

Réf : 2019/ II2

Soutenu à la session de Juin 2019

---



**Université de la Manouba  
École Nationale des Sciences de  
l'Informatique**

**Projet de Conception et de Développement**

Réalisé par

**Abir Bel haj Youssef    Bouthaina Ben Guirat**

---

**Sujet : TrustBIOMED : Système de recherche  
d'information biomédicale basé sur le Deep Learning**

---

*Encadré par :*  
**Mme. Anja HABAICHI CHAIBI**

*Organisme :*  
**Laboratoire RIADI/ENSI**



# Signature de l'encadrant

Dr Anja HABACHA CHAIBI

# Remerciements

Tout d'abord, il nous est agréable de remercier toute personne qui nous a donné le courage et la volonté pour achever ce travail.

Particulièrement, nous adressons notre profonde reconnaissance à notre encadrante Dr Anja HABACHA CHAIBI pour son aide, ses conseils et ses explications pertinentes tout au long de la période de la réalisation du projet.

Nous tenons aussi à remercier et à exprimer notre gratitude au doctorant Houssemeddine DERBEL pour sa disponibilité et son partage de connaissances avec nous.

Enfin, nous adressons nos remerciements à nos familles et nos proches pour leur soutien moral et leur sollicitude.

# Table des matières

|  |           |
|--|-----------|
| <b>Introduction générale</b>                                       | <b>8</b>  |
| <b>1 Étude bibliographique</b>                                     | <b>9</b>  |
| 1.1 Recherche d'Information . . . . .                              | 9         |
| 1.1.1 Processus de la Recherche d'Information . . . . .            | 9         |
| 1.1.2 Modèles de la Recherche d'Information . . . . .              | 10        |
| 1.1.3 Évaluation des Systèmes de Recherche d'Information . . . . . | 11        |
| 1.1.4 Recherche d'information biomédicale . . . . .                | 12        |
| 1.2 Word Embedding . . . . .                                       | 13        |
| 1.2.1 Le modèle Latent Semantic Analysis . . . . .                 | 13        |
| 1.2.2 Le modèle Word2Vec . . . . .                                 | 14        |
| 1.2.3 Le modèle FastText . . . . .                                 | 14        |
| 1.3 Réseaux de Neurones . . . . .                                  | 15        |
| 1.3.1 Les réseaux de neurones convolutifs . . . . .                | 15        |
| 1.3.2 Les réseaux de neurones récurrents . . . . .                 | 16        |
| 1.3.3 LSTM . . . . .   | 16        |
| 1.4 Deep Learning . . . . .  | 17        |
| 1.5 Synthèse . . . . .   | 18        |
| <b>2 Analyse et spécification des besoins de TrustBIOMED</b>       | <b>19</b> |
| 2.1 Problématique et objectifs . . . . .                           | 19        |
| 2.2 Spécification des besoins . . . . .                            | 20        |
| 2.2.1 Acteurs . . . . .  | 20        |
| 2.2.2 Besoins fonctionnels . . . . .                               | 20        |
| 2.2.3 Besoins non fonctionnels . . . . .                           | 20        |
| 2.3 Modélisation . . . . .   | 20        |
| 2.3.1 Diagramme des cas d'utilisation . . . . .                    | 21        |
| 2.3.2 Diagramme de séquence système . . . . .                      | 21        |
| 2.4 Conclusion . . . . .   | 23        |

|          |  |           |
|----------|--|-----------|
| <b>3</b> | <b>Conception de TrustBIOMED</b>   | <b>24</b> |
| 3.1      | Conception globale de TrustBIOMED . . . . .                              | 24        |
| 3.1.1    | Architecture pipeline . . . . .  | 24        |
| 3.1.2    | Modèle MVC . . . . .   | 25        |
| 3.2      | Conception détaillée de TrustBIOMED . . . . .                            | 26        |
| 3.2.1    | Couche Modèle . . . . .  | 26        |
| 3.2.2    | Couche Vue . . . . .   | 27        |
| 3.2.3    | Couche Contrôleur . . . . .  | 28        |
| 3.2.4    | Diagramme de séquence : Cas de la recherche d'infor-<br>mation . . . . . | 29        |
| 3.2.5    | Diagramme de séquence : Cas d'ajout de document(s) .                     | 29        |
| 3.3      | Architecture du pré-traitement des données : FastText . . . .            | 30        |
| 3.4      | Architecture du réseau de neurones : Auto-encodeur LSTM . .              | 31        |
| 3.4.1    | Architecture du modèle de similarité : LearnToRank . .                   | 32        |
| 3.5      | Conclusion . . . . .   | 33        |
| <b>4</b> | <b>Réalisation et évaluation du système TrustBIOMED</b>                  | <b>34</b> |
| 4.1      | Environnement de développement . . . . .                                 | 34        |
| 4.1.1    | Environnement matériel . . . . .   | 34        |
| 4.1.2    | Environnement logiciel . . . . .   | 35        |
| 4.1.3    | Justification des choix techniques . . . . .                             | 35        |
| 4.2      | Description de TrustBIOMED . . . . .                                     | 35        |
| 4.2.1    | Aperçu du travail réalisé pour l'espace utilisateur . . .                | 35        |
| 4.2.2    | Aperçu du travail réalisé pour l'espace administrateur .                 | 38        |
| 4.3      | Évaluation de TrustBIOMED . . . . .                                      | 41        |
| 4.3.1    | Exemple du résultat de FastText . . . . .                                | 41        |
| 4.3.2    | Exemple du résultat de l'auto-encodeur . . . . .                         | 42        |
| 4.3.3    | Métrique d'évaluation de TrustBIOMED . . . . .                           | 43        |
| 4.4      | Conclusion . . . . .   | 43        |
|          | <b>Conclusion et perspectives</b>  | <b>45</b> |
|          | <b>Bibliographie</b>   | <b>46</b> |

# Table des figures

|      |  |    |
|------|--|----|
| 1.1  | Processus de la Recherche d'Information [Sauvagnat(2005)]                          | 10 |
| 1.2  | ensemble des documents utilisées pour l'évaluation d'un SRI [Sauvagnat(2005)]      | 11 |
| 1.3  | Architecture des modèles de Word2Vec [Mikolov et al.(2013)Mikolov, Chen, Corrao]   |    |
| 1.4  | Architecture d'un réseau de neurones convolutifs [Nielsen(2015)]                   | 15 |
| 1.5  | Différence entre la structure interne des CNN et LSTM [Stérin(2016)]               | 17 |
| 2.1  | Diagramme des cas d'utilisation  | 21 |
| 2.2  | Diagramme de séquence système pour un simple utilisateur                           | 22 |
| 2.3  | Diagramme de séquence système pour l'administrateur                                | 23 |
| 3.1  | Architecture pipeline du TrustBIOMED   | 25 |
| 3.2  | Diagramme de classes de TrustBIOMED  | 26 |
| 3.3  | Diagramme de classe de la couche Modèle  | 27 |
| 3.4  | Arborescence de l'accueil de TrustBIOMED   | 27 |
| 3.5  | Arborescence de l'espace de l'administrateur                                       | 28 |
| 3.6  | Diagramme de classe de la couche Contrôleur  | 28 |
| 3.7  | Diagramme de séquence pour la recherche d'information                              | 29 |
| 3.8  | Diagramme de séquence pur l'ajout de document(s)                                   | 30 |
| 3.9  | Architecture de FastText   | 30 |
| 3.10 | Structure d'un auto-encodeur [Gensler et al.(2016)Gensler, Henze, Sick, and Raabe] |    |
| 3.11 | LearnToRank  | 32 |
| 4.1  | Page d'accueil de TrustBIOMED  | 36 |
| 4.2  | Page des résultats de la recherche   | 37 |
| 4.3  | Page de l'historique de recherches   | 37 |
| 4.4  | Page d'authentification  | 38 |
| 4.5  | Page de l'espace administrateur  | 39 |
| 4.6  | Page de consultation de la liste des documents                                     | 40 |
| 4.7  | Page d'ajout d'un ou plusieurs documents   | 40 |

|      |  |    |
|------|--|----|
| 4.8  | Choix du chemin adéquat . . . . .                      | 41 |
| 4.9  | Vecteur du mot "AIDS" . . . . .                        | 42 |
| 4.10 | Vecteur du mot "Fever" . . . . .                       | 42 |
| 4.11 | Courbe de la perte pour l'apprentissage des documents  | 42 |
| 4.12 | Courbe de la perte pour l'apprentissage des requêtes . | 43 |

# Introduction générale

Vu le nombre croissant de documents textuels ces dernières années, la recherche biomédicale a généré plusieurs problèmes liés à la difficulté de l'accès à ces données. Alors, la motivation d'avoir des moyens performants et efficaces de recherche d'information est devenu de plus en plus fondamentale. Cette motivation donne naissance à notre système de recherche : TrustBIO-MED.

En effet, la préparation des documents ainsi que les requêtes est l'étape la plus importante. Elle se base sur une représentation vectorielle des données. TrustBioMed effectue un appariement entre la requête saisie par l'utilisateur et les documents déjà préparés pour retourner le classement de ces derniers selon leur pertinence.

Donc pour réaliser notre projet de conception et de développement, nous allons organiser notre rapport en quatre chapitres :

Le premier chapitre intitulé "Étude bibliographique" définit le processus de la recherche d'information ainsi que les principales approches de la représentation d'un corpus textuel. Le deuxième chapitre intitulé "Analyse et spécification des besoins de TrustBIOMED" présente la problématique, l'objectif ainsi que les besoins fonctionnels et non fonctionnels du système. Nous présentons aussi dans ce chapitre les diagrammes de cas d'utilisation et de séquence de TrustBIOMED. Le troisième chapitre est dédié à la "Conception de TrustBIOMED" ainsi que l'étude détaillée du projet à travers la modélisation conceptuelle de la solution proposée. Le dernier chapitre intitulé "Réalisation et évaluation de TrustBIOMED" comporte une présentation de l'environnement de travail accompagné d'une évaluation des performances du système. Enfin, nous clôturons ce rapport avec une conclusion générale qui donnera une synthèse sur le travail effectué ainsi que les perspectives futures.



# Chapitre 1

## Étude bibliographique

La phase d'étude théorique est très importante pour la résolution de tout projet. Au cours de cette phase essentielle, une certaine recherche s'impose dans le but de définir les concepts de base. La section 1.1 de ce chapitre est consacrée à une introduction à la recherche d'information en particulier dans le secteur biomédical. Puis, nous nous intéressons au problème du Word Embedding dans la section 1.2. Nous expliquons ensuite, dans la section 1.3, le modèle de réseau de neurones récurrents et les bases de son apprentissage. Enfin, nous définissons la notion du Deep Learning dans la section 1.4.

### 1.1 Recherche d'Information

La Recherche d'Information (RI) est le fait de trouver des documents pour répondre à un besoin d'information parmi une large collection.

#### 1.1.1 Processus de la Recherche d'Information

Historiquement, les moyens de la recherche d'information sont limités puisqu'elle est liée à la bibliothéconomie et aux données qui sont en croissance de manière exponentielle. Pourtant, l'informatique permet le développement des outils de traitement de l'information en établissant la représentation des documents de corpus et leur indexation. L'objectif fondamental du processus RI, illustré par la Figure 1.1, est de faciliter l'accès aux données. Il permet en fait de relier les mots clés figurant dans la requête avec l'ensemble des documents de corpus et puis d'ordonner les documents.

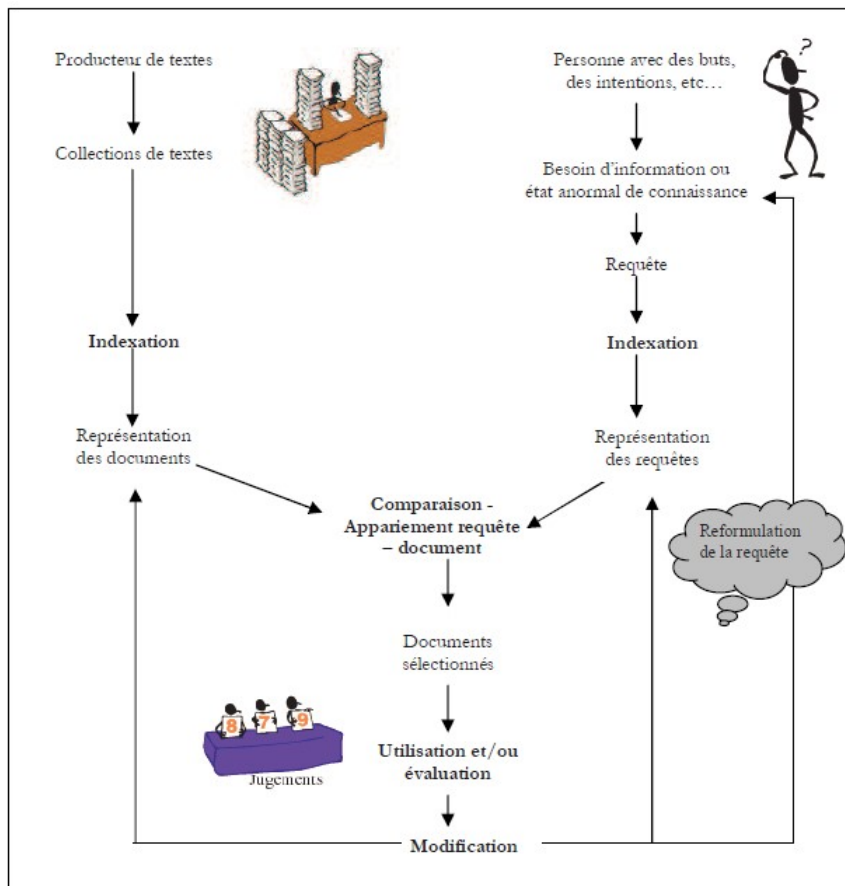


FIGURE 1.1 – Processus de la Recherche d’Information [Sauvagnat(2005)]

## 1.1.2 Modèles de la Recherche d’Information

La RI a développé différentes approches pour bien représenter des textes. Les techniques basiques de recherche d’information sont les modèles booléen, vectoriel et probabiliste.

