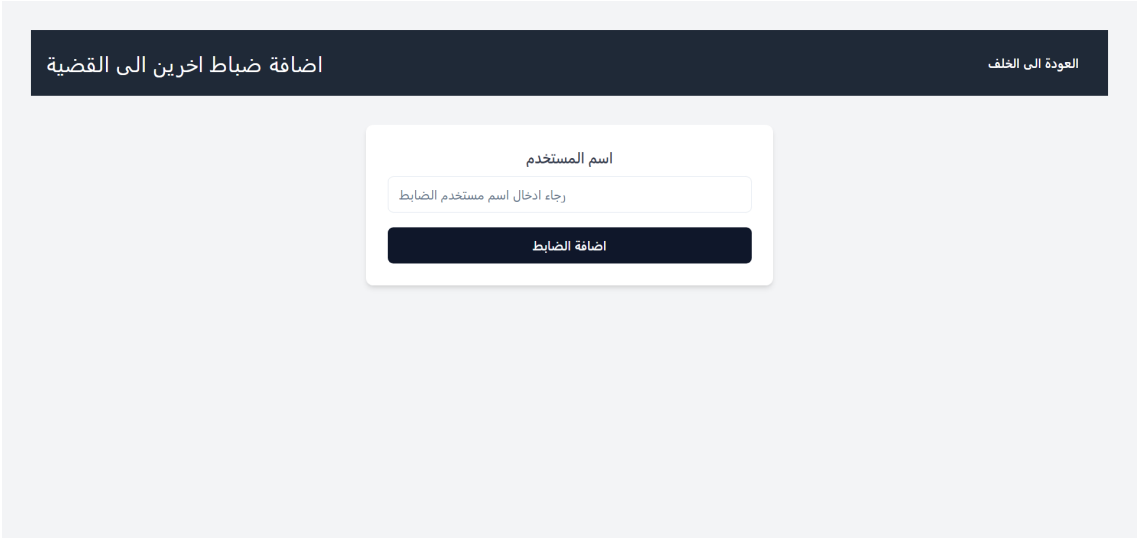

FIGURE 4.5 – Interface pour l'ajoute des agents a l'affaire

Après avoir cliqué sur "Confirmer", l'agent est dirigé vers la page de l'audition, où il peut saisir toutes les questions et réponses de l'audition.

Un agent peut saisir des questions non prédéfinies et leurs réponses.

L'agent peut utiliser une question prédéfinie comme montré dans la figure suivante. Pour les relations de type personne-personne, l'agent doit seulement saisir le nom de la personne et la réponse de l'interviewé.

L'agent peut sélectionner une question prédéfinie pour la relation personne-lieu. Il peut ensuite saisir la date exacte, l'heure, la période, ainsi que la réponse de l'interviewé en termes de wilaya, daïra, commune et adresse.

À tout moment, un agent peut demander une recommandation de question en cliquant sur le bouton "Demander une recommandation de question".

La figure suivante montre le résultat après que l'agent a cliqué sur le bouton "Demander une recommandation de question". Des questions recommandées apparaissent.

