2.3 Concepts de base

Les systèmes de recommandation sont des systèmes capables de fournir des recommandations personnalisées permettant de guider l'utilisateur vers des ressources intéressantes et utiles au sein d'un espace de données important. [3] Les ressources en question sont communément appelées « items ».

Dans cette section, nous allons définir tous les concepts nécessaires au développement de notre approche, à savoir :

1. L'ontologie.
2. Le profil utilisateur.
3. Les techniques de filtrages d'informations.
4. Les réseaux de neurones.
5. Le traitement automatique des langages.

2.3.1 Définition d'une ontologie

Dans le contexte de l'informatique et des sciences de l'information, une ontologie définit un ensemble de primitives de représentation pour modéliser un domaine de connaissances. Les primitives de représentation sont généralement des concepts (ou des classes), des attributs (ou des propriétés), et des relations (ou des liens qui relient des éléments de classe). Les définitions des primitives de représentation incluent des informations sur leurs significations et des contraintes sur leurs applications, qui doit être logiquement cohérente.[4]

Une ontologie peut être vue comme un graphe où les nœuds représentent des concepts et les arcs représentent des relations entre ces concepts. La figure 2.5 présente un exemple d'ontologie.

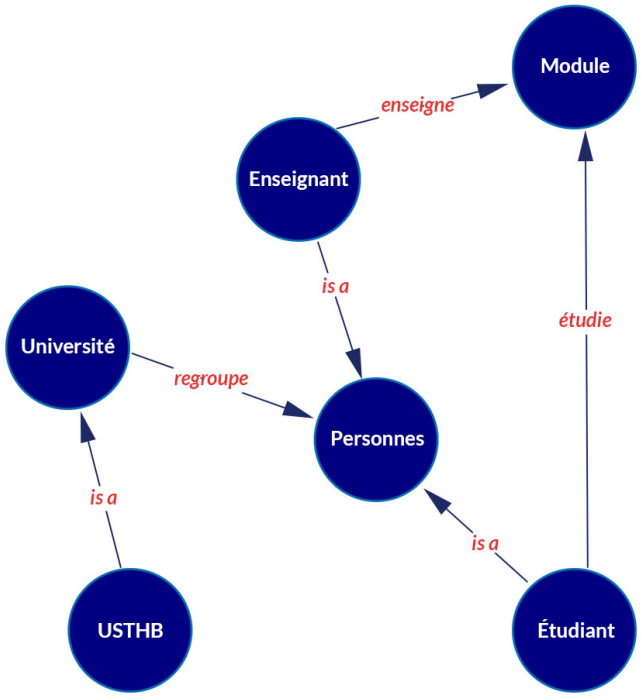

FIGURE 2.5 – Exemple d'une ontologie

2.3.2 Profil utilisateur

L'utilisateur est au centre des systèmes de recommandation. Il est à la fois consommateur et contributeur. Ainsi, la qualité des informations disponibles en entrée sur l'utilisateur est une condition *sine qua non* pour l'efficacité des systèmes de recommandation. Ces informations sont stockées dans le profil utilisateur, qui se définit généralement comme une structure qui permet de stocker et modéliser les préférences des utilisateurs et leurs centres d'intérêts [5]

Le profil utilisateur est alimenté par deux types d'informations :

1. **Les informations explicites :** Ce sont les informations que déclare directement un utilisateur en ce qui concerne ses préférences.
2. **Les informations implicites :** Ce sont les informations inférées à partir de l'activité de l'utilisateur concernant ses préférences.

Modélisation du profil utilisateur

Afin d'agencer les informations des profils utilisateur dans une structure de données, il existe deux modèles répondus dans la littérature qui sont les suivants :

— **Le modèle vectoriel** : Dans cette représentation, le contenu du profil utilisateur est caractérisé par un ou plusieurs vecteurs de termes pondérés. Ces termes sont obtenus à partir de plusieurs sources d'information recueillies sur l'utilisateur.[6]

— **Le modèle à base d'ontologie :** Les ontologies sont utilisées pour représenter les relations sémantiques entre les unités d'informations constituant le profil utilisateur. Dans ce modèle, le profil utilisateur est vu comme une hiérarchie de concepts pondérés.