### 1.1.2.1 Le modèle booléen

C’est un modèle classique de recherche d’informations utilisé par plusieurs SRI jusqu’à ce jour. Ce modèle est basé sur la théorie des ensembles car il modélise le corpus et la requête comme un ensemble de terme. La similarité document-requête est incluse dans l’ensemble  $0,1$  vue que la récupération des documents dépend du fait que ces derniers contiennent les mots clés de la requête ou non.

### 1.1.2.2 Le modèle vectoriel

Ce modèle s'appuie sur la représentation des documents et des requêtes dans l'espace vectoriel des termes de l'index afin de déterminer lequel des documents doit être récupéré. En effet, si un terme figure dans le document, sa valeur dans le vecteur prend un entier non nul.

### 1.1.2.3 Le modèle probabiliste

C'est un modèle classique qui repose sur la distribution des probabilités des termes. Le principe de ce modèle est d'estimer la probabilité  $P$  de trouver si un document  $D$  est pertinent pour une requête  $R$  ou non. La pertinence du document est estimée selon l'équation (1.1) :

$$s(D|R) = \frac{P(R|D)}{P(\bar{R}|D)} \quad (1.1)$$

## 1.1.3 Évaluation des Systèmes de Recherche d'Information

L'évaluation des SRI permet de vérifier la pertinence des documents en fonction des requêtes émises. En fait, il existe plusieurs mesures d'évaluation qui ont pour but d'identifier la capacité à retourner le maximum des documents convenables pour chaque requête du SRI. D'ailleurs, la Figure 1.2 indique l'ensemble des documents utilisées pour l'évaluation d'un SRI.

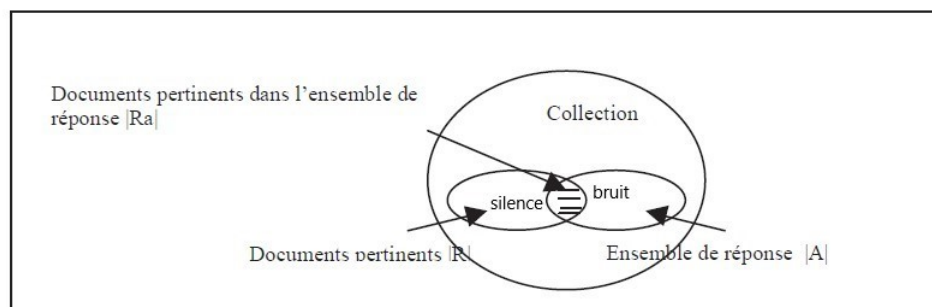


FIGURE 1.2 – ensemble des documents utilisées pour l'évaluation d'un SRI [Sauvagnat(2005)]

#### **1.1.3.1 Le Silence**

C'est l'ensemble des documents pertinents faisant partie du corpus mais non récupérés par le SRI lors de la recherche d'une information.

#### **1.1.3.2 Le Bruit**

C'est l'ensemble des documents non pertinents récupérés par le SRI en réponse à une requête.

#### **1.1.3.3 Le Rappel**

Il mesure l'efficacité du SRI et sa capacité à minimiser le silence. Cette métrique est estimée par le calcul du ratio du nombre des documents pertinents trouvés par le système et le nombre des documents pertinents total pour une requête donnée.

#### **1.1.3.4 La Précision**

C'est une métrique pour la mesure de l'efficacité du SRI et sa capacité à minimiser le bruit. Elle est établie à partir du ratio entre le nombre de documents pertinents retournés par le système et le nombre de documents trouvés pour la requête en question.

#### **1.1.3.5 La F-mesure**

Elle combine les mesures de rappel et de précision et estime à quel point le système minimise le silence et le bruit qui est l'objectif fondamental d'un SRI. Autrement dit, cette métrique mesure la capacité du système à récupérer le maximum des documents pertinents par l'équation (1.2).

$$F = 2 \cdot \frac{(\text{précision} \cdot \text{rappel})}{(\text{précision} + \text{rappel})} \quad (1.2)$$

### **1.1.4 Recherche d'information biomédicale**

La recherche d'informations biomédicales joue un rôle de plus en plus important pour avoir un meilleur accès aux connaissances et à faciliter la prise de décision. Elle exige un traitement des synonymes et des acronymes des mots et certaines abréviations pratiquées dans le domaine biomédical. Par exemple : maladie de Kahler est associée à myélome multiple, et SLA est pour Sclérose Latérale Amyotrophique.

La machine ne peut pas accepter les textes directement, nous avons besoin d'une représentation numérique de telle sorte que les algorithmes peuvent traiter les données.

## 1.2 Word Embedding

Le Word Embedding est la représentation vectorielle des mots en des vecteurs de réels dans un espace vectoriel de faible dimension. Il peut être utilisé pour calculer les similarités entre les mots, créer des groupes de mots associés, classifier des textes en des sujets, regrouper des documents, et enfin traiter le langage naturel. Le calcul de la similarité sémantique entre les mots est l'un des défis clés dans de nombreuses applications basées sur la langue. Auparavant, nous utilisons la représentation one-hot en créant un vecteur rempli de zéros de taille correspondante au nombre de mots dans le vocabulaire et ensuite en mettant un seul 1 à la position associée au mot. Mais, ce vecteur est de dimension élevée. L'idée est qu'au lieu d'utiliser cette représentation fixe qui ne fait aucune hypothèse de similitude des mots, nous essayons d'apprendre des représentations vectorielles des mots directement.

Pour bien représenter les mots, il faut que les vecteurs portent une certaine signification. Par exemple, les mots liés doivent avoir des vecteurs très proches et les différences entre les vecteurs doivent aussi porter un sens. Le sens d'un mot peut être approximé par l'ensemble des contextes dans lequel il se produit. Les systèmes de Word Embedding font cette signification vectorielle si bien que nous pouvons prendre la représentation vectorielle du mot Roi, soustraire celle du mot Homme, et ajouter celle du mot Femme, pour obtenir le mot Reine.

Plusieurs techniques du Word Embedding attirent l'attention ces dernières années, citons l'Analyse Sémantique Latente, le Word2vec et le FastText.

### 1.2.1 Le modèle Latent Semantic Analysis

Le principe de l'analyse sémantique latente (LSA) consiste à construire une matrice d'occurrence de chaque mot dans chaque document à partir d'un corpus textuel. C'est une matrice rectangulaire dont les lignes sont les termes et les colonnes sont les documents du corpus. La décomposition en valeurs singulières est une propriété des matrices rectangulaires. En effet, la matrice originale de l'apparition d'un mot dans un document est écrite sous la forme du produit de deux matrices orthogonales et une matrice diagonale.

### 1.2.2 Le modèle Word2Vec

Word2vec est développé par Tomas Mikolov et son équipe chez Google en 2013 [Cousyn(2018)] et utilise l'architecture d'un réseau de neurones. Grâce à cette méthode, les ordinateurs peuvent comprendre, non seulement une définition d'un mot et sa signification, mais aussi son contexte, puisqu'elle donne une représentation vectorielle à un mot en extrayant ses caractéristiques dans son contexte à l'intérieur d'un corpus de texte. Les représentations vectorielles des mots par les modèles Word2Vec sont utiles dans diverses tâches du traitement de langage naturel. D'ailleurs, deux modèles sont définis, le Modèle skip-gramme et le modèle Continuous Bag Of Words, illustrés à la Figure 1.3. Ils restent un choix populaire pour représenter les textes en raison de leur efficacité et de leur simplicité.

La Figure 1.3 a) représente la variante CBOW où le réseau utilise les mots de contexte, par exemple les cinq mots qui précèdent et les cinq mots qui suivent ce mot, pour prédire le mot cible. Alors que la Figure 1.3 b) représente la variante Skip-gram où le réseau utilise le mot cible pour obtenir son contexte.

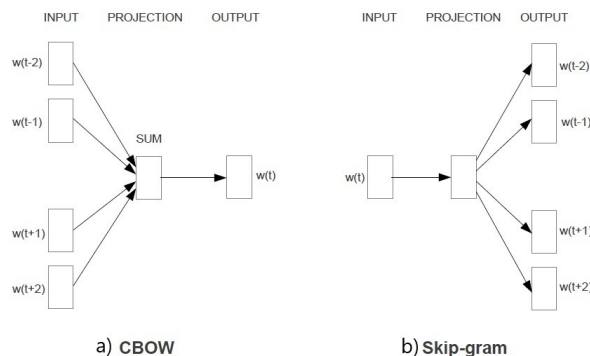


FIGURE 1.3 – **Architecture des modèles de Word2Vec** [Mikolov et al.(2013)Mikolov, Chen, Corrado, and Dean]

### 1.2.3 Le modèle FastText

Il existe plusieurs autres modèles de Word Embedding mais nous sommes intéressés par le modèle FastText qui est une bibliothèque open source, gratuite de Facebook. Ce modèle est efficace pour le traitement du langage naturel, la représentation vectorielle des mots et surtout la classification des textes.

Son idée principale est de déterminer le contexte de chaque mot en exploitant la morphologie, de façon que les mots ayant les mêmes radicaux ne sont pas représentés avec différents paramètres. Le modèle FastText est plus performant que le modèle Word2Vec pour les mots interchangeables et riches en morphologie. Mais, il considère plusieurs autres vecteurs. Par exemple : le mot going est composé en 6 vecteurs : go + oi + in + ng + goi + ing.

## 1.3 Réseaux de Neurones

Les réseaux de neurones artificiels tirent leur inspiration du neurone biologique qui constitue l'unité fonctionnelle de base du système nerveux dont la transmission de l'information est sa principale fonction. Il existe des dizaines de différents types de réseaux de neurones. Chaque modèle a sa particularité, certains apprennent plus facilement à traiter des images, d'autres, à mémoire, peuvent traiter des séquences de mots ou d'images.

### 1.3.1 Les réseaux de neurones convolutifs

Les réseaux de neurones convolutifs (CNN) est un ensemble de neurones organisés en couches et reliés entre eux par des connexions appelées des poids. Chacun de ces poids est une valeur qui nous sert à propager nos informations. Ils sont à la base du fonctionnement de notre réseau car c'est à partir d'eux que tout l'apprentissage peut se faire. Comme le montre la Figure 1.4, la première couche de neurones d'entrées s'appelle input layer et celle de sortie est le output layer. Toutes les couches qui se trouvent entre les deux sont des hidden layers, leur nombre diffère selon l'application.

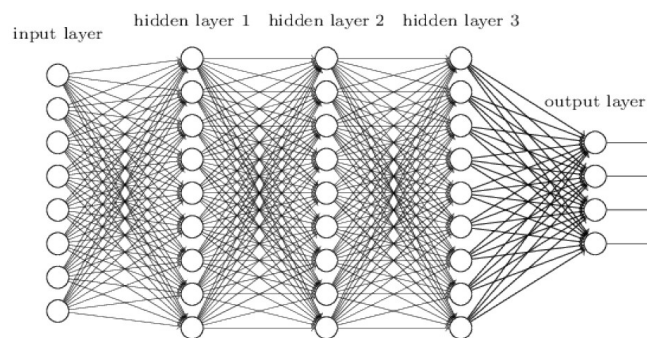


FIGURE 1.4 – Architecture d'un réseau de neurones convolutifs [Nielsen(2015)]

### 1.3.2 Les réseaux de neurones récurrents

L'idée fondamentale des réseaux récurrents (RNN) est d'introduire dans le réseau de neurones des synapses qui bouclent afin de récupérer l'information des mots précédents pour mieux comprendre le mot présent. D'un point de vue d'architecture du réseau de neurones, les signaux ne vont plus nécessairement dans un seul sens, certains signaux reviennent vers l'arrière du réseau de neurones.

Tous les RNN ont des boucles dans la couche récurrente. Cela leur permet de conserver les informations en mémoire. Cependant, il est difficile d'utiliser les RNN standards pour la résolution de problèmes nécessitant l'apprentissage de dépendances temporelles à long terme. Cela est dû au fait que le gradient de la fonction de perte décroît de manière exponentielle avec le temps. Pour éviter ce problème, les informaticiens Sepp Hochreiter et Jürgen Schmidhuber ont introduit en 1997 l'architecture des réseaux à mémoire de long terme (LSTM) [Bourgeade et al.(2018)Bourgeade, Muller, and Serrurier].