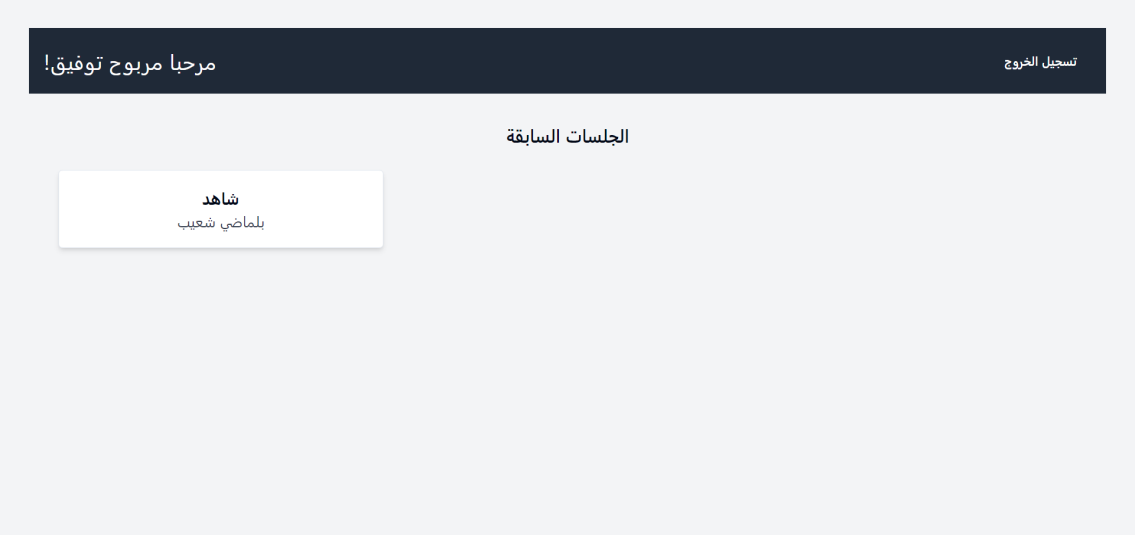

FIGURE 4.6 – Interface des Auditions déjà fait pour l'affaire

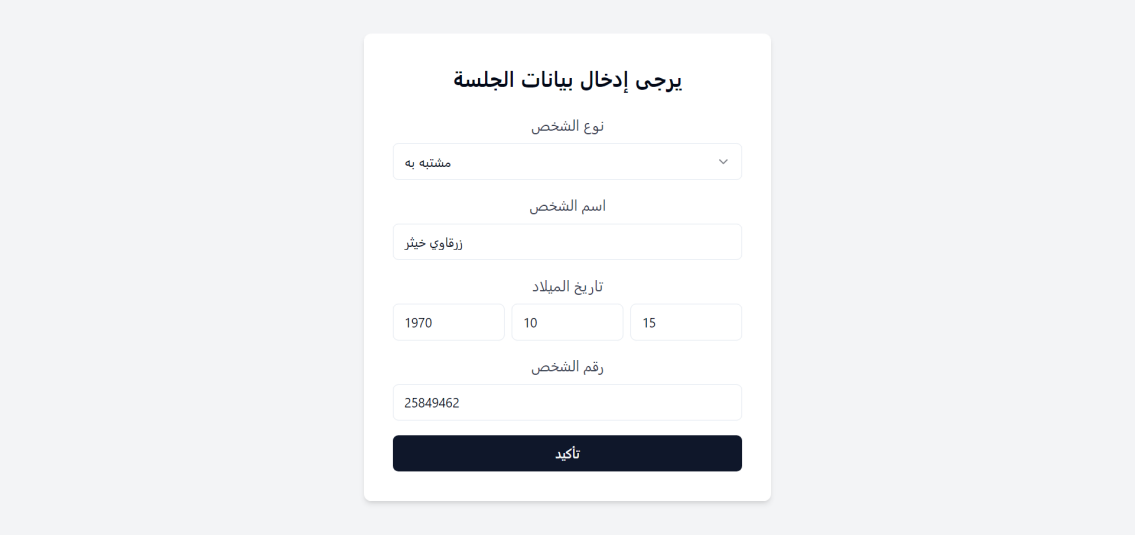

FIGURE 4.7 – Interface pour la création d'une nouvelle audition

Lorsque l'agent souhaite clôturer l'audition, il peut le faire en cliquant sur le bouton "Clôturer l'audition". L'interview est alors clôturée et le processus de détection des contradictions commence. La figure suivante montre qu'aucune contradiction n'a été détectée.

Pour les besoins de l'exemple, supposons qu'une nouvelle audition a été réalisée après la précédente avec la même personne. Cette audition contient deux contradictions par rapport à l'audition précédente, chaque contradiction étant pour un type de relation différent.

Après que l'agent a clôturé l'audition, des contradictions ont été détectées comme montré dans la figure suivante.

Interface d'authentification pour l'administrateur :

Après s'être authentifié, l'administrateur accède à la page d'accueil où il peut gérer tous les entités du système :