2.3.3 Techniques de filtrage d'informations

Une quantité abondante d'informations est créée et diffusée sur les médias électroniques. Les utilisateurs sont submergés par le flux d'informations. Le filtrage des informations est l'une des méthodes qui évolue rapidement pour gérer de grands flux d'informations. Le but du filtrage des informations est d'exposer les utilisateurs uniquement aux informations qui les concernent. [7] Les systèmes de recommandations sont une sous-classe des systèmes de filtrage d'information dont le but est de prédire la "note" ou la "préférence" qu'un utilisateur donnerait à un item. [8] Nous allons présenter quatre techniques de filtrage : Le filtrage collaboratif, le filtrage sémantique, le filtrage social et le filtrage hybride.

1 - Filtrage collaboratif

Le filtrage collaboratif (FC) est un algorithme de recommandation qui se base principalement sur l'exploitation des évaluations que des utilisateurs ont faites sur certains items afin de recommander ces mêmes items à d'autres utilisateurs « similaires ».[9]

La représentation commune de ces évaluations repose sur une matrice appelée « matrice d'usage » où chaque ligne correspond à un utilisateur, et chaque colonne correspond à un item. Chaque case de la matrice de coordonnées (i, j) contient la note que l'utilisateur i a attribué à l'item j.

Nous pouvons distinguer deux types de FC : le FC basé mémoire, et le FC basé modèle. Nous allons nous intéresser dans notre recherche au second type.

- FC basé modèle

Les algorithmes basés « modèles » appliquent des techniques d'apprentissage automatique sur une base de données d'évaluations d'utilisateurs afin de construire une structure de données appelée « modèle » qui sera utilisé pour effectuer les prédictions.

Parmi les méthodes les plus performantes de FC, nous mentionnons les techniques de factorisation matricielle et les méthodes basées sur des réseaux de neurones artificiels. Nous présenterons ces techniques plus en détails par la suite.

- Inconvénient majeur du FC :

— Le démarrage à froid : Comme expliqué précédemment, l'algorithme de FC se base sur d'anciennes évaluations d'un utilisateur afin de calculer des prédictions. Cependant, lorsqu'un nouvel utilisateur s'inscrit dans la base de données, l'algorithme ne dispose pas d'évaluations sur lesquels se baser pour effectuer des prédictions. Analogiquement, lorsqu'un nouvel item apparait dans la base de données, cet item ne dispose d'aucune évaluation de la part des utilisateurs. Dans ce cas, l'algorithme ne dispose donc pas d'informations pour le recommander à des utilisateurs. Il existe donc deux problèmes liés au démarrage à froid : nouvel item et nouvel utilisateur.

2 - Filtrage sémantique

Le filtrage sémantique (Fsem) est un algorithme de recommandation qui se base sur l'aspect sémantique des items et des utilisateurs. Cet algorithme repose sur un calcul de similarité entre les informations sémantiques des items et des utilisateurs pour ensuite calculer des prédictions. L'aspect sémantique est souvent modélisé à l'aide d'ontologies.

- Inconvénient majeur du filtrage sémantique

— **Effet entonnoir :** Le filtrage sémantique ne peut pas recommander des items dont les catégories n'appartiennent pas aux catégories auxquels s'intéresse l'utilisateur.

3 - Filtrage social

Le filtrage social (Fsoc) est un algorithme de recommandation qui se base sur les relations sociales entre les utilisateurs. En effet les utilisateurs sont souvent influencés pas leur cercle social lors de la prise de décision.

Un cercle social peut être représenté par un graphe dans lequel les noeuds représentent des individus et les arcs la relation entre eux. Parmi les relations sociales les plus utilisées, on retrouve l'amitié [10] où le degré de confiance est définie par « la croyance en la capacité des autres à fournir des notes crédibles » [11]. L'algorithme de filtrage social se base sur ce cercle social afin de calculer des prédictions. La figure 2.6 présente un exemple de représentation d'un cercle social en considérant les utilisateurs suivants : u1, u2, u3, u4, u5 et u6.

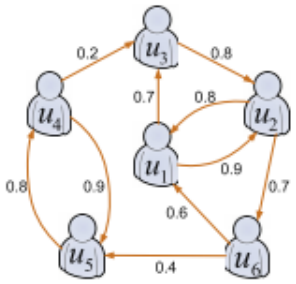

FIGURE 2.6 – Représentation d'un cercle social

- Inconvénient du filtrage social :

Démarrage à froid : Lors de l'introduction d'un nouvel utilisateur à la base de données, il ne pourra être intégré dans le graphe social que lorsqu'il aura fait au moins une interaction avec les autres utilisateurs. En attendant ces interactions, la recommandation d'items pour ce nouvel utilisateur à l'aide du filtrage social restera impossible.