### 1.3.3 LSTM

Grâce à une cellule de mémoire capable de conserver les informations en mémoire pendant de longues périodes, les réseaux LSTM sont un type spécial de RNN capable d'apprendre les dépendances à long terme en utilisant des unités spéciales en plus des unités standards.

La première étape de fonctionnement du LSTM consiste à identifier les informations qui ne sont pas nécessaires et qui sont rejetées de l'état de la cellule. Cette décision est prise par une couche sigmoïde appelée couche d'oublis. La prochaine étape consiste à décider quelles nouvelles informations stockées dans l'état de la cellule. Une couche sigmoïde appelée couche d'entrée détermine les valeurs à mettre à jour. Ensuite, une couche tanh crée un vecteur de nouvelles valeurs candidates pouvant être ajoutées à l'état. La dernière étape est de mettre à jour l'ancien état de la cellule dans le nouvel état de la cellule et d'exécuter une couche sigmoïde qui décide quelles parties de l'état de la cellule à afficher.

La Figure 1.5 montre la différence entre la structure interne des réseaux de neurones convolutifs et ceux à mémoire de long terme.



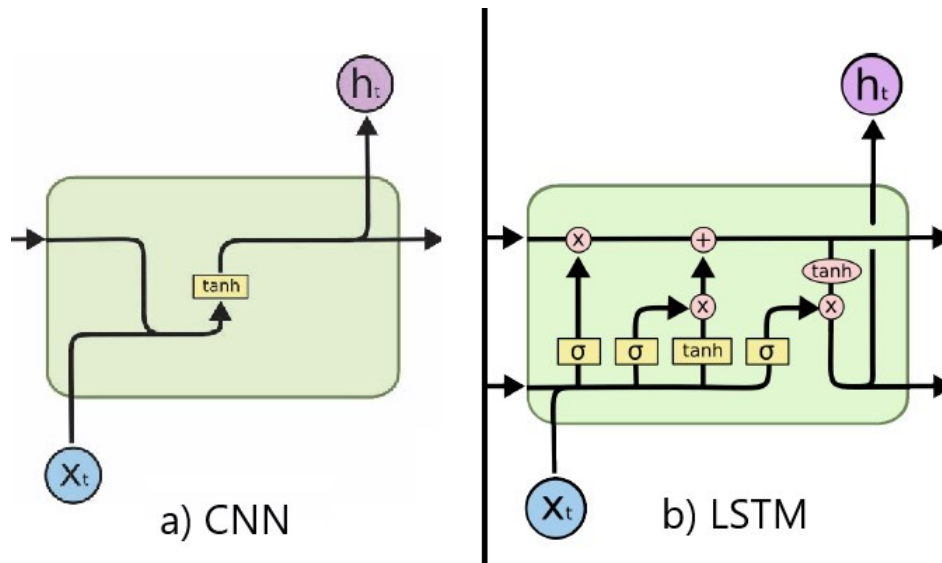


FIGURE 1.5 – Différence entre la structure interne des CNN et LSTM [Stérin(2016)]

## 1.4 Deep Learning

Nous parlons de Deep Learning pour les réseaux de neurones ayant au moins deux hidden layers pour être capable de traiter des problèmes complexes. En effet, les premières apparitions du Deep Learning datent des années 80, et pourtant nous en entendons beaucoup parler que depuis quelques années principalement pour deux raisons. En premier lieu, la capacité de calcul de nos ordinateurs qui montent en flèche permet de gérer des modèles bien plus riches en ressources et donc d'avoir des résultats plus intéressants et plus rapidement. La deuxième principale raison est la quantité de données.

Ce domaine est un type particulier du Machine Learning qui a conduit au concept de réseau de neurones artificiels. La différence la plus importante entre le Deep Learning et le Machine Learning est sa performance lorsque le volume des données augmente et sa forte dépendance des machines haut de gamme. Le modèle basé sur le Deep Learning structure les données énormes en couches pour créer un réseau de neurones artificiels capable d'apprendre et de prendre des décisions intelligentes par ses propres moyens.

## 1.5 Synthèse

Dans ce chapitre, nous avons consacré une partie pour décrire le processus de Recherche d'Information en particulier dans le domaine biomédical. Puis, nous avons expliqué les différents modèles du Word Embedding comme l'Analyse Sémantique Latente, le Word2Vec et le FastText. Enfin, nous avons défini la notion du Deep Learning après avoir détailler l'architecture des réseaux de neurones récurrents particulièrement, ceux à mémoire de long terme. Dans le chapitre suivant, nous nous intéressons aux objectifs d'un système de recherche d'information biomédicale basé sur le Deep Learning, qui nous mènent à l'analyse des besoins fonctionnels.

# Chapitre 2

## Analyse et spécification des besoins de TrustBIOMED

Ce chapitre est consacré à l'analyse et la spécification des besoins qui est une phase primordiale pour la conception et l'implémentation du système. Nous commençons par la formulation de la problématique et la définition des objectifs dans la section 2.1. Ensuite, la section 2.2 présente le style architectural général du TrustBIOMED. Puis, nous introduisons les besoins fonctionnels, les besoins non fonctionnels dans la section 2.3. Enfin, nous exposons les diagrammes d'analyse dans la section 2.4 : diagramme de cas d'utilisation (section 2.4.1) et diagramme de séquence système (section 2.4.2).

### 2.1 Problématique et objectifs

L'obtention des informations qui correspondent aux besoins de l'utilisateur devient de plus en plus difficile vu la croissance de la littérature surtout biomédicale. Donc trouver un système de recherche d'information biomédicale qui répond aux besoins des utilisateurs avec un taux de précision important demeure une nécessité pour les chercheurs. Nous nommons notre système TrustBIOMED. En effet, grâce à sa précision et sa performance, le chercheur peut lui donner confiance et croire aux résultats biomédicales qu'il procure. D'où l'idée de ce nom.

C'est ainsi que l'objectif principal de TrustBIOMED est la recherche et la récupération des documents biomédicaux pertinents suite à des requêtes émises par les utilisateurs.

## 2.2 Spécification des besoins

L'expression des besoins consiste à définir les différents besoins fonctionnels qui décrivent les fonctionnalités ou les services que peut offrir le système. Elle contient aussi les besoins non fonctionnels qui sont des contraintes à respecter pour tout le système.

### 2.2.1 Acteurs

Les acteurs sont le groupe de chercheurs d'informations biomédicales qui interagissent avec le système. En effet notre système comporte :

- L'utilisateur : ne peut accéder qu'aux services qui ne nécessitent pas une authentification.
- L'administrateur : c'est l'acteur qui gère les documents du système et possède le droit d'accès.

### 2.2.2 Besoins fonctionnels

Les principaux besoins fonctionnels de notre système sont :

- Saisir la requête par l'utilisateur.
- Chercher l'information adéquate avec la requête, c'est-à-dire la maladie associée aux symptômes précisés.
- Indexer les documents en les affectant des scores.
- Classer les documents selon leurs pertinences.

### 2.2.3 Besoins non fonctionnels

Les principaux besoins non fonctionnels de notre système sont :

- La fiabilité : le système doit assurer la recherche attendue sans erreur.
- L'ergonomie : le chercheur doit trouver le système agréable et facile à utiliser.
- La réutilisabilité : les composantes du système peuvent être réutilisables dans de nouvelles applications.

## 2.3 Modélisation

Notre modélisation se base sur deux diagrammes d'analyse : le diagramme de cas d'utilisation et le diagramme séquence système.

### 2.3.1 Diagramme des cas d'utilisation

Le diagramme des cas d'utilisation (DCU) illustré par la Figure 2.1, représente les fonctionnalités nécessaires aux acteurs : chercher l'information biomédicale, consulter l'historique de la recherche, et ajouter des documents. De plus, il existe une relation de généralisation entre l'administrateur et l'utilisateur.

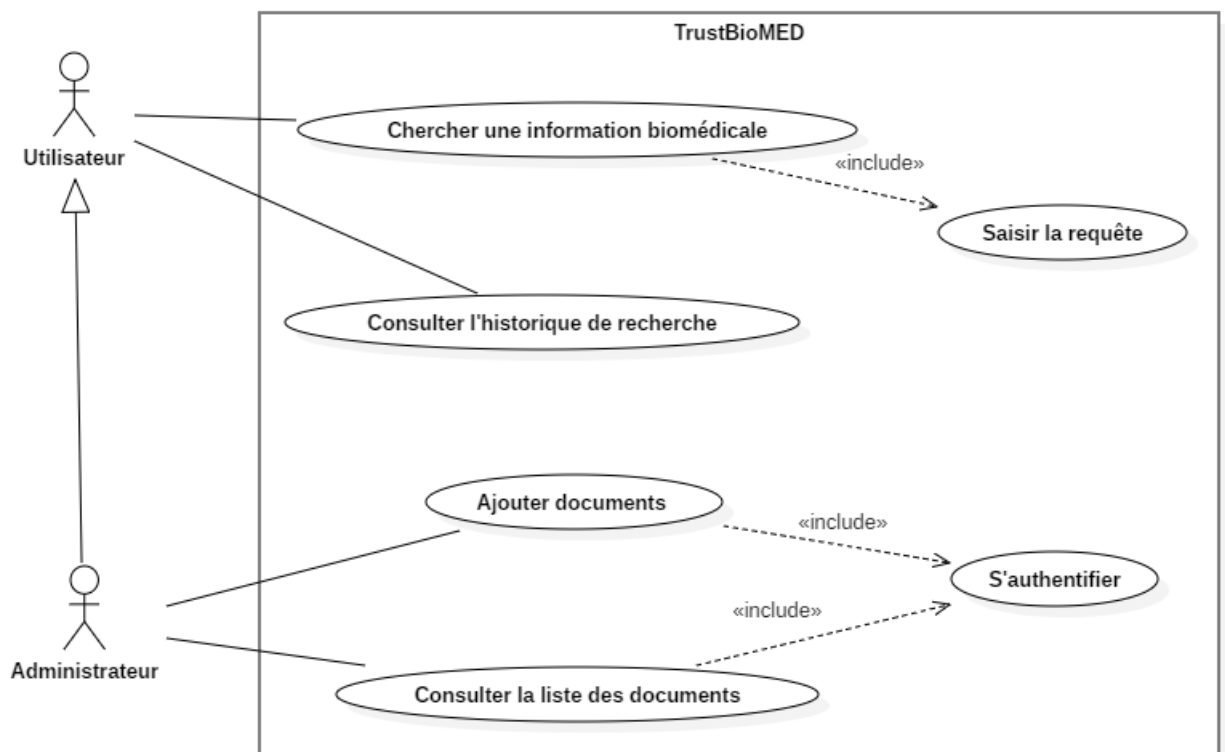


FIGURE 2.1 – Diagramme des cas d'utilisation

### 2.3.2 Diagramme de séquence système

Le diagramme de séquence système présenté par la Figure 2.2 décrit les interactions entre un simple chercheur d'information et TrustBIOMED selon un point de vue temporel (un scénario lié au cas d'utilisation).

En effet, il peut soit consulter l'historique de ses recherches, soit saisir une requête que TrustBIOMED doit indexer, effectuer l'appariement requête-document, classer les documents selon leurs pertinences, pour enfin afficher le résultat.