FIGURE 4.8 – Interface de saisie des questions et réponses

4.5 Conclusion

Conclusion

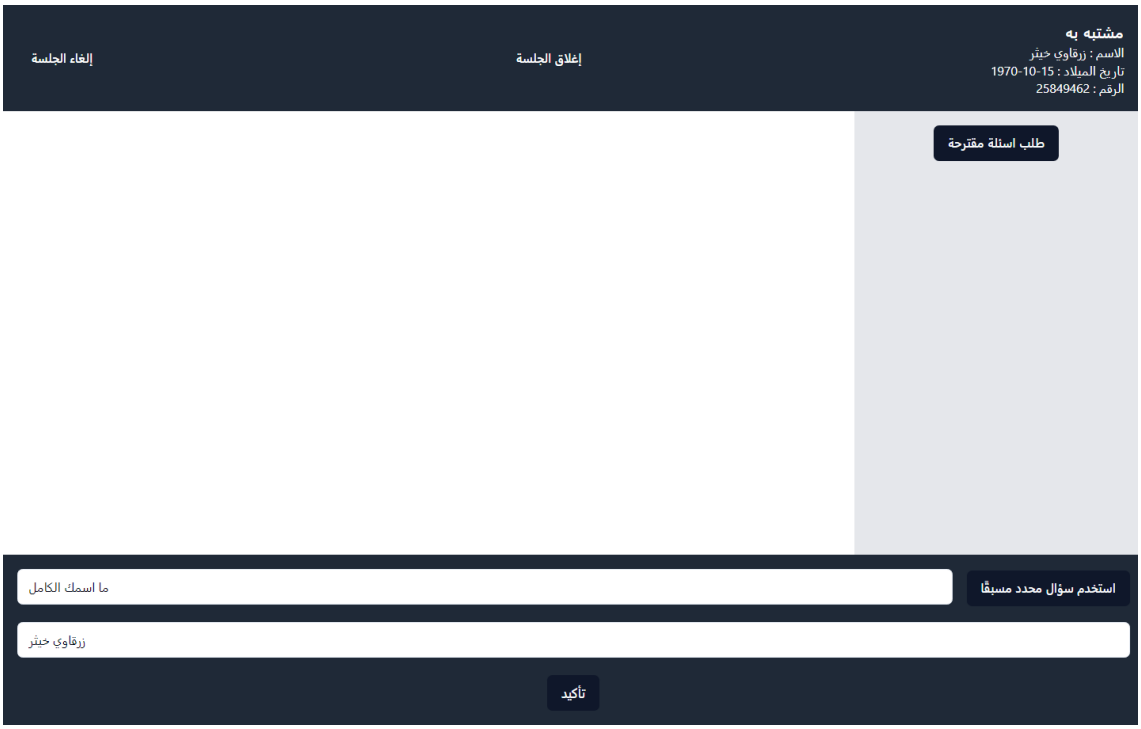

FIGURE 4.9 – saisie des question non prédéfinie

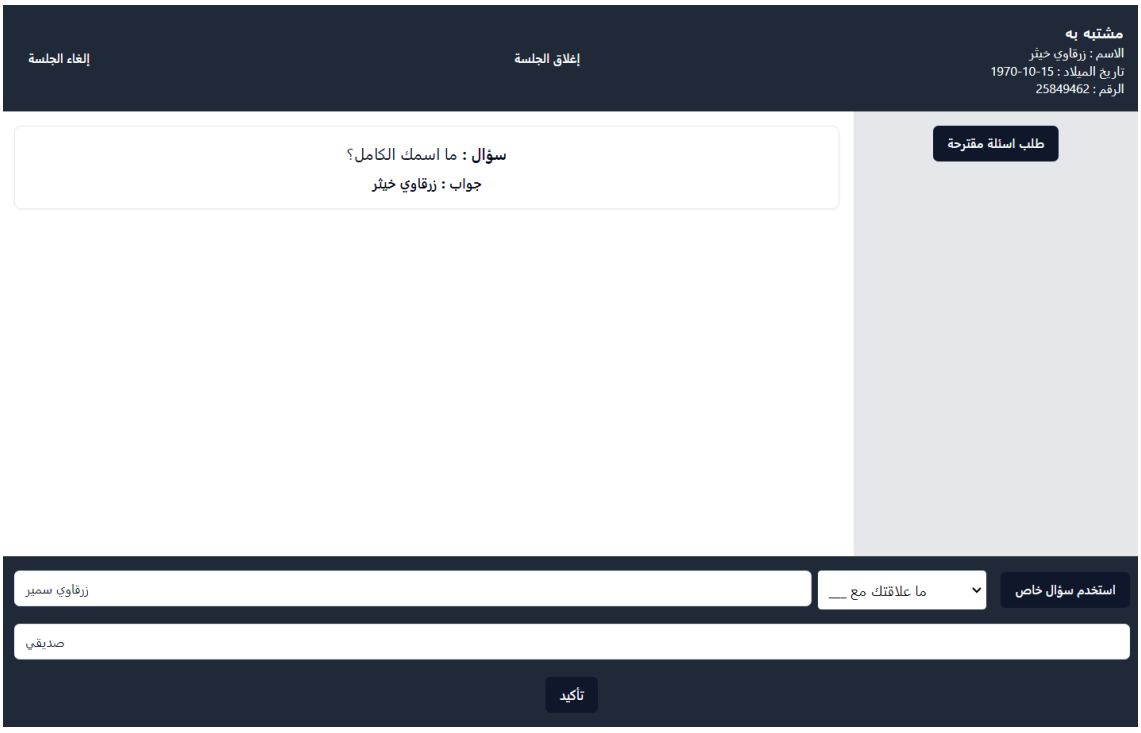

FIGURE 4.10 – saisie des questions prédéfinie, cas relation personne personne

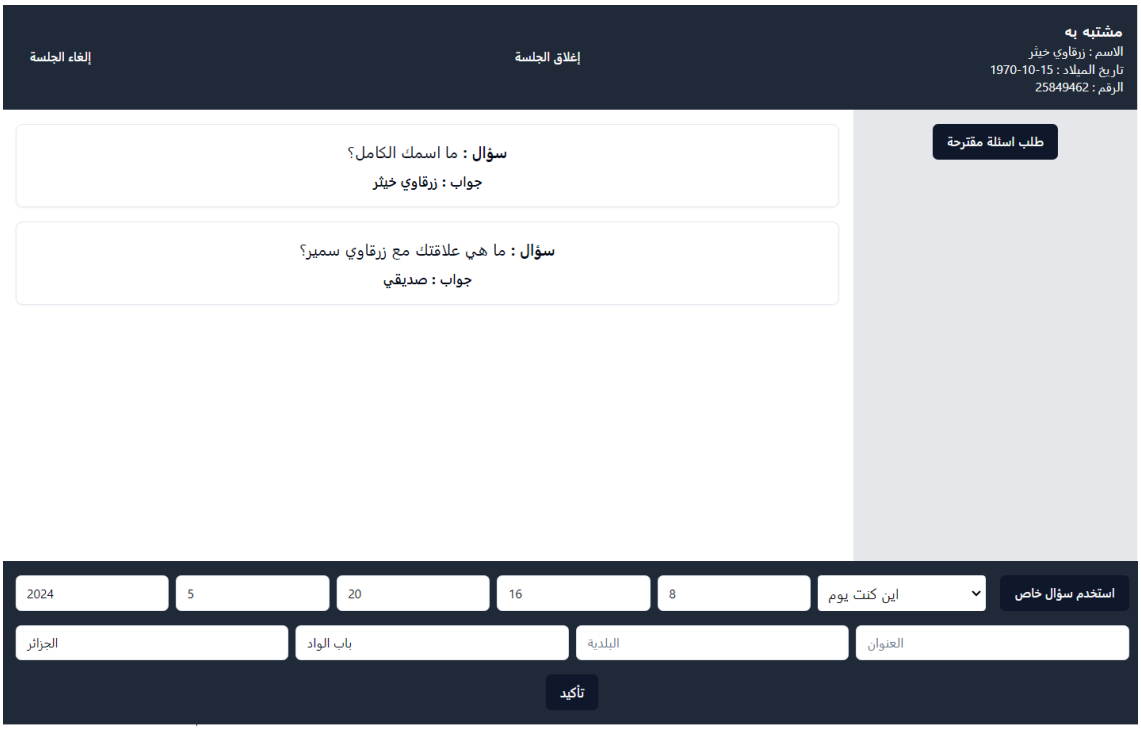

FIGURE 4.11 – saisie des questions prédéfinie, cas relation personne lieu

FIGURE 4.12 – Fin de saisie, avant recommandation de questions

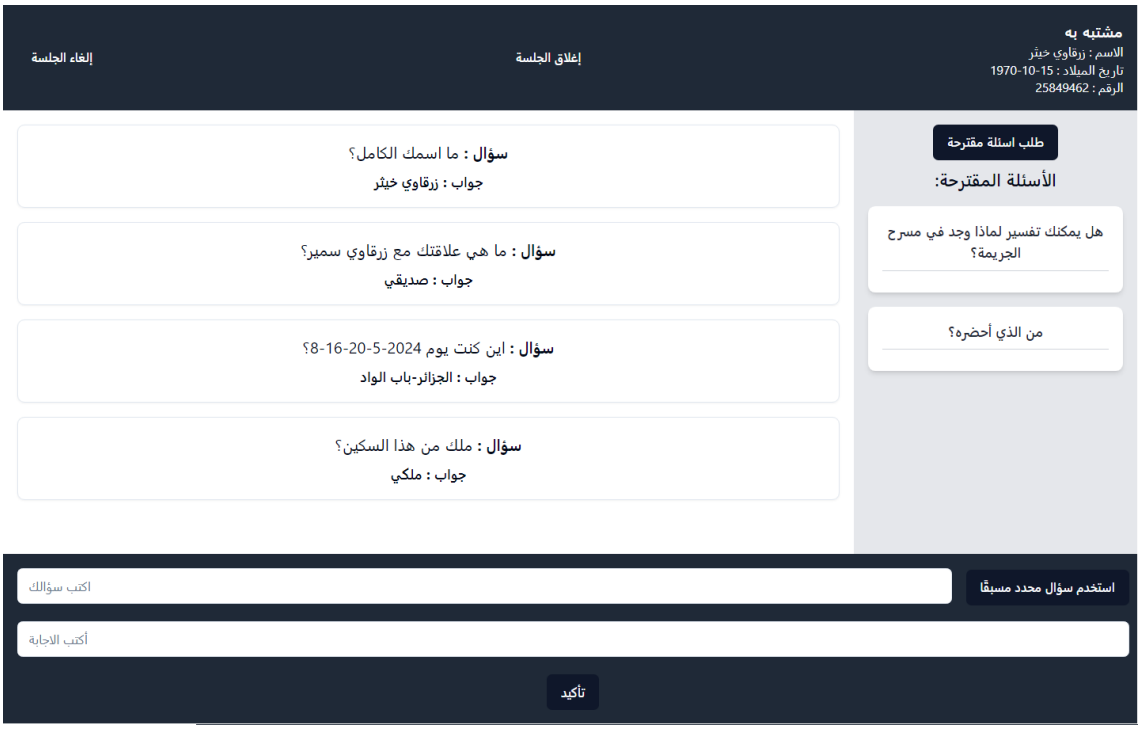

FIGURE 4.13 – recommandation de questions

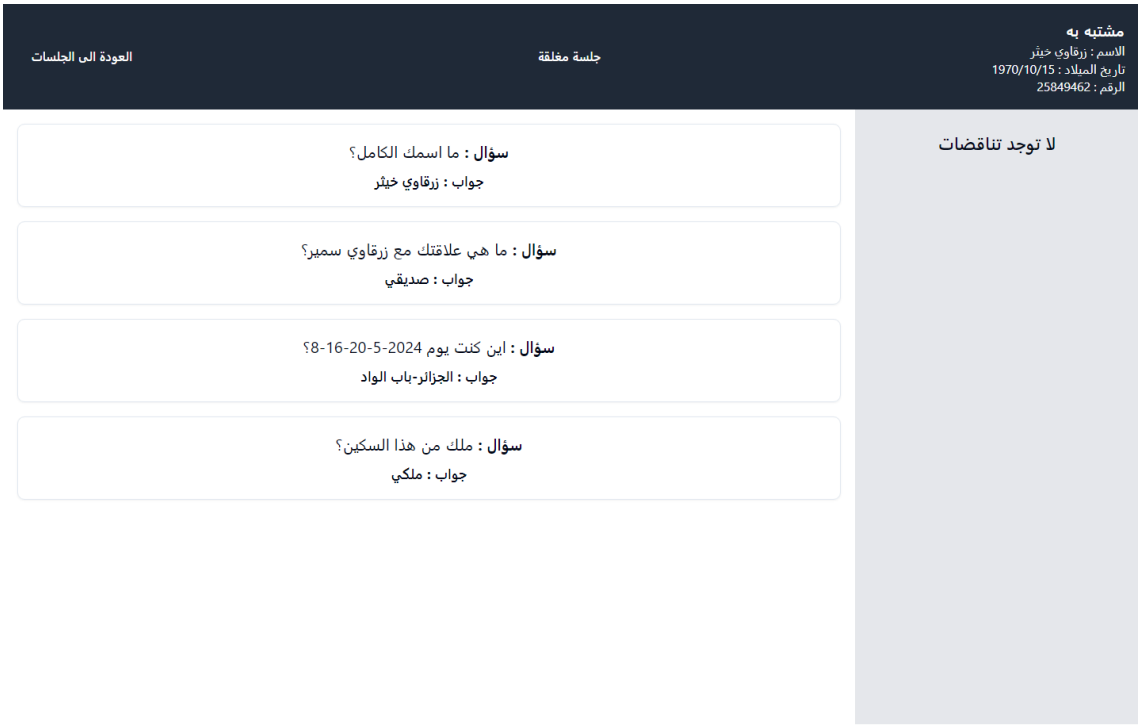

FIGURE 4.14 – Après Clôture de l'audition

FIGURE 4.15 – Saisie d'une deuxième audition

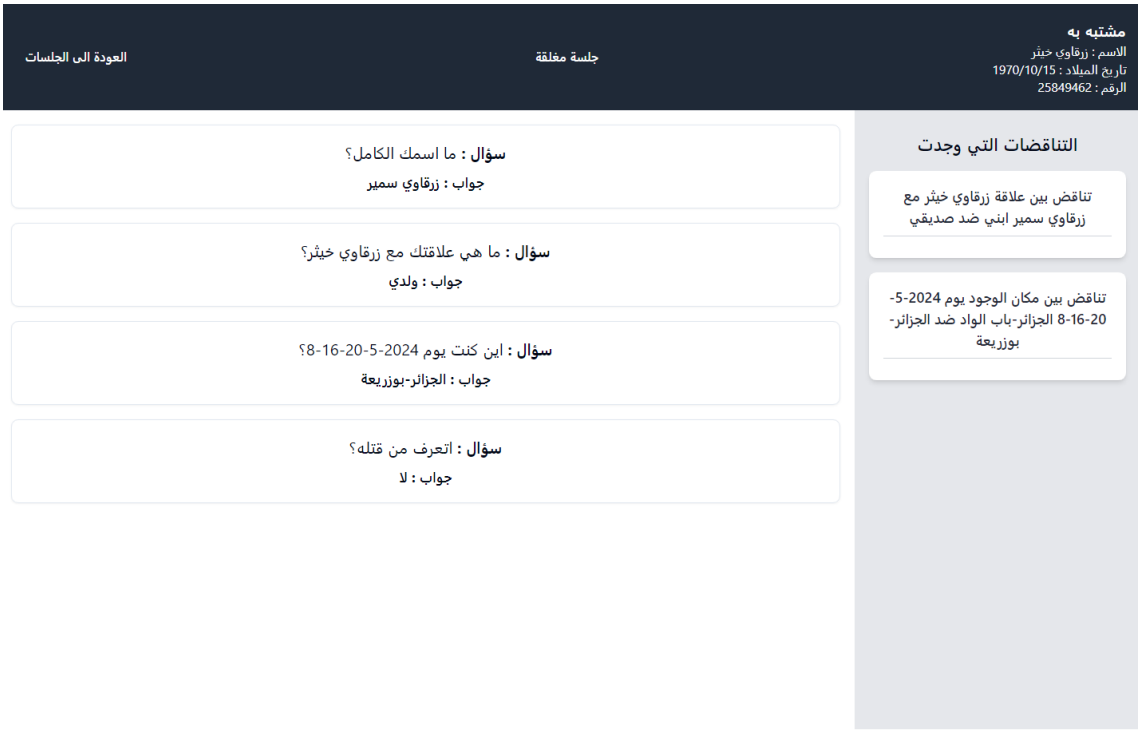