4 - Filtrage hybride

Le filtrage hybride est un algorithme qui combine plusieurs autres algorithmes de recommandation.

Il existe plusieurs techniques d'hybridation dont les suivantes :

- **Hybridation pondérée :** Les algorithmes sont combinés selon un degré d'importance pour chacun.
- **Hybridation en cascade :** Les résultats d'une technique de recommandation sont raffinés par une autre technique.

2.3.4 Réseau de neurones artificiel

Un réseau neuronal artificiel (RN) est la partie d'un système informatique conçu pour simuler la façon dont le cerveau humain analyse et traite les informations. C'est le fondement de l'intelligence artificielle (IA). Il résout des problèmes qui s'avéreraient impossibles ou difficiles selon les normes humaines ou statistiques.

Le but d'un réseau de neurones est d'exécuter des calculs complexes et de trouver, par apprentissage, une relation non linéaire entre des données numériques et des paramètres. [12]

Principe de fonctionnement

Chaque neurone porte une valeur entre 0 et 1, et chaque association entre deux neurones représente le poids du nœud de départ. Le nœud d'arrivé porte la valeur de la somme pondérée des nœuds avec les poids auxquels il est relié. Cette somme est ensuite passée dans une fonction dite « fonction d'activation » qui retourne une valeur entre 0 et 1. Le but de l'algorithme est d'ajuster ces « poids » afin de faire correspondre les neurones d'entrée aux neurones de sortie selon les valeurs passées pour l'entrainement. la figure ci-dessous illustre cette explication :

Note :

Le « bias » peut être vu comme le seuil d'activation d'un neurone.

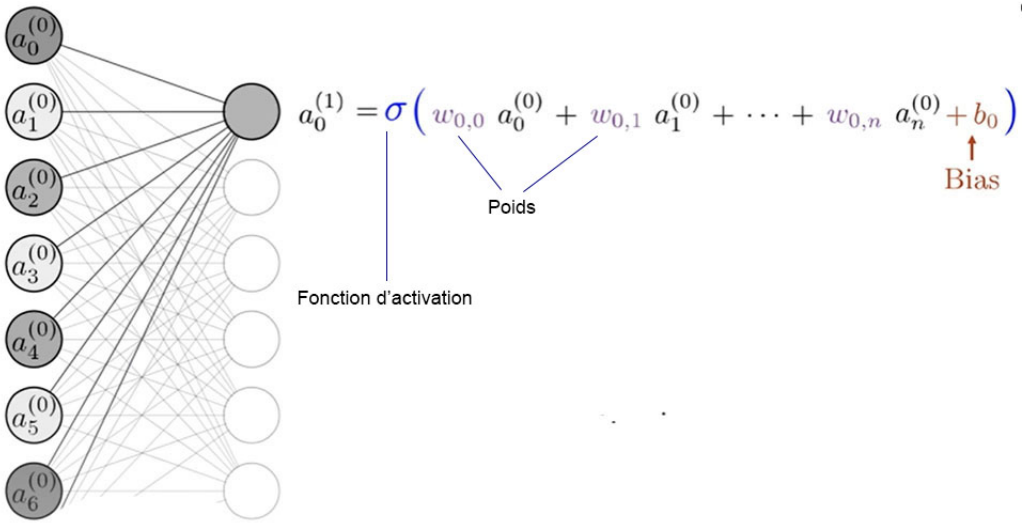

FIGURE 2.7 – Fonctionnement d'un réseau de neurones artificiel

2.3.5 Traitement automatique des langages

Le traitement automatique des langages (TAL) est un sous-domaine de l'informatique et de l'intelligence artificielle qui s'intéresse aux interactions entre les ordinateurs et les langages (naturels) humains. Il est utilisé pour appliquer des algorithmes d'apprentissage automatique au texte et à la parole. [13]

Document embedding (Doc2Vec)

Doc2Vec est une technique de TAL représentée par un modèle qui incorpore des textes dans un espace vectoriel de dimension réduite à l'aide de techniques d'apprentissage. Le résultat est un ensemble de vecteurs de texte où les vecteurs rapprochés dans l'espace vectoriel ont des significations similaires basées sur le contexte, et les vecteurs de textes distants les uns des autres ont des significations différentes.[14] Par exemple :

$$vecteur(\text{"}Cet\ appareil\ est\ genial\ !\text{"}) \approx vecteur(\text{"}Cet\ ordinateur\ est\ super\ !\text{"})$$

2.4 Modélisation des profils

Nous avons conçu les profils utilisateurs et items en fonction des informations implicites et explicites nécessaires à l'exécution des différents algorithmes que nous avons développés.