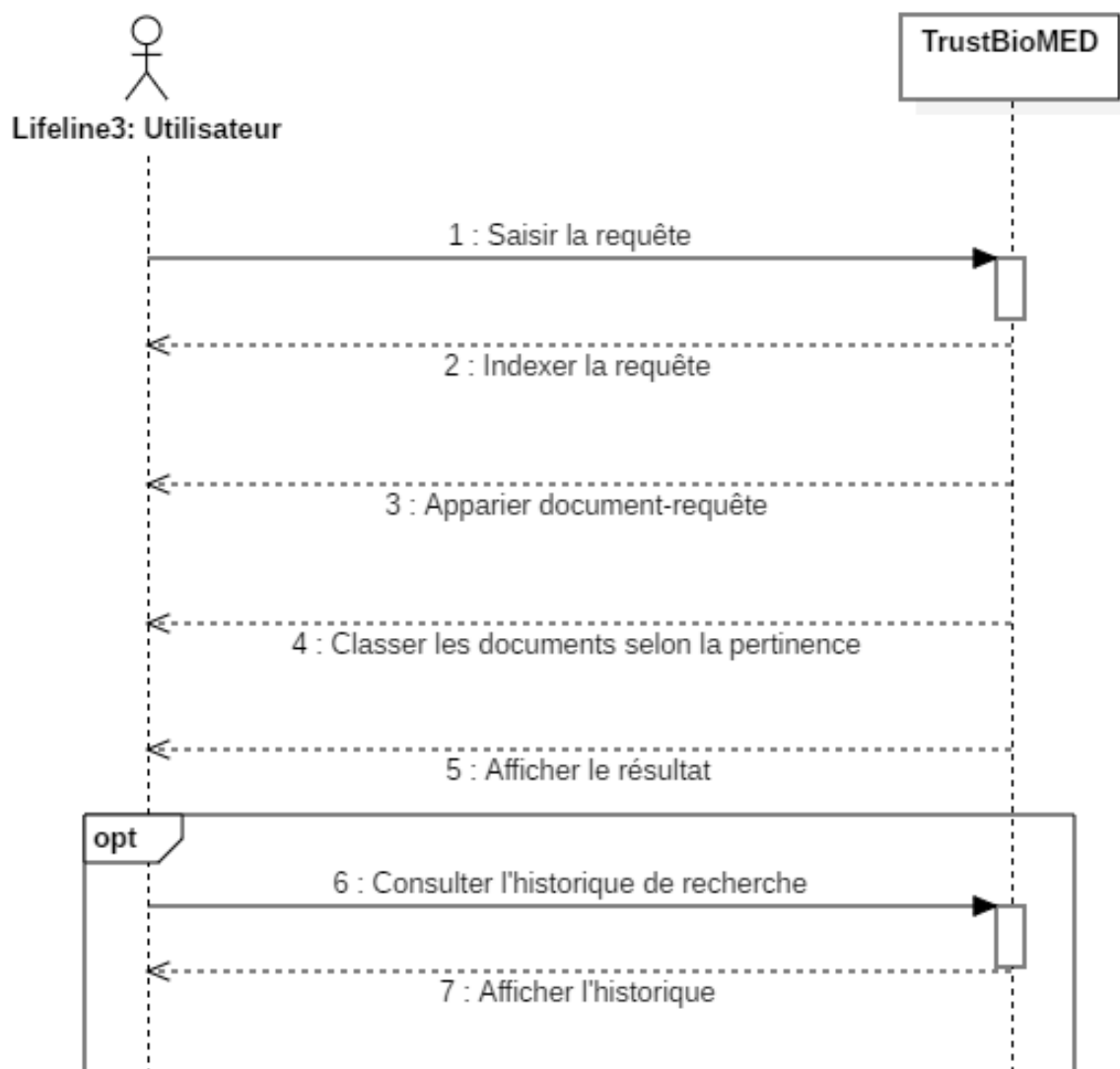


FIGURE 2.2 – Diagramme de séquence système pour un simple utilisateur

La Figure 2.3 illustre les interactions entre un administrateur et TrustBIOMED. En effet, il peut s'authentifier et ajouter des documents à TrustBIOMED.

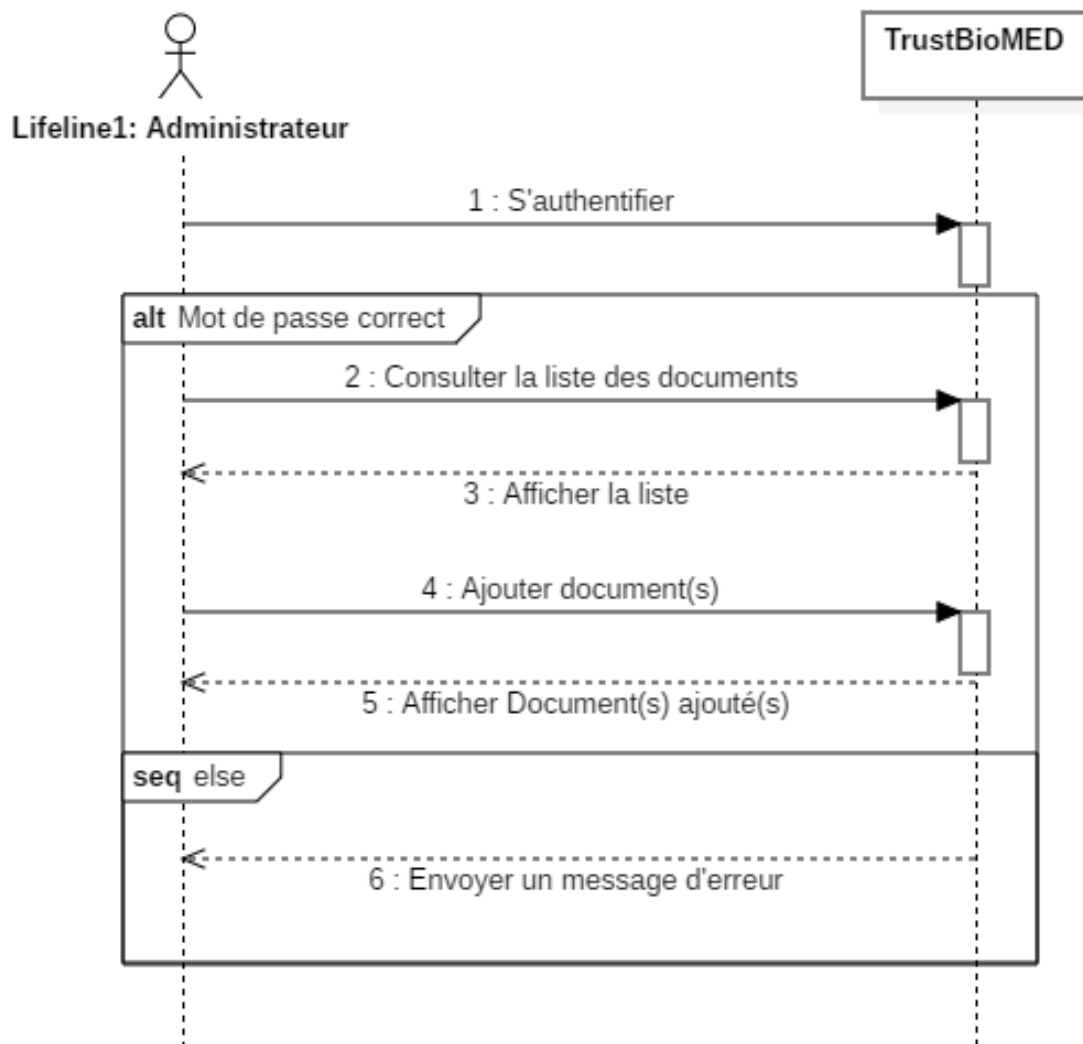


FIGURE 2.3 – Diagramme de séquence système pour l'administrateur

## 2.4 Conclusion

Tout au long de ce chapitre nous avons analysé les besoins fonctionnels et non fonctionnels du TrustBIOMED, ainsi que les principaux cas d'utilisation. Dans le chapitre suivant, nous présentons l'architecture de TrustBIOMED et de présenter une description détaillée des modules, interface et données figurant dans notre système.

## Chapitre 3

# Conception de TrustBIOMED

Après avoir tracé les grandes lignes à travers la phase de l'analyse des besoins, mettons l'accent maintenant sur une phase fondamentale dans le cycle de vie d'un système : la phase de conception. Cette phase est majeure pour la réalisation du projet.

Ce chapitre est consacrée à concevoir l'architecture du réseau de neurones de TrustBIOMED dans la section 3.1. Nous nous intéressons ensuite, d'une part, à la conception globale de TrustBIOMED dans la section 3.2, en détaillant son architecture physique pipeline et son modèle logique MVC. D'autre part, nous expliquons la conception détaillée dans la section 3.3, en illustrant les diagrammes de classe et de séquence

### 3.1 Conception globale de TrustBIOMED

Nous nous intéressons maintenant aux architectures matérielle et logicielle que nous avons choisies pour notre système.

#### 3.1.1 Architecture pipeline

La Figure 3.1 présente l'architecture pipeline du système. En effet, la sortie de chaque phase est l'entrée de la suivante.

D'une part, nous commençons par l'extraction des données textuelles suivie de l'étape d'indexation. D'autre part, la requête saisie par l'utilisateur est aussi indexée. La phase suivante est l'appariement requête-document. Finalement, les documents sont classés selon leurs pertinences par rapport à la requête.



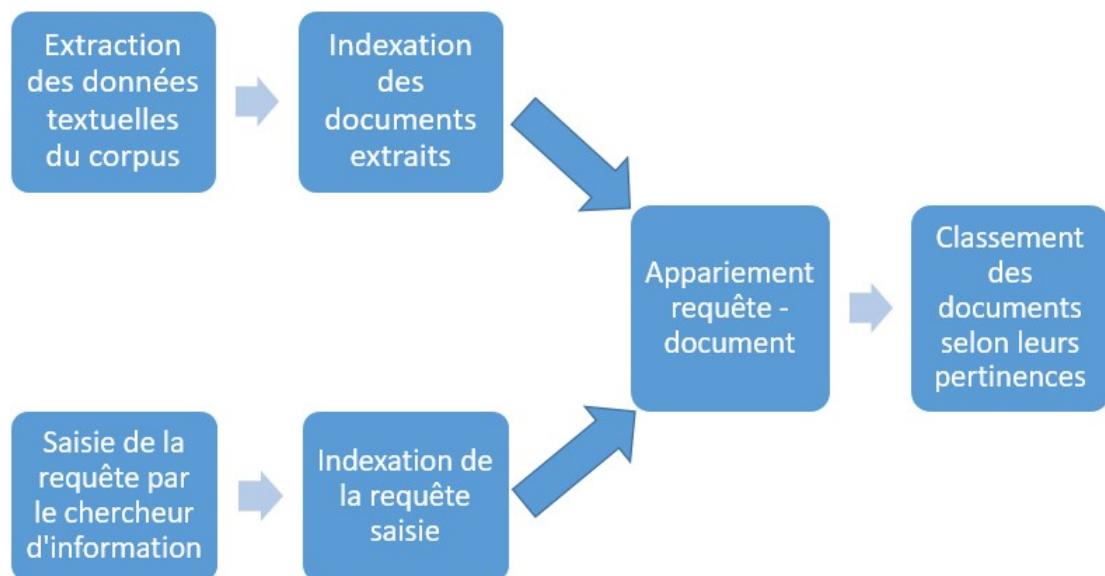


FIGURE 3.1 – Architecture pipeline du TrustBIOMED

### 3.1.2 Modèle MVC

L'architecture MVC (Modèle, Vue et Contrôleur) est celle que nous avons choisie dans la réalisation de TrustBIOMED pour séparer les données (modèle), l'affichage (vue) et les actions (contrôleur). Le paradigme Model-View-Controller repose sur le fait que l'interface utilisateur graphique doit être séparée de la logique de l'application, avec des liens très bien définis entre les deux. Cela facilite beaucoup la modification de la logique du modèle sans avoir à changer de code relatif à l'interface utilisateur. De même, le code d'édition relatif au fonctionnement de l'interface utilisateur en réponse aux entrées de l'utilisateur ne doit pas influencer le code contrôlant la logique du modèle.

- Le Modèle représente le cœur de TrustBIOMED : traitement et manipulation des données.

- La Vue représente l'interface avec quoi l'utilisateur interagit. Sa première tâche est de présenter les résultats renvoyés par le modèle, sa seconde tâche est de recevoir toute action de l'utilisateur : clic de souris et entrée de texte, qui sont envoyés au contrôleur.

- Le Contrôleur gère la synchronisation entre la Vue et le Modèle. Il réagit aux actions de l'utilisateur en effectuant les actions nécessaires sur le Modèle, surveille les modifications du modèle et informe la Vue des mises à jour nécessaires.

## 3.2 Conception détaillée de TrustBIOMED

Dans cette section, nous présentons le diagramme de classes de TrustBIOMED. En effet, les classes spécifient la structure et le comportement d'un ensemble d'objets de même nature. Nous présentons dans la Figure 3.2 la structure statique du système, en termes de classes et de relations entre ces classes.

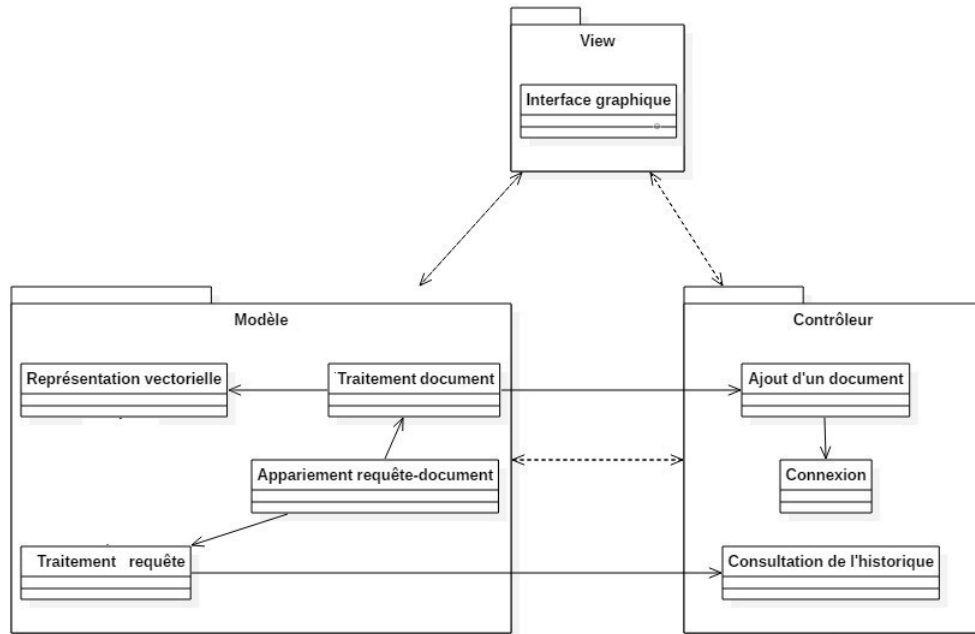


FIGURE 3.2 – Diagramme de classes de TrustBIOMED

Au cours de cette étape, nous décrivons les classes des différentes couches.