FIGURE 4.16 – Après Clôture de la deuxième audition, détection de contradiction

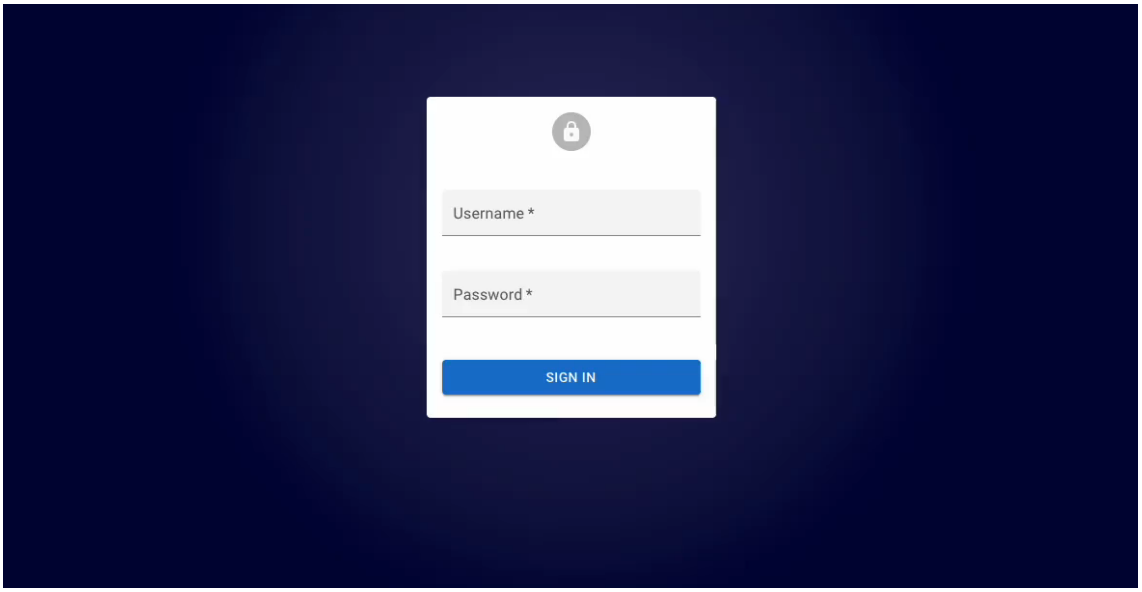

FIGURE 4.17 – Interface d'authentification pour l'admin

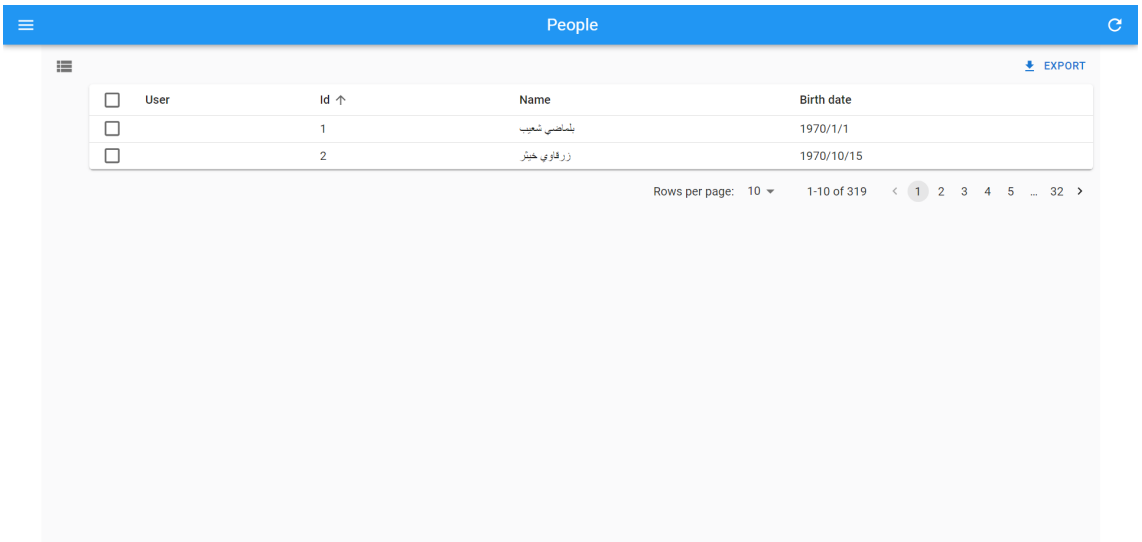

FIGURE 4.18 – Interface de la gestion pour l'admin

Conclusion Générale & travaux futures

Bibliographie

[1] Advancing policing through ai : Insights from the global law enforcement community. https://www.police1.com/iacp/articles/advancing-policing-through-ai-insights-from-the-global-law-enforcement-commun

[2] Greg Friese. Ai in law enforcement : Predictive policing and crime analysis. https://thideai.com/ai-in-law-enforcement-predictive-policing-and-crime-analysis.