2.4.1 Profil utilisateur

La figure suivante illustre les informations implicites et explicites incluses dans le profil d'un utilisateur.

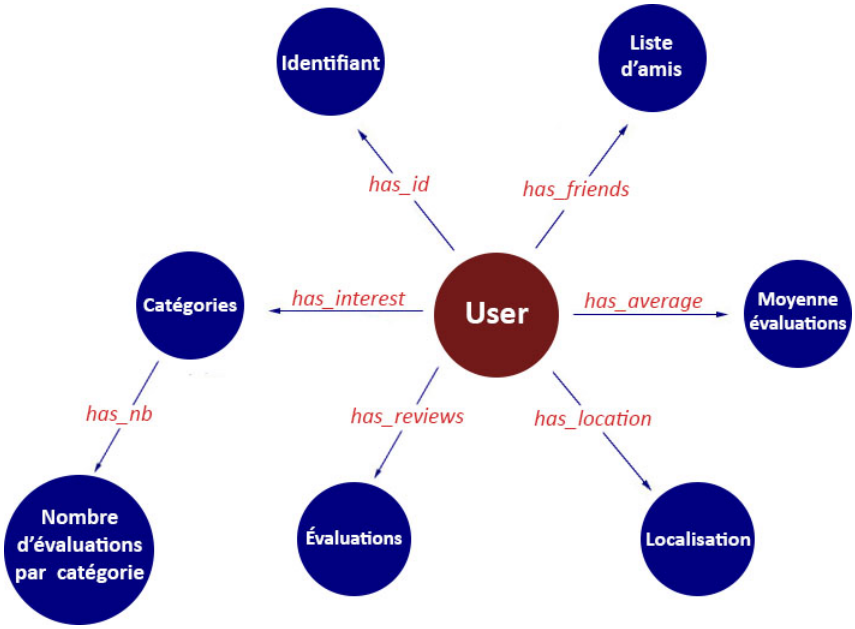

FIGURE 2.8 – Modélisation du profil utilisateur

2.4.2 Profil item

La figure suivante illustre les informations implicites et explicites incluses dans le profil d'un item.

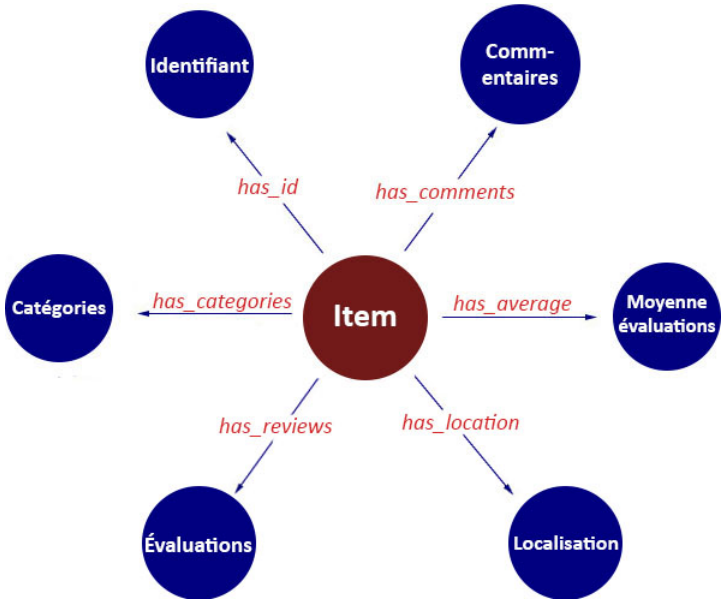

FIGURE 2.9 – Modélisation du profil item

2.5 Ontologie de domaines

Nous avons modélisé le concept « catégorie » en ontologie. Ce concept représente les différentes catégories auxquelles peut appartenir un item et auxquelles peut s'intéresser un utilisateur. L'importance de cette ontologie repose sur le calcul de similarité nécessaire pour le filtrage sémantique. La figure suivante illustre la taxonomie du concept « catégorie » de la base de données Yelp. Nous présenterons cette base de données en détail dans le chapitre suivant.