### 3.2.1 Couche Modèle

La couche Modèle contient une classe "Représentation vectorielle" qui prend en entrée le document ou la requête initiale et retourne leurs représentations vectorielles correspondantes en utilisant le FastText.

Elle contient trois autres classes "Traitement document" et "Traitement requête" responsables à l'apprentissage du document et de la requête en utilisant LSTM, et une classe "Appariement requête-document" qui calcule la similarité entre le document et la requête en utilisant LearnToRank. Ces différentes classes de la couche Modèle sont présentées dans la Figure 3.3.

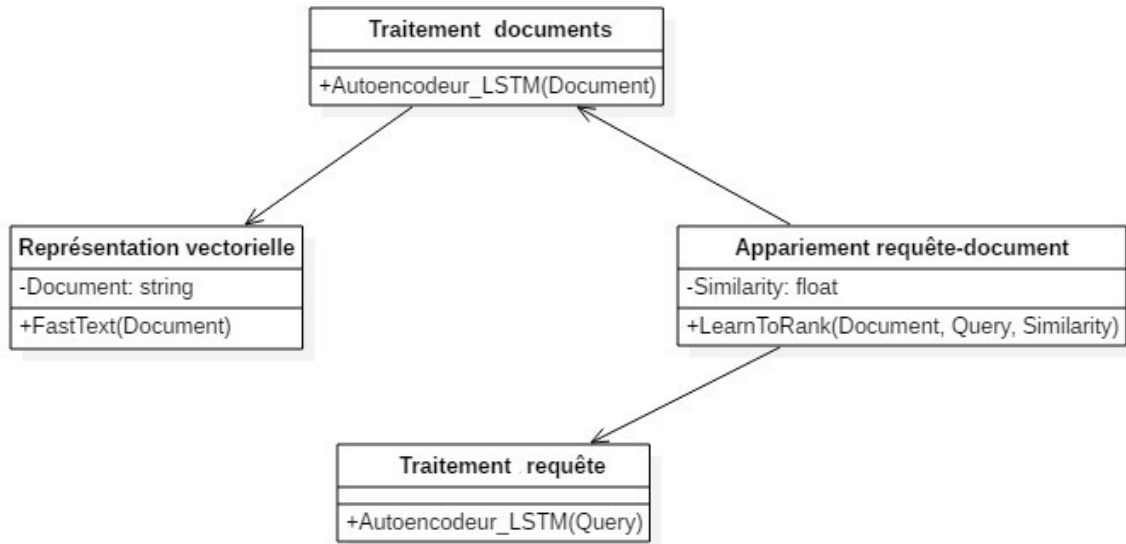


FIGURE 3.3 – Diagramme de classe de la couche Modèle

### 3.2.2 Couche Vue

La couche Vue contient une seule classe d'interface graphique. Nous commençons par illustrer l'arborescence de l'accueil de TrustBIOMED dans la Figure 3.4. Il contient la zone de recherche, la bouton de la consultation de l'historique et celle de la connexion.

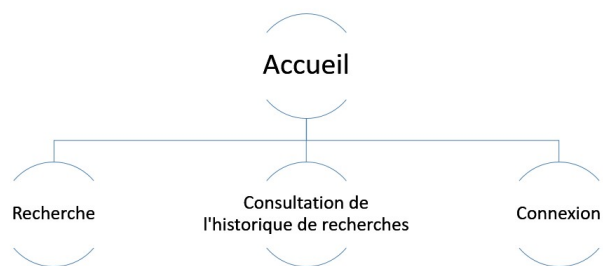


FIGURE 3.4 – Arborescence de l'accueil de TrustBIOMED

L'arborescence de l'espace de l'administrateur après sa connexion est illustrée dans la Figure 3.5. En plus des services offerts à l'utilisateur, il gère les documents en consultant la liste et en ajoutant un ou plusieurs documents.

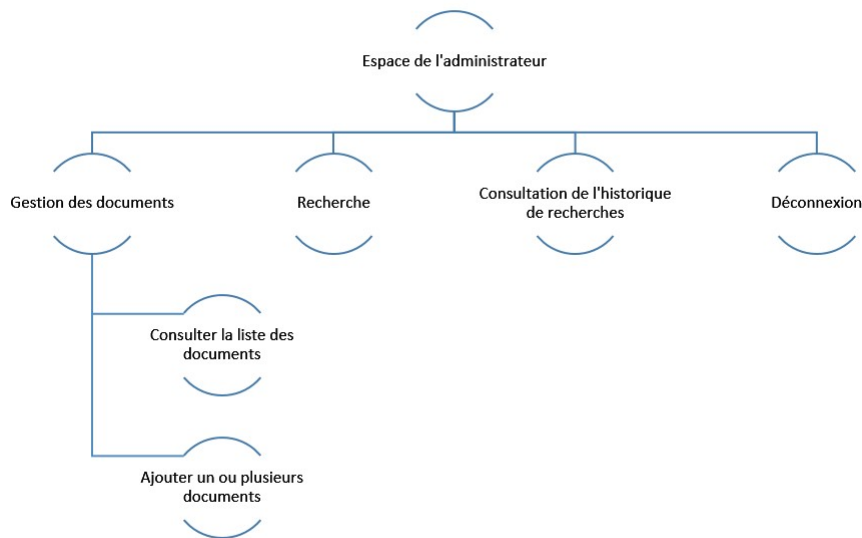


FIGURE 3.5 – Arborescence de l'espace de l'administrateur

### 3.2.3 Couche Contrôleur

La couche contrôleur contient une classe de "Authentification" pour qu'il puisse ajouter, par la classe "Ajout d'un document", un nouveau document afin d'enrichir le corpus. Une autre classe dans cette couche est la classe "Consultation de l'historique". Elle aide le chercheur d'information biomédicale à consulter son historique de recherche. La Figure 3.6 illustre le diagramme de classe de la couche Contrôleur.

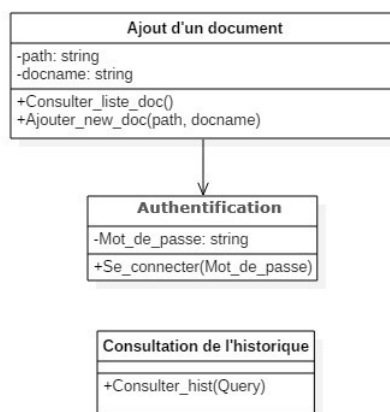


FIGURE 3.6 – Diagramme de classe de la couche Contrôleur

### 3.2.4 Diagramme de séquence : Cas de la recherche d'information

Ce cas est la fonctionnalité principale du système système. En fait, la Figure 3.7 décrit les interactions entre l'utilisateur et les différents composants du système lors de la recherche d'information.

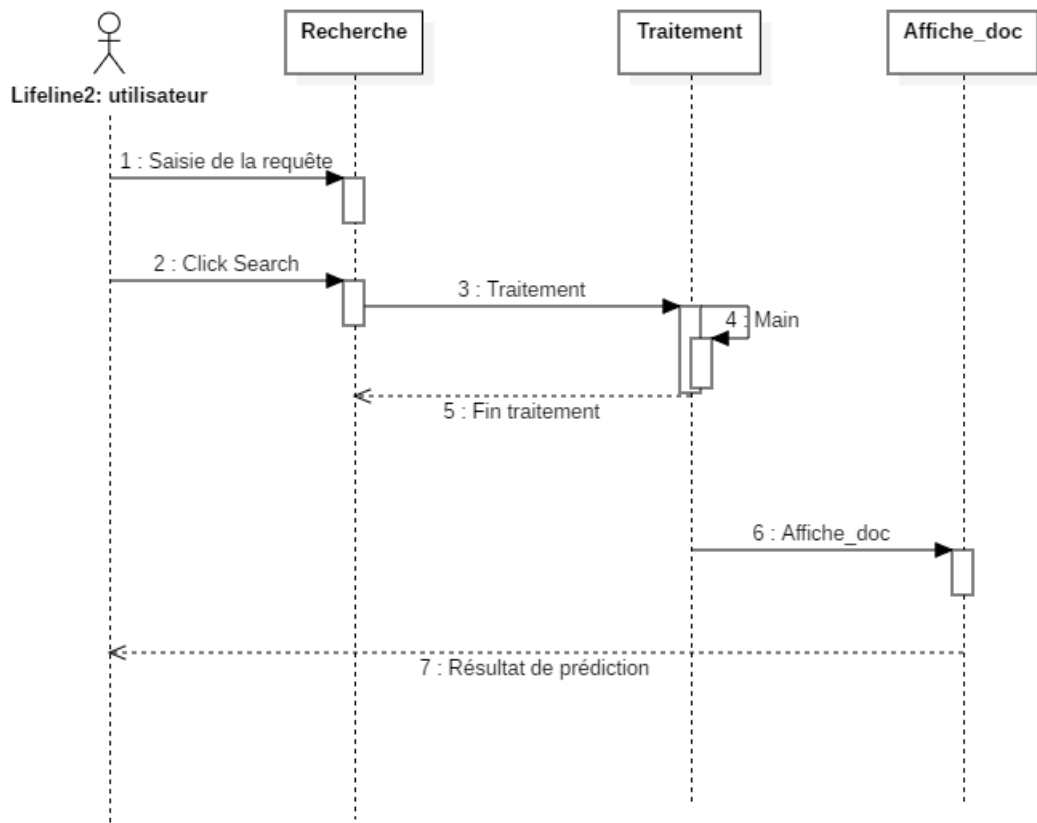


FIGURE 3.7 – Diagramme de séquence pour la recherche d'information

### 3.2.5 Diagramme de séquence : Cas d'ajout de document(s)

Ce cas est une concrétisation de l'interface d'interaction entre l'administrateur et le système. En fait, l'administrateur a les privilèges, non seulement de consulter la liste des documents, mais aussi d'ajouter un ou plusieurs documents comme le montre le diagramme de séquence illustré par la Figure 3.8.

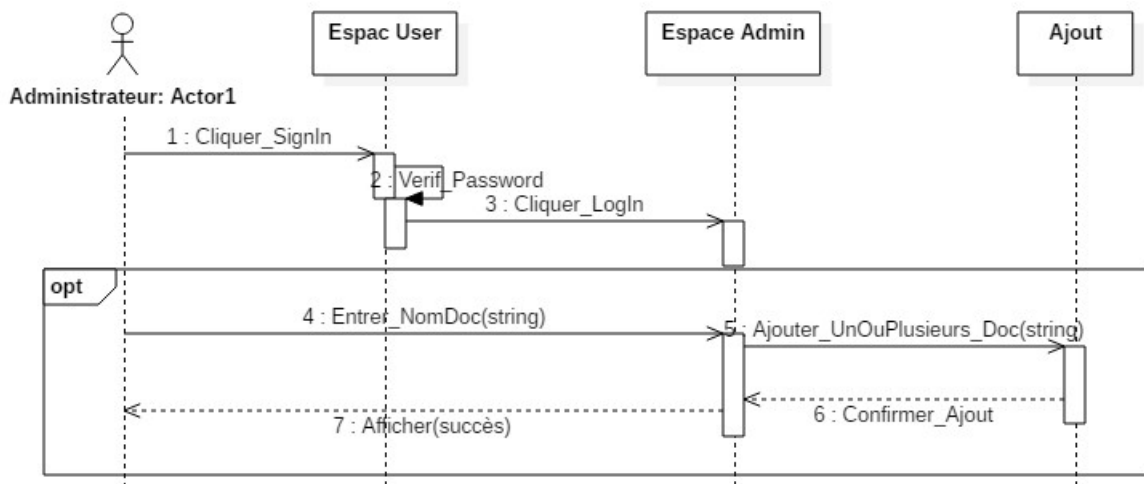


FIGURE 3.8 – Diagramme de séquence pur l'ajout de document(s)

### 3.3 Architecture du pré-traitement des données : FastText

La Figure 3.9 présente l'architecture suivie pour le pré-traitement des données. D'ailleurs, pour représenter chaque document et requête par l'ensemble de vecteurs de leurs mots, nous représentons chaque mot par un vecteur par la méthode de FastText après avoir former le dictionnaire de mots.