[3] Site web de la gendarmerie nationale. https://www.mdn.dz/site_cgn/sommaire/presentation/histoire/historique_fr.php. Consulté en janvier 2024.

[4] Jérémy Robert. Natural language processing (nlp) : Définition et principes. https://datascientest.com/introduction-au-nlp-natural-language-processing.

[5] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57 :345–420, 2016.

[6] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Pearson, Feb. 2024.

[7] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018.

[8] Ayush Thakur, Laxmi Ahuja, Rashmi Vashisth, and Rajbala Simon. Nlp ai speech recognition : An analytical review. In *2023 10th International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1390–1396, 2023.

[9] Venkata Sai Rishita Middi, Middi Raju, and Tanvir Ahmed Harris. Machine translation using natural language processing. *MATEC Web of Conferences*, 277 :02004, 01 2019.

[10] R. Patil, S. Boit, V. Gudivada, and J. Nandigam. A survey of text representation and embedding techniques in nlp. *IEEE Access*, 11 :36120–36146, 2023.

[11] Glossaire du machine learning - one hot encoding. https://developers.google.com/machine-learning/glossary?hl=fr#one-hot-encoding.

[12] Mustafa Abdul Salam. Sentiment analysis of product reviews using bag of words and bag of concepts. *International Journal of Electronics*, 11 :49–60, 2019.

[13] Djoerd Hiemstra. A probabilistic justification for using tf-idf term weighting in information retrieval. *International Journal on Digital Libraries*, 3(2) :131–139, Aug. 2000.

[14] Samir KECHID. Module recherche d'information (ri), chapitre 3 - pondération des termes, 2023-2024. Master 2, Systèmes Informatiques Intelligents.

[15] K. Lund and C. Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, Computers*, 28(2) :203–208, June 1996.

[16] Tomáš Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Workshop Track Proceedings*, Scottsdale, Arizona, USA, May 2-4 2013.

[17] S.J. Johnson, M.R. Murty, and I. Navakanth. A detailed review on word embedding techniques with emphasis on word2vec. *Multimed Tools Appl*, 83 :37979–38007, 2024.

[18] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove : Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, 2014. Association for Computational Linguistics.