FIGURE 2.10 – Modélisation des catégories de la base de données Yelp

2.6 Notre approche de recommandation

Dans cette section nous allons décrire notre approche pour la recommandation des services. Cet algorithme sera intégré dans notre système mobile d'offre de services. Notre système de recommandation peut être décomposé en les modules suivants :

- Module de filtrage collaboratif : FC basé modèle en utilisant la décomposition matricielle PMF et le FC basé modèle en utilisant les réseaux de neurones (avec et sans la technique du TAL).
- Module de filtrage sémantique.
- Module de filtrage social.
- Module d'hybridation.
- Module de localisation.

La figure suivante résume notre approche :

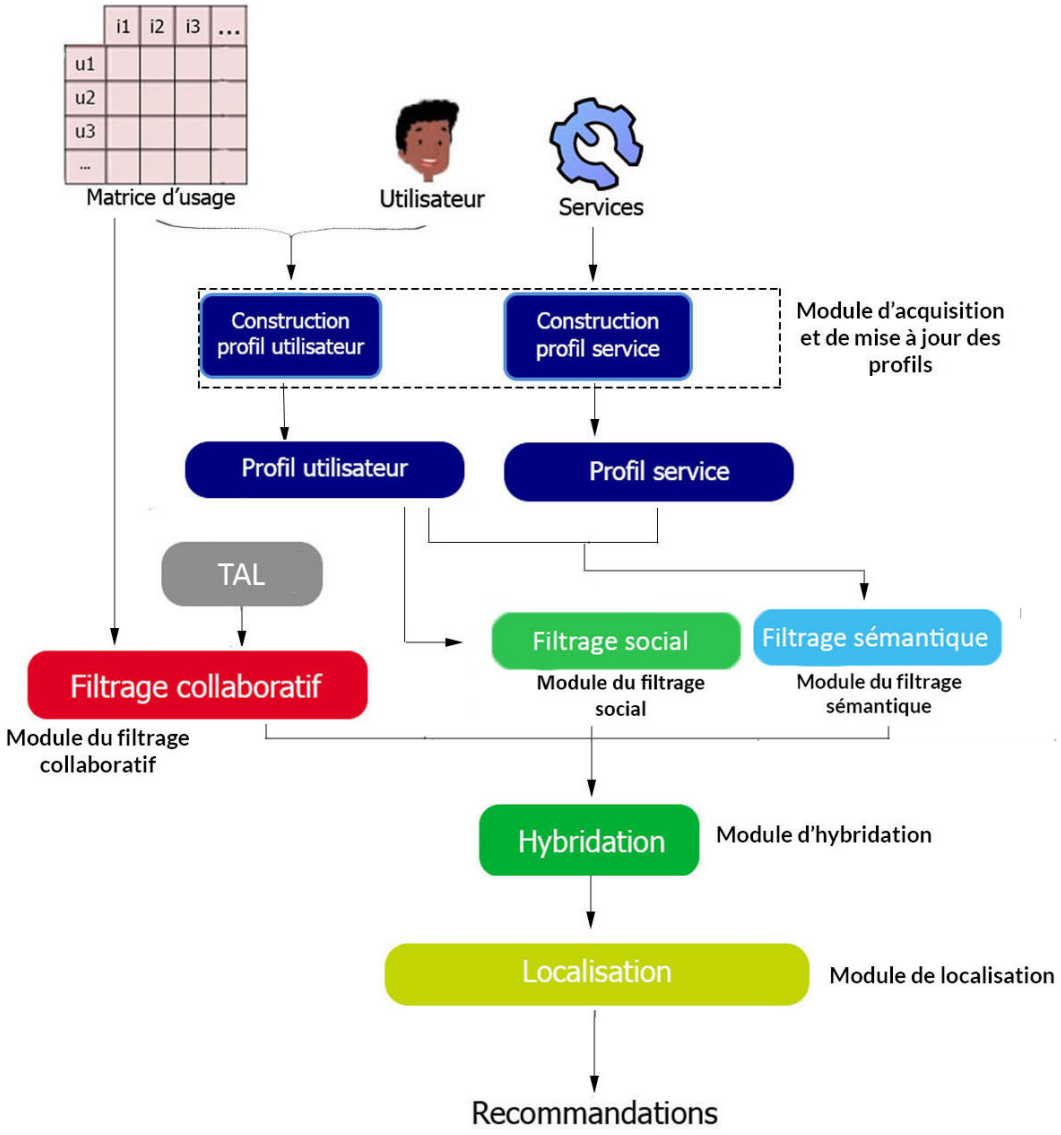

FIGURE 2.11 – Description de l'approche globale

2.6.1 Module d'acquisition et de mise à jour des profils

Avant de pouvoir recommander, nous avons besoin d'alimenter notre ontologie. Pour notre approche nous avons besoin des informations suivantes :