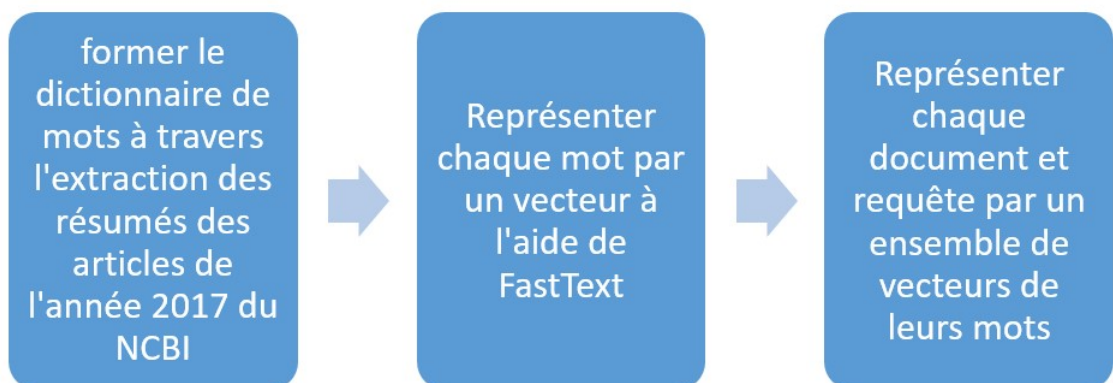


FIGURE 3.9 – Architecture de FastText

### 3.4 Architecture du réseau de neurones : Auto-encodeur LSTM

Notre réseau de neurones est un réseau profond. La profondeur signifie que nous sommes en train de traiter un cube de 3 dimensions que nous ne pouvons pas le voir avec un simple moniteur bidimensionnel. Ainsi, notre objectif est de réduire la quantité de documents. C'est pour cela que nous recourons à un auto-encodeur LSTM qui est un ensemble d'encodeurs automatiques où les sorties de chaque couche LSTM sont câblées aux entrées de la couche LSTM suivante. Il prend un vecteur ayant une dimension élevée à travers son réseau de neurones et essaye de compresser les données en une représentation plus petite. Le véritable avantage ici est la capacité de faire des prédictions basées sur une série de valeurs. L'auto-encodeur LSTM a deux composantes principales, comme le montre la Figure 3.10.

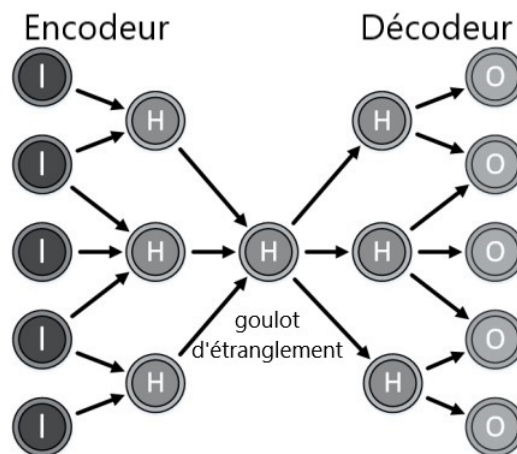


FIGURE 3.10 – Structure d'un auto-encodeur [Gensler et al.(2016)Gensler, Henze, Sick, and Raabe]

Le premier est l'encodeur qui est tout simplement un ensemble de couches LSTM entièrement connectées qui vont compresser l'entrée jusqu'à une plus petite représentation ayant moins de dimension : c'est le goulot d'étranglement. Cela signifie que les neurones dans cette couche LSTM sont beaucoup moins nombreux que les autres. Puis, le décodeur reconstruit l'entrée. Le nombre d'entrées est le même à l'entrée et à la sortie, donc nous pouvons attendre que la sortie est un vecteur non seulement ayant la même taille que l'entrée, mais en fait c'est le même vecteur. Nous devons enfin calculer la perte de reconstitution et comparer les différences du vecteur par rapport à celui de la sortie.

L'architecture de notre auto-encodeur est la suivante :

- deux couches LSTM, une représente l'encodeur et l'autre le décodeur avec un nombre de neurones égal à 400.
- une couche Dense qui représente la sortie de l'auto-encodeur avec un nombre de neurones égal à 200.

En but d'avoir une perte proche de 0, nous utilisons la méthode prédéfinie de Keras appelée Cross Validation qui nous permet de vérifier le nombre de neurones le plus adéquat.

### 3.4.1 Architecture du modèle de similarité : LearnToRank

Le modèle LearnToRank est un modèle de similarité qui, à partir de la représentation vectorielle de la requête ainsi que celle des documents, calcule les scores et les classe du plus proche de 1 jusqu'au plus proche de 0, comme le montre la Figure 3.11. En effet, nous effectuons le produit scalaire entre le vecteur de chaque document et sa requête et lui attribuons la valeur 1, ainsi que le produit scalaire entre le vecteur du document et sa requête-bar et lui attribuons 0.

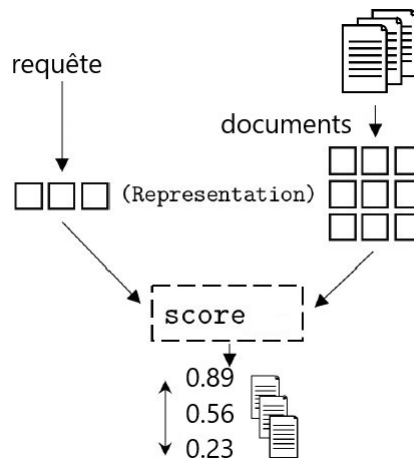


FIGURE 3.11 – LearnToRank

L'architecture du modèle LearnToRank est une architecture MLP (Multi Layer Perceptron) : deux couches Dense avec un nombre de neurones égale à 400 (dimension des vecteurs d'entrée) pour la première et 1 pour la la deuxième. Nous choisissons Sigmoid comme fonction d'activation car la sortie du modèle est entre 0 et 1.



## 3.5 Conclusion

Tout au long de ce chapitre, nous avons présenté une vue conceptuelle de notre projet. Dans le prochain chapitre, nous allons entamer la dernière étape qui est la réalisation du projet. Cette partie est dédiée à l'environnement de développement ainsi que les outils de travail.

# Chapitre 4

## Réalisation et évaluation du système TrustBIOMED

Ce chapitre est consacré à l'aspect implémentation, qui est la phase la plus délicate, et à la présentation de la solution finale. Nous abordons, dans la section 4.1, une description de l'environnement du travail et les technologies utilisées. Par la suite, dans la section 4.2, nous présentons les principales interfaces de TrustBIOMED à travers quelques captures d'écran que nous avons développées. La section 4.3 est consacrée à l'évaluation de notre travail.

### 4.1 Environnement de développement

Dans cette partie nous présentons les différents outils matériel et logiciel nécessaires pour le développement de notre système.

#### 4.1.1 Environnement matériel

Pour la réalisation de notre projet, nous avons opté pour l'utilisation de l'environnement matériel ayant comme caractéristiques :

- Marque : Asus.
- Processeur : Intel Core i7 @1.80GHz.
- Mémoire RAM : 8 Go.
- Disque dur : 900 Go.
- Système d'exploitation : Windows 10.

### 4.1.2 Environnement logiciel

Durant le cycle de développement, nous avons eu recours à un ensemble de logiciels utiles pour la réalisation des différentes tâches du système. En effet, nous avons utilisé :

- Le langage de programmation interprété : Python.
- L'environnement de développement libre pour Python : Spyder.
- La bibliothèque de Python : Keras avec backend TensorFlow.
- L'utilitaire graphique de création d'interfaces permettant de générer le code Python d'interfaces graphiques : QtDesigner.
- Le logiciel de modélisation UML : StarUML.
- Le service de cloud computing : Amazon Web Services (AWS).
- Le logiciel de composition de documents : Latex.

### 4.1.3 Justification des choix techniques

Choisir un outil de développement constitue une étape critique du projet puisque ce choix peut entraîner des gains en temps et en effort. La première raison pour choisir Python comme langage de programmation est qu'il est devenu la référence dans l'éducation nationale comme premier langage de programmation. De plus, Python est le langage le plus populaire dans le domaine de Deep Learning. Enfin, tous les géants de la technologie utilisent Python, que ce soit Google, Facebook, Netflix, Amazon, ou même la NASA ou IBM.

## 4.2 Description de TrustBIOMED

Afin de décrire les services offerts par notre système de recherche d'information biomédicale, nous présentons dans cette section l'aperçu de TrustBIOMED pour l'espace utilisateur, ainsi que l'espace administrateur.

### 4.2.1 Aperçu du travail réalisé pour l'espace utilisateur

L'espace utilisateur est composé de la page d'accueil, le résultat de sa recherche et son historique.

#### 4.2.1.1 Page d'accueil

Lorsqu'un chercheur accède à la page du système de recherche, il trouve l'interface illustrée par la Figure 4.1. Nous choisissons le slogan "Dans le Deep Learning, nous faisons confiance" pour notre système.

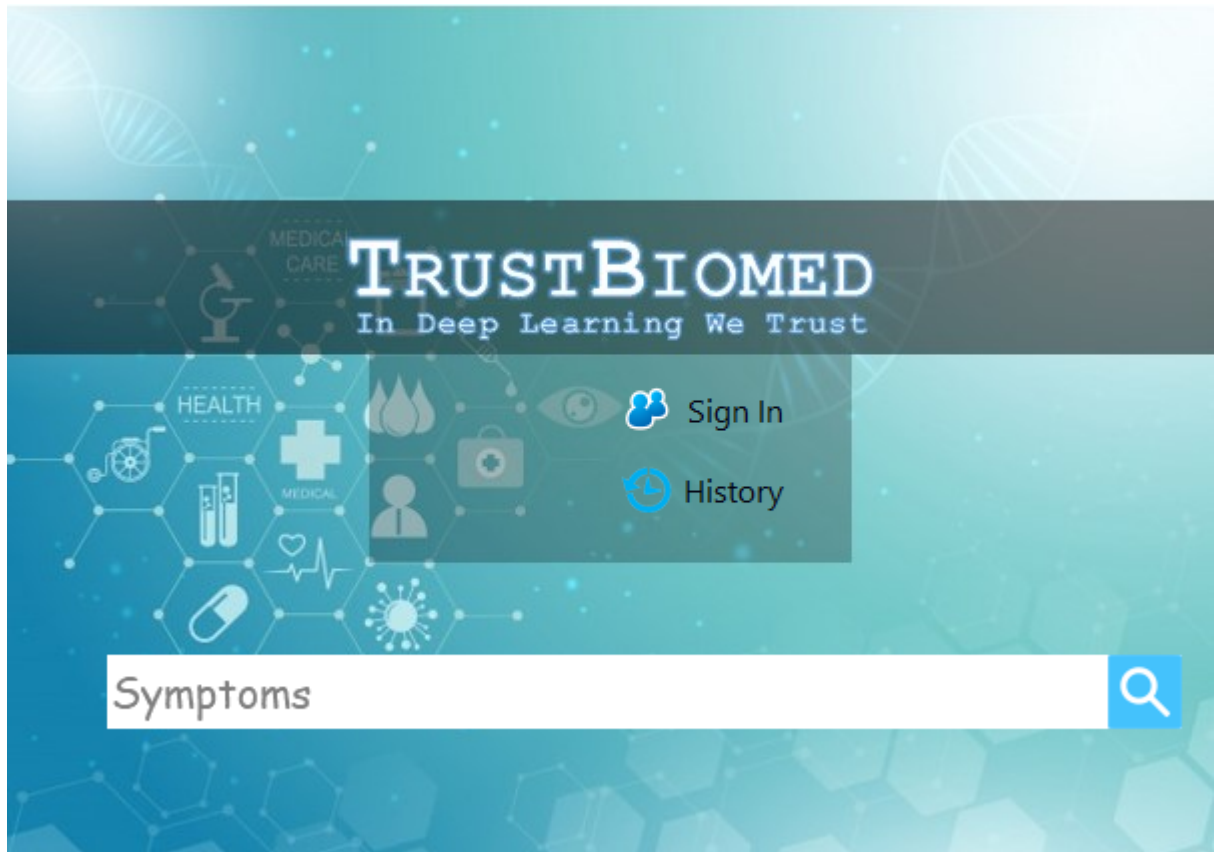


FIGURE 4.1 – Page d'accueil de TrustBIOMED

#### 4.2.1.2 Page des résultats de la recherche

Lorsque le chercheur procède à la recherche de l'information à travers notre système, les 5 premiers documents apparaissent comme le montre la Figure 4.2. Elle présente le résultat de prédiction associée à la requête "pain vomitting fever".

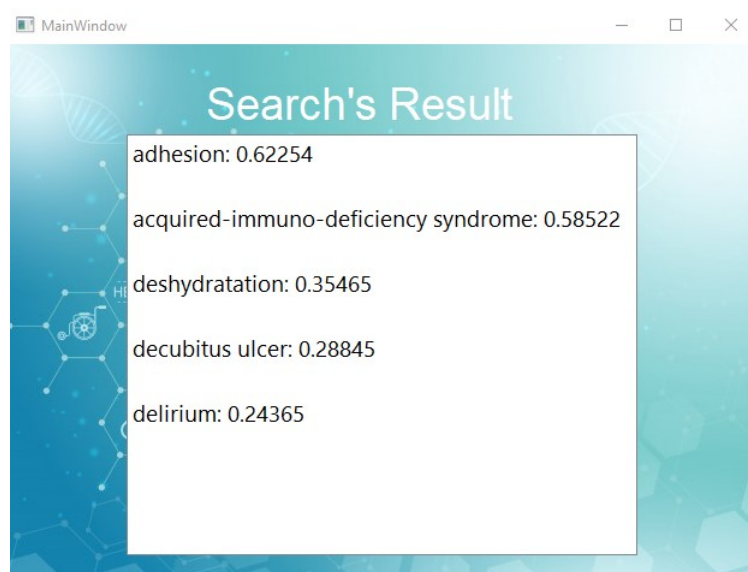


FIGURE 4.2 – Page des résultats de la recherche

#### 4.2.1.3 Page de l'historique

Pour consulter son historique de recherche dans la page d'historique présentée dans la Figure 4.3, l'utilisateur clique sur le bouton "History" de la page d'accueil.