[19] Robin M. Schmidt. Recurrent neural networks (rnns) : A gentle introduction and overview. *CoRR*, abs/1912.05911, 2019.

[20] *all-about-recurrent-neural-networks*, 05 2024. `https://medium.com/@jianqiangma/all-about-recurrent-neural-networks-9e5ae2936f6e`.

[21] Ebin Babu Thomas. Understanding lstm : An in-depth look at its architecture, functioning, and pros cons. `https://www.linkedin.com/pulse/understanding-lstm-in-depth-look-its-architecture-pros-babu-thomas/`.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30 : Annual Conference on Neural Information Processing Systems 2017*, pages 5998–6008, Long Beach, CA, USA, December 4-9 2017.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert : Pre-training of deep bidirectional transformers for language understanding. In

Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[24] Nils Reimers and Iryna Gurevych. Sentence-bert : Sentence embeddings using siamese bert-networks, 2019.

[25] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm : Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020.

[26] modèle de transformateur de phrase all-minilm-l6-v2. `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`.

[27] Hridoy Jyoti Mahanta. A study on the approaches of developing a named entity recognition tool. *02(14)*, pages 58–61.

[28] Andrea Gasparetto, Matteo Marcuzzo, Alessandro Zangari, and Andrea Albarelli. A survey on text classification algorithms : From text to predictions. *Inf.*, 13 :83, 2022.

[29] Martin Krallinger, Florian Leitner, Carlos Rodriguez-Penagos, and Alfonso Valencia. Overview of the protein–protein interaction annotation extraction task of biocreative ii. *Genome Biology*, 9(2) :1–19, 2008.

[30] Beth M. Sundheim. Overview of the fourth message understanding evaluation and conference. Technical report, Naval Command Control and Ocean Surveillance Center, RDT & E Division, San Diego, CA, 1992.

[31] Nguyen Bach and Sameer Badaskar. A review of relation extraction. 05 2011.

[32] Kartik Detroja, C.K. Bhensdadia, and Brijesh S. Bhatt. A survey on relation extraction. *Intelligent Systems with Applications*, 19 :200244, 2023.

[33] Sentence similarity. `https://huggingface.co/tasks/sentence-similarity`.

[34] Valentina Dragos. Detection of contradictions by relation matching and uncertainty assessment. *Procedia Computer Science*, 112 :71–80, 2017.

[35] Samuel R. Bowman et al. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[36] Jonas Mueller and Aditya Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[37] Shane Storks, Qiaozi Gao, and Joyce Chai. Recent advances in natural language inference : A survey of benchmarks, resources, and approaches, 11 2019.

[38] Mohamed Boubenia. *Mobile Recommendation System.* PhD thesis, Unknown, 2020.

[39] Oren Sar Shalom, Haggai Roitman, and Pigi Kouki. *Natural Language Processing for Recommender Systems*, pages 447–483. 11 2021.

[40] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations : Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1) :76–80, 2003.

[41] Yue Shi et al. Collaborative filtering beyond the user-item matrix : A survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR)*, 47(1) :3, 2014.

[42] Yu Sun et al. Bert4rec : Sequential recommendation with bidirectional encoder representations from transformers. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019.

[43] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. Content-based recommender systems : State of the art and trends. In *Recommender Systems Handbook*, pages 73–105. Springer, Boston, MA, 2011.

[44] Michael J. Pazzani and Daniel Billsus. Content-based recommendation systems. In *The Adaptive Web*, pages 325–341. Springer, Berlin, Heidelberg, 2007.

[45] Alexandros Karatzoglou et al. Content-based recommendation systems. In *Recommender Systems Handbook*, pages 627–681. Springer, 2015.

[46] Ruisheng Zhang, Qi-dong Liu, Chun-Gui, Jia-Xuan Wei, and Huiyi-Ma. Collaborative filtering for recommender systems. In *2014 Second International Conference on Advanced Cloud and Big Data*, pages 301–308, 2014.

[47] Jie Zou, Yifan Chen, and Evangelos Kanoulas. Towards question-based recommender systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 881–890, New York, NY, USA, 2020. Association for Computing Machinery.

[48] M Yavorsky, RZ Useev, SA Kurushin, et al. Information technologies in law enforcement : Overview of implements and opportunities. *European Proceedings of Social and Behavioural Sciences*, 2021.

[49] Abu Bakr Soliman, Kareem Eisa, and Samhaa R. El-Beltagy. AraVec : A set of arabic word embedding models for use in arabic NLP. In *Proceedings of the 3rd International Conference on Arabic Computational Linguistics (ACLing 2017)*, Dubai, UAE, 2017.

[50] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013.