Information explicites

- Identifiants des utilisateurs et des services.
- Les évaluations effectuées par les utilisateurs.
- Commentaires des utilisateurs.
- Les catégories auxquelles appartiennent les services.
- Liste d'amis des utilisateurs.
- Localisation des utilisateurs et des services.

Informations implicites

— Les catégories auxquelles s'intéressent les utilisateurs.
— Évaluation moyenne d'un utilisateur.
— Le nombre d'évaluations par catégories des utilisateurs.
— Évaluation moyenne des services.

Remarque : Dans notre contexte, *i.e.* la recommandation de services, les notes attribuées par les utilisateurs reflètent uniquement la qualité des services. Ce qui veut dire que si un utilisateur attribue une mauvaise note, par exemple à un restaurant, alors cette note reflète la mauvaise qualité du service et non le fait que l'utilisateur n'apprécie pas les restaurants. Il est au contraire intéressé par les restaurants mais insatisfait du service d'un restaurant en particulier. C'est pour cela qu'on assume qu'un utilisateur est intéressé par la catégorie d'un service qu'il a noté quelle que soit la note attribuée (*i.e.* cela montre son intérêt à la catégorie restaurant du système).

Algorithme pour l'acquisition des informations implicites

L'algorithme de mise à jour du profil utilisateurs par acquisition des informations implicites s'énonce comme suit :

Algorithme 1 : Extraction d'informations implicites

Data : Liste évaluations, Liste catégorie par service
Result : Profil utilisateur
for *each* *Review* **do**
  Récupérer identifiant utilisateur U;
  Récupérer identifiant service S;
  Ajouter cette catégorie aux catégories auxquelles s'intéressent U;
  Mettre à jour le nombre d'évaluation par catégorie;
  Calculer moyenne des évaluations d'un utilisateur;
end

2.6.2 Module du filtrage collaboratif

Algorithme PMF

L'algorithme PMF (Probabilistic Matrix factorisation) est un algorithme de FC basé modèle qui utilise la factorisation matricielle, ce dernier est un simple modèle probabiliste linéaire avec observation de bruit Gaussien.[15]

Considérons une matrice d'usage R de taille $N \times M$ pour N utilisateurs et M items. L'algorithme prend cette matrice en entrée et produit en sortie deux matrices : une matrice U de taille $D \times N$ et une autre V de taille $D \times M$ tel que $U^T V \approx R$ L'algorithme minimise la valeur de l'expression suivante en utilisant une technique de machine learning appelée « Gradient descent » :

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \underbrace{\frac{\lambda_U}{2}\sum_{i=1}^{N}||U_i||_{Fro}^2}_{\text{terme de regularisation}} + \underbrace{\frac{\lambda_V}{2}\sum_{i=1}^{M}||U_i||_{Fro}^2}_{\text{terme de regularisation}} \quad (2.1)$$

Notations :
— E : Erreur à minimiser.

— I_{ij} : coefficient égale à 1 si R_{ij} est observé, sinon $I_{ij} = 0$. Cela permet d'ignorer les valeurs non-observées.

Remarque : Le terme de régularisation : permet d'éviter le problème d' « overfitting » (sur-apprentissage).