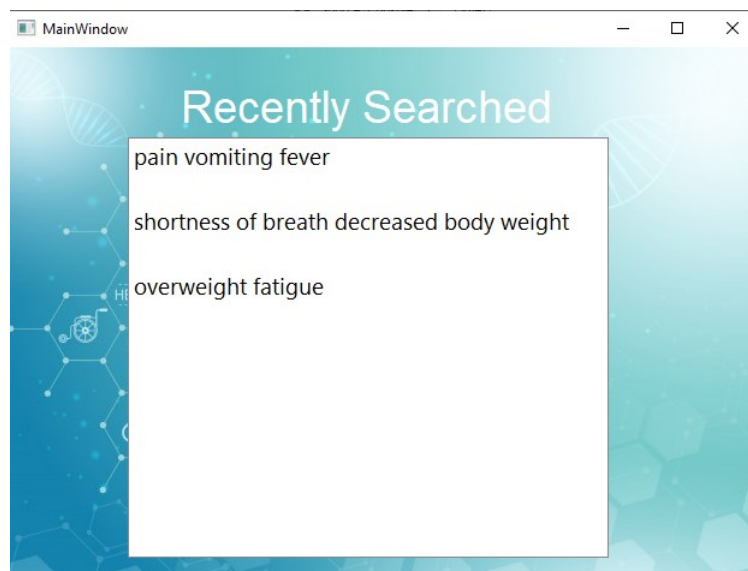


FIGURE 4.3 – Page de l'historique de recherches

#### 4.2.1.4 Page d'authentification

Afin d'accéder à l'espace administrateur, il suffit de cliquer sur le bouton "Sign In", illustré dans la Figure 4.4, et entrer le mot de passe.

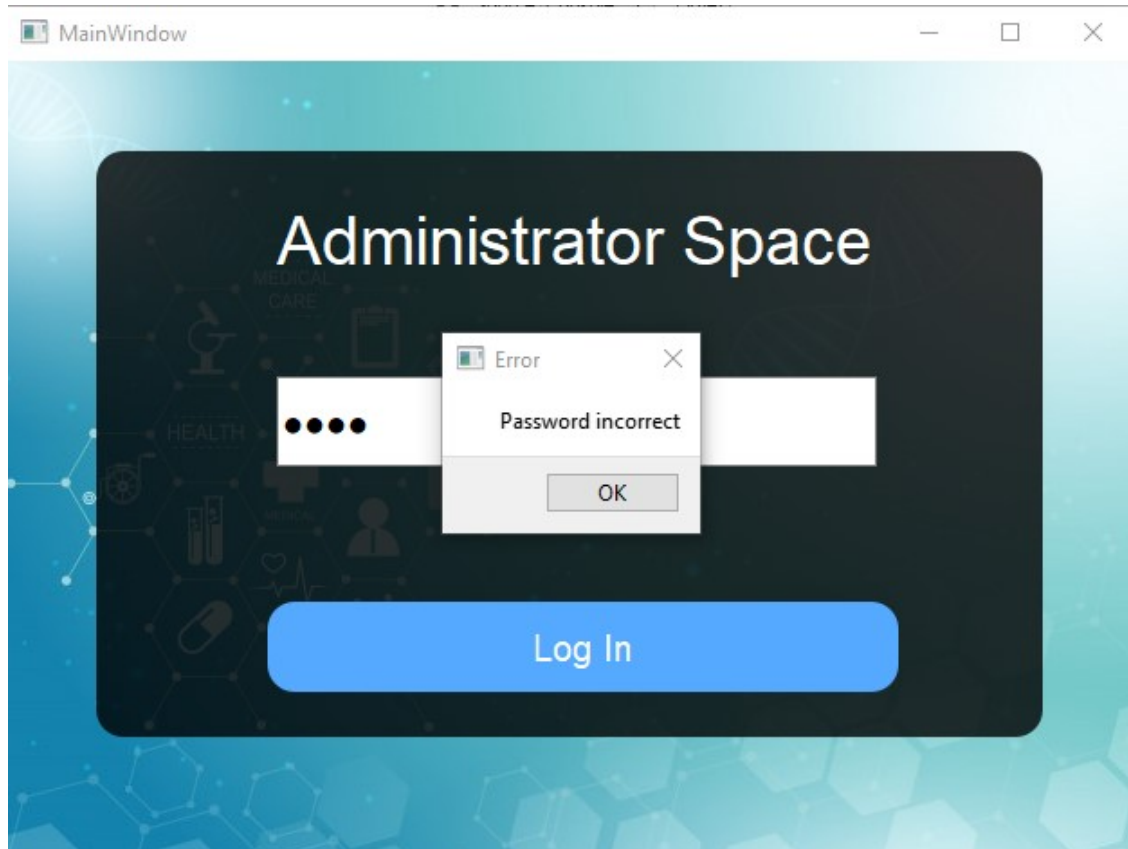


FIGURE 4.4 – Page d'authentification

#### 4.2.2 Aperçu du travail réalisé pour l'espace administrateur

En plus des pages de résultat et de l'historique des recherches, l'espace administrateur est composé d'une page affichant la liste des documents, ainsi qu'une page dédiée à l'ajout d'un ou plusieurs documents. Pour revenir à l'espace utilisateur, il suffit de cliquer sur le bouton "Sign Out".

#### 4.2.2.1 Page d'administration

Deux possibilités s'offrent : soit l'administrateur veut chercher une information, alors il saisit sa requête dans la zone de texte et en cliquant sur le bouton "Search", soit il veut ajouter de nouveaux documents dans la base avec un simple click sur le lien "Add documents" présenté dans la Figure 4.5.

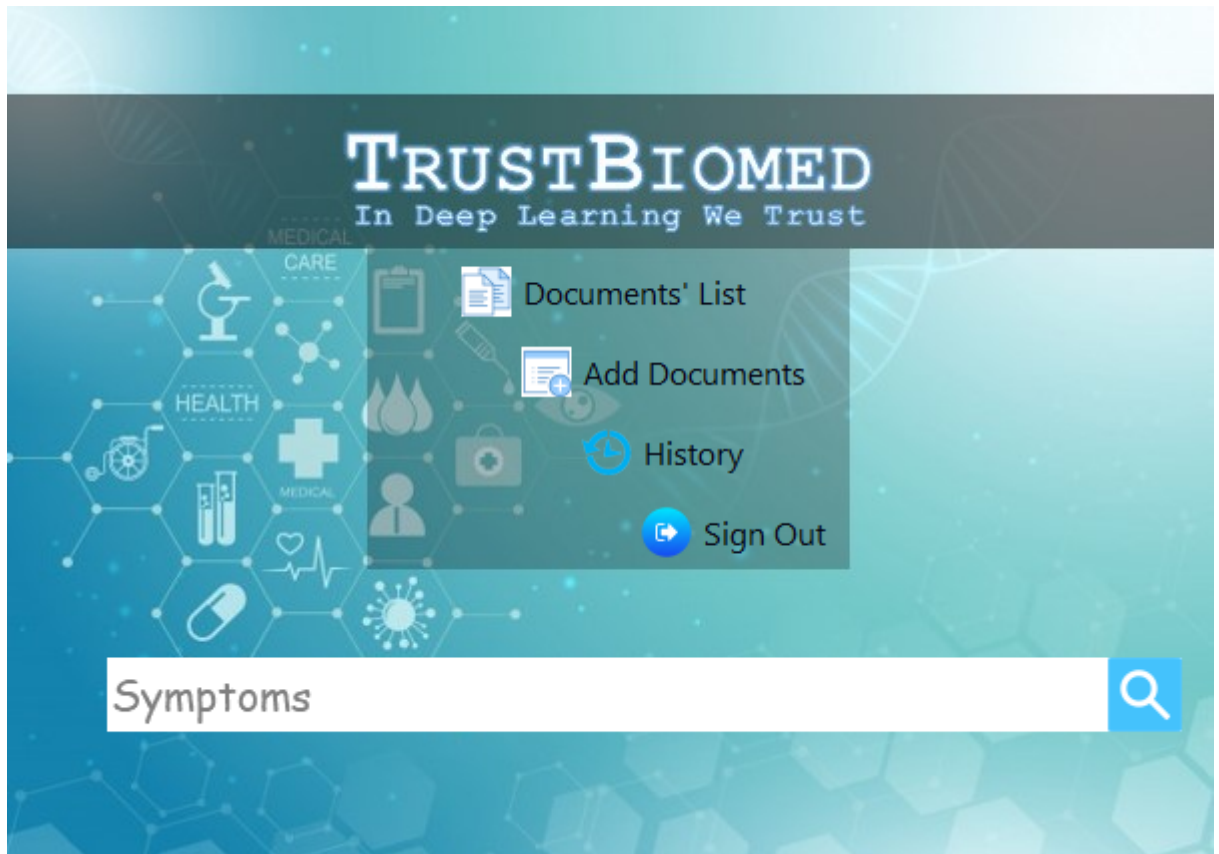
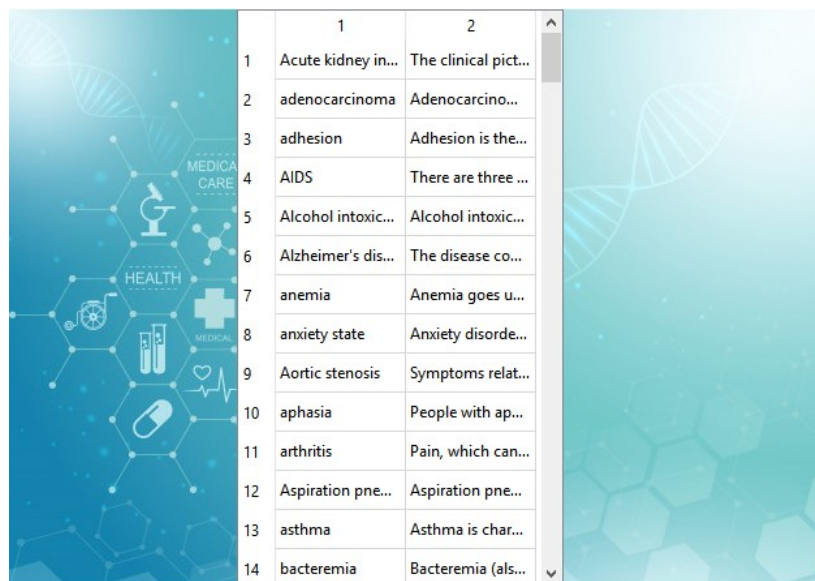


FIGURE 4.5 – Page de l'espace administrateur

#### 4.2.2.2 Page de consultation de la liste des documents

Le bouton "Documents' List" permet l'administrateur de consulter la liste des documents classés par ordre alphabétique comme le montre la Figure 4.6.

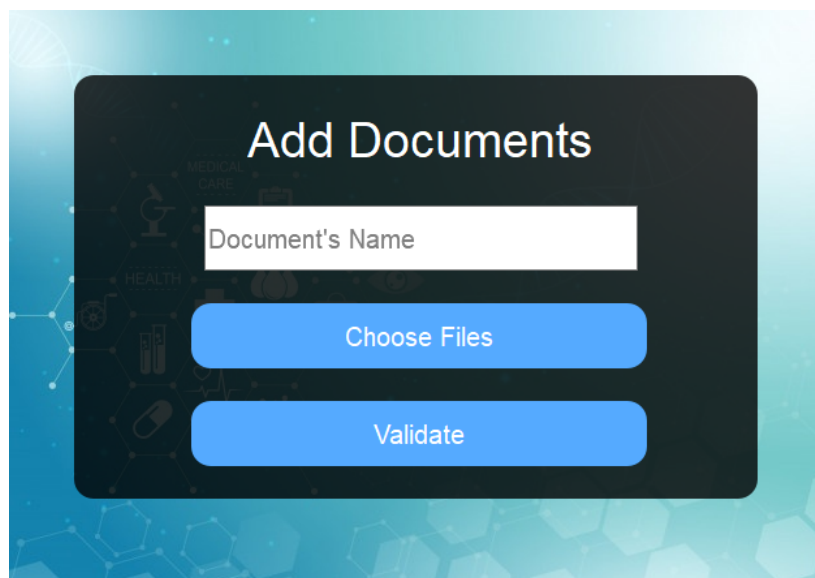


|    | 1                  | 2                    |
|----|--------------------|----------------------|
| 1  | Acute kidney in... | The clinical pict... |
| 2  | adenocarcinoma     | Adenocarcino...      |
| 3  | adhesion           | Adhesion is the...   |
| 4  | AIDS               | There are three ...  |
| 5  | Alcohol intoxic... | Alcohol intoxic...   |
| 6  | Alzheimer's dis... | The disease co...    |
| 7  | anemia             | Anemia goes u...     |
| 8  | anxiety state      | Anxiety disorde...   |
| 9  | Aortic stenosis    | Symptoms relat...    |
| 10 | aphasia            | People with ap...    |
| 11 | arthritis          | Pain, which can...   |
| 12 | Aspiration pne...  | Aspiration pne...    |
| 13 | asthma             | Asthma is char...    |
| 14 | bacteremia         | Bacteremia (als...   |

FIGURE 4.6 – Page de consultation de la liste des documents

#### 4.2.2.3 Page d'ajout d'un ou plusieurs documents

En cliquant sur le bouton "Add Documents", l'administrateur peut ajouter des documents après avoir choisi le chemin adéquat comme le montre les deux Figures 4.7 et 4.8.



## Add Documents

Choose Files

Validate

FIGURE 4.7 – Page d'ajout d'un ou plusieurs documents



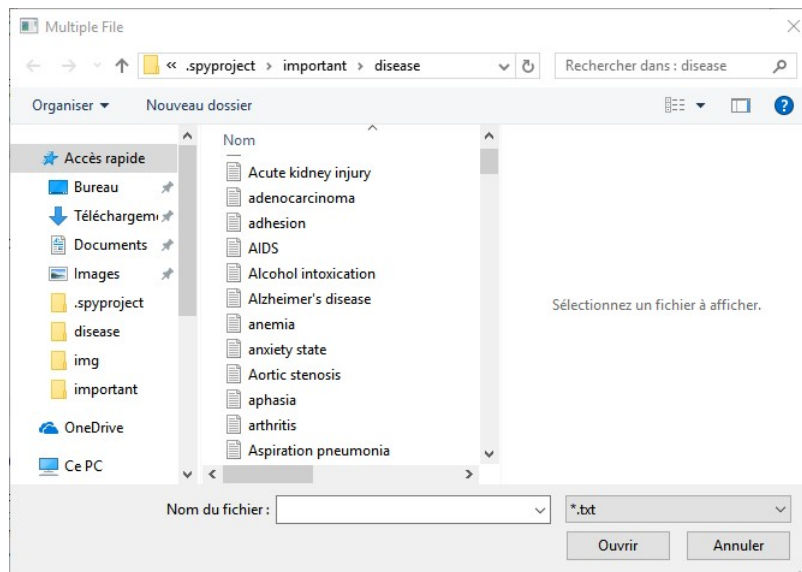


FIGURE 4.8 – Choix du chemin adéquat

## 4.3 Évaluation de TrustBIOMED

Afin de concrétiser notre travail, nous évaluons notre modèle dans cinq expériences : une évaluation du résultat de FastText. Nous décrivons ces expériences en détail dans la suite sections.

### 4.3.1 Exemple du résultat de FastText

Tout d’abord, nous formons le dictionnaire de mots à travers l’extraction des résumés des articles de l’année 2017 du NCBI. Lors de la construction du dictionnaire de mots, nous conservons ceux qui apparaissent au moins 5 fois dans l’ensemble de formation. Quelques mots des ensembles de données n’apparaissent pas dans nos données de formation, et ainsi, nous ne pouvons pas obtenir de représentation vectorielle pour ces mots. Donc afin de fournir des résultats comparables, nous proposons par défaut d’utiliser des vecteurs nuls pour ces mots. Pour notre modèle FastText, nous utilisons le paramètre suivant : les vecteurs des mots sont de dimension 200.

Les Figures 4.9 et 4.10 sont deux exemples du résultat de FastText. Les valeurs des vecteurs FastText sont compris entre -1 et 1.

```
In [28]: print(new_dict.get("AIDS"))
[-0.05911401  0.15665674  0.03686748 -0.23946212 -0.43930283 -0.2741789
 -0.24446622  0.28171852  0.26561242 -0.999522   0.1992125   0.33669266
  0.28964242  0.37081087 -0.23091342 -0.28420827 -0.2478952   0.07325625
 ...
  0.28141668  0.17440203  0.24753775 -0.02666834  0.50467217 -0.26006746
  0.08130024  0.37410653  0.01823288  0.21832389 -0.38364366  0.24700779
 -0.17122668  0.07932722 -0.02370384 -0.22286604  0.33914736  0.56336594]
```

FIGURE 4.9 – Vecteur du mot "AIDS"

```
In [29]: print(new_dict.get("Fever"))
[-0.33911097  0.30350766  0.54346186 -0.13032751  0.02141028  0.19747193
 -0.05059336  0.11496731  0.49916568 -0.42510903  0.31775054  0.529079
 -0.42723656  0.28204128 -0.30431592 -0.1912813  -0.35860342 -0.65322304
 ...
  0.22197087  0.25950375  0.92550075 -0.27221948  0.019928   -0.51907355
  0.08434539  0.37646574  0.14566769 -0.34213498 -0.33919114 -0.22132349
 -0.07819756 -0.97752535  0.6134495  -0.67335093 -0.04792448 -0.05062163]
```

FIGURE 4.10 – Vecteur du mot "Fever"

### 4.3.2 Exemple du résultat de l'auto-encodeur

Nous pouvons évaluer la qualité de l'apprentissage de l'auto-encodeur des documents par la courbe de la perte illustrée par la Figure 4.11, et celle de l'auto-encodeur des requêtes par la Figure 4.12. Elles montrent que la perte est très proche de 0, ce qui explique que le vecteur d'entrée et celui de sortie des auto-encodeurs est presque le même.

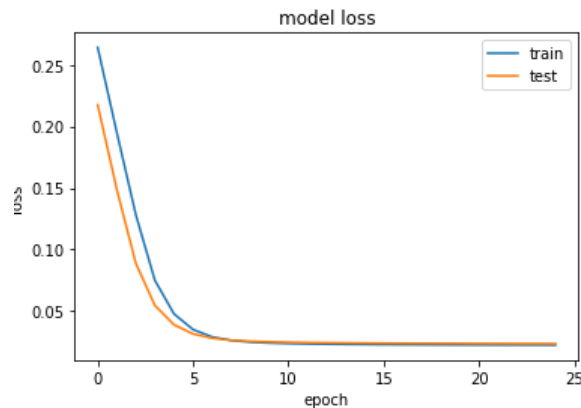


FIGURE 4.11 – Courbe de la perte pour l'apprentissage des documents

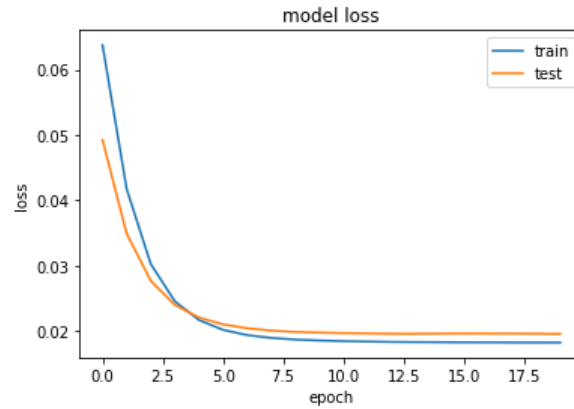


FIGURE 4.12 – Courbe de la perte pour l'apprentissage des requêtes

### 4.3.3 Métrique d'évaluation de TrustBIOMED

Le rappel est le rapport entre le nombre de documents pertinents trouvés et le nombre total de documents pertinents.

Exemple : Pour la requête "pain vomiting fever" saisie, considérons les documents attendus : (10 éléments) "adhesion, acquired-immuno-deficiency syndrome, deshydratation, decubitus ulcer, deliriumet, dementia, diverticulitis, exantema, fibroid tumor, gastroenteritis" l'ensemble extrait trouvé : (5 éléments) "adhesion, acquired-immuno-deficiency syndrome, deshydratation, decubitus ulcer, delirium". Donc le rappel est égal à :  $R = 5 / 10 = 0.5$ .

## 4.4 Conclusion

Tout au long de ce dernier chapitre, nous avons exposé la réalisation de notre travail. Dans un premier lieu, nous avons explicité l'environnement matériel et logiciel, ainsi que les technologies utilisées. Ensuite, nous avons décrit les principales fonctionnalités implémentées, par la présentation des captures d'écrans témoignant les différentes tâches. Nous avons clôturé, enfin, par une évaluation de notre travail.

# Conclusion et perspectives

Vu la croissance séquentielle du nombre de documents biomédicales, il s'agit de proposer une solution qui permettra d'améliorer et de faciliter la procédure de la recherche d'informations. Nous avons proposé de concevoir un système de recherche d'informations biomédicales qui sert à construire une liste des documents les plus pertinents (contenant les maladies) en fonction d'une requête saisie.

Notre démarche consiste à définir quelques modèles de la représentation vectorielle des mots : LSA, Word2Vec et FastText, le modèle que nous avons utilisé dans notre travail. Ensuite, nous avons spécifié les besoins fonctionnels et non fonctionnels qui représentent les fonctionnalités et les critères que TrustBIOMED doit respecter. Par la suite, nous avons traité la partie conception globale et détaillée, et décrit l'architecture MVC suivie par TrustBIOMED. Enfin, nous avons introduit la phase de réalisation du système accompagnée par des captures d'écran. Ensuite, nous avons évalué la performance des méthodes utilisées en se basant sur la métrique rappel qui a donné 0.5 comme valeur.

Finalement, nous tenons à signaler qu'il existe encore plusieurs chemins pour l'amélioration du système. A court terme, nous envisageons à une amélioration de l'IHM de TrustBIOMED, et à moyen terme, nous travaillons sur l'augmentation des données reconnues par le système, ainsi que l'intégration d'autres méthodologies de représentation vectorielle et de "Ranking" pour avoir une performance meilleure.

# Bibliographie

- [Sauvagnat(2005)] Karen Sauvagnat. *Modele flexible pour la Recherche d'Information dans des corpus de documents semi-structurés*. PhD thesis, Université Paul Sabatier-Toulouse III, 2005.
- [Mikolov et al.(2013)] Mikolov, Chen, Corrado, and Dean] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv :1301.3781*, 2013.
- [Nielsen(2015)] Michael A Nielsen. *Neural networks and deep learning*, volume 25. Determination press USA, 2015.
- [Stérin(2016)] Tristan Stérin. Réseaux de neurones récurrents et mémoire : application à la musique. 2016.
- [Gensler et al.(2016)] Gensler, Henze, Sick, and Raabe] André Gensler, Janosch Henze, Bernhard Sick, and Nils Raabe. Deep learning for solar power forecasting—an approach using autoencoder and lstm neural networks. In *2016 IEEE international conference on systems, man, and cybernetics (SMC)*, pages 002858–002865. IEEE, 2016.
- [Cousyn(2018)] Charles Cousyn. *Exploration d'articles scientifiques sur les maladies rares pour l'extraction d'informations*. PhD thesis, Université du Québec à Chicoutimi, 2018.
- [Bourgeade et al.(2018)] Bourgeade, Muller, and Serrurier] Tom Bourgeade, Philippe Muller, and Mathieu Serrurier. Représentation sémantique et structurelle de conversations par chat. 2018.

# Résumé

Ce projet est réalisé dans le cadre des projets de conception et de développement pour les étudiants de deuxième année de l'Ecole Nationale des Sciences de l'Informatique(ENSI).

Il a pour objectif la conception d'un système de recherche d'informations biomédicales basé sur le Deep Learning. Ce système permet aux utilisateurs d'avoir un classement des documents (comportant les maladies) les plus adéquates à leurs requêtes saisies (comportant des symptômes).

Nous nommons notre système TrustBIOMED. En Effet, c'est grâce à lui que l'utilisateur peut avoir une idée sur la maladie à travers les documents pertinents qu'il affiche. D'où l'utilisateur doit relativement faire confiance aux informations biomédicales fournies par TrustBIOMED.

Mots clés : Word Embedding, FastText, LSTM, Learn to Rank

# Abstract

This project is carried out as a part of design and development projects for second-year students of the National School of Computer Science.

It aims to design a biomedical information search system based on Deep Learning. This system allows users to have a ranking of documents (include diseases) most appropriate to their queries (include symptoms).

We name our system : TrustBIOMED. Indeed, it is thanks to him that the user can have an idea about the disease through the relevant documents it displays. Hence the user must relatively trust in the biomedical information provided by TrustBIOMED.

Keys Words : Word Embedding, FastText, LSTM, Learn to Rank

## ملخص

يتم تنفيذ هذا المشروع في إطار مشاريع التصميم والتطوير لطلاب السنة الثانية بالمدرسة الوطنية لعلوم الاعلامية.

يهدف هذا المشروع إلى تصميم نظام بحث عن المعلومات الطبية الحيوية باستعمال التعلم العميق. يتيح هذا النظام للمستخدمين تصنيف المستندات (التي تحتوي الأمراض) الملائمة لاستفساراتهم (التي تحتوي الأعراض).

اخترنا TrustBIOMED كإسم لهذا النظام. في الواقع، بفضلته يمكن للمستخدم أن تكون لديه فكرة عن المرض من خلال الوثائق ذات الصلة التي يعرضها. وبالتالي يجب على المستخدم أن يثق نسبياً بالمعلومات الطبية الحيوية التي يقدمها TrustBIOMED.