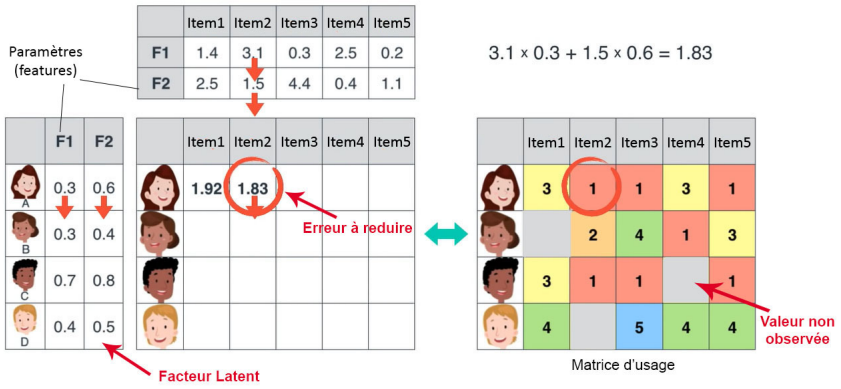

FIGURE 2.12 – Illustration de la factorisation par apprentissage automatique

Suite à la factorisation, nous obtenons deux matrices qui sont le modèle du FC. Ainsi, pour prédire une note non-observée d'un utilisateur i à un item j, il suffit d'effectuer le produit scalaire $U_i^T V_j$ comme le décrit la figure suivante :

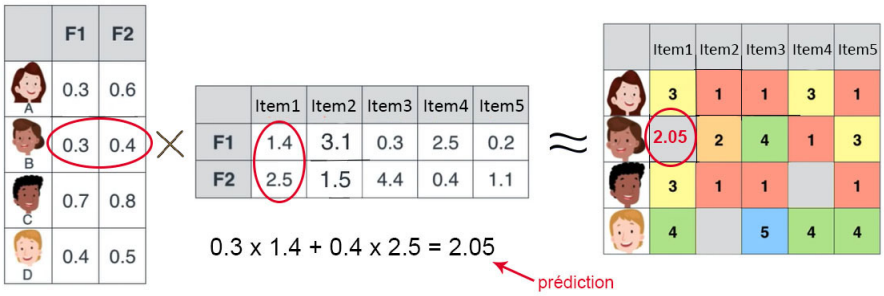

FIGURE 2.13 – Illustration de la phase de prédiction

Algorithme à base de réseaux de neurones

Bien que la technique soit différente, cette approche (FC-RN) est similaire à celle de factorisation matricielle car on cherche toujours à associer chaque utilisateur et item à un vecteur tel que la prédiction du modèle entrainé se rapproche le plus possible de la note attribuée par l'utilisateur à l'item. La figure suivante illustre l'architecture du réseau de neurones que nous avons développé :

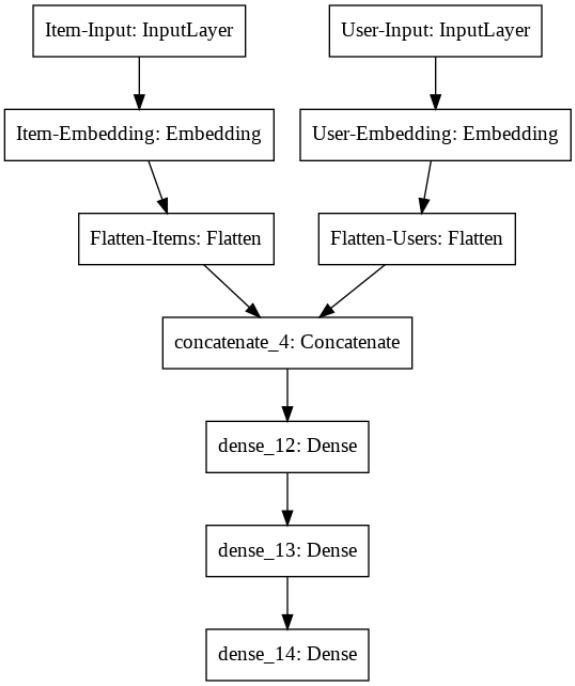

FIGURE 2.14 – Architecture du réseau de neurones

Explication :
— Input Layer : Cette couche représente les données en entrée, à savoir les identifiants des items et des utilisateurs.
— Embedding Layer : Cette couche fait correspondre chaque utilisateur/item à son vecteur.
— Faltten Layer : Comme son nom l'indique, cette couche réduit la dimension de la couche précédente.
— Concatenate Layer : Cette couche concatène les deux couches précédentes.
— Dense Layer : Cette couche est un ensemble de neurones où chaque neurone est connecté à tous les neurones de la couche précédente.

Algorithme de réseaux de neurones basé sur le TAL

Dans les techniques précédentes, on ne donne à l'algorithme en entrée que les évaluations des utilisateurs, et on essaye de créer des vecteurs pour représenter chaque utilisateur et chaque item à partir de ces évaluations.

L'approche à base de TAL (FC-RN-TAL) consiste à associer un vecteur à chaque item avant même d'avoir commencé l'entrainement, et cela grâce à la technique de Doc2vec (*cf.* section 2.3.5). Nous concaténons un certain nombre de commentaires par item pour former un document texte contenant les avis des utilisateurs. Ensuite, grâce à Doc2Vec nous générons le vecteur correspondant à ces documents textes. Enfin, chaque item est associé au vecteur résultant du traitement de ces commentaires.

On rappelle que la technique Doc2Vec permet d'encoder des informations sémantiques concernant un texte. Ainsi, si par exemple les commentaires d'un restaurant se plaignent de la lenteur de service, alors cette information sémantique sera encodée dans le vecteur. Généralement, on peut s'attendre à ce que les services similaires aient des vecteurs similaires. Il est à noter aussi que les vecteurs des items sont immuables durant l'entrainement, *i.e.* seuls les vecteurs des utilisateurs changeront après

l'initialisation. La figure ci-dessous illustre cette explication en présentant une comparaison entre FC-RN et FC-RN-TAL :

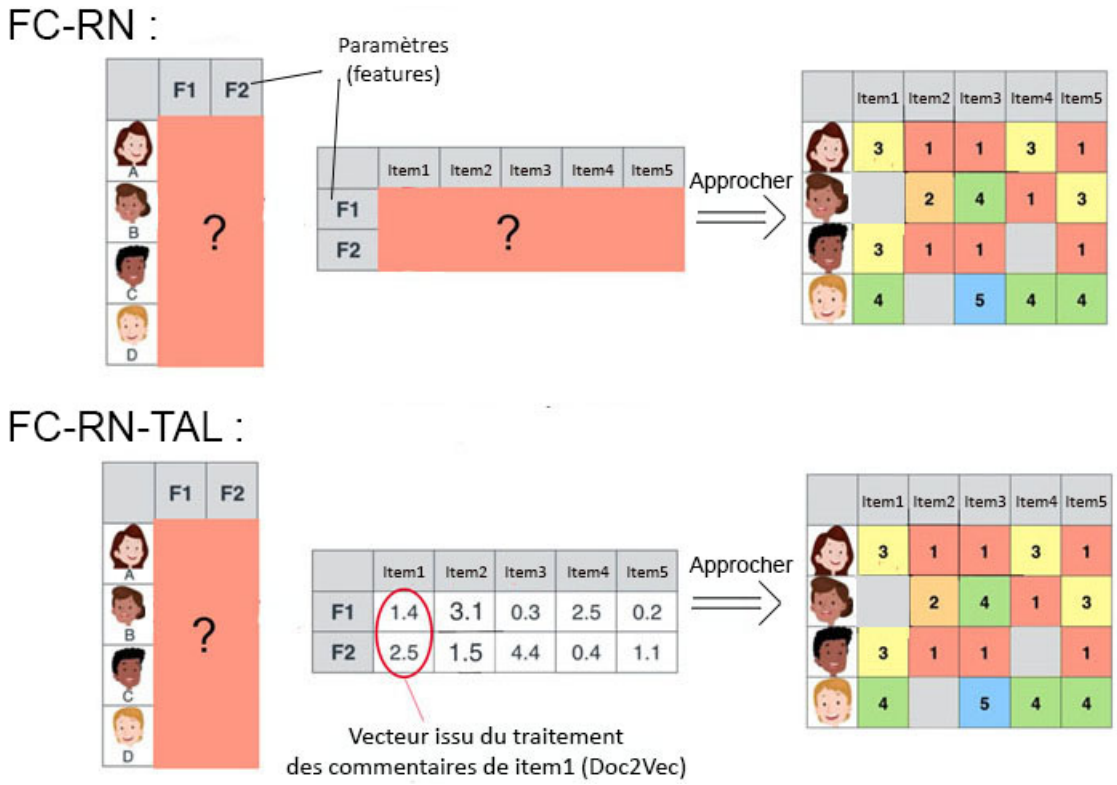

FIGURE 2.15 – Comparaison des algorithmes FC-RN et FC-RN-TAL

2.6.3 Module du filtrage sémantique

Pour le Filtrage sémantique (FSem), notre approche consiste à calculer la similarité sémantique entre les catégories auxquelles appartiennent les services avec les catégories auxquelles s'intéressent les utilisateurs. La figure suivante illustre le principe de fonctionnement de notre algorithme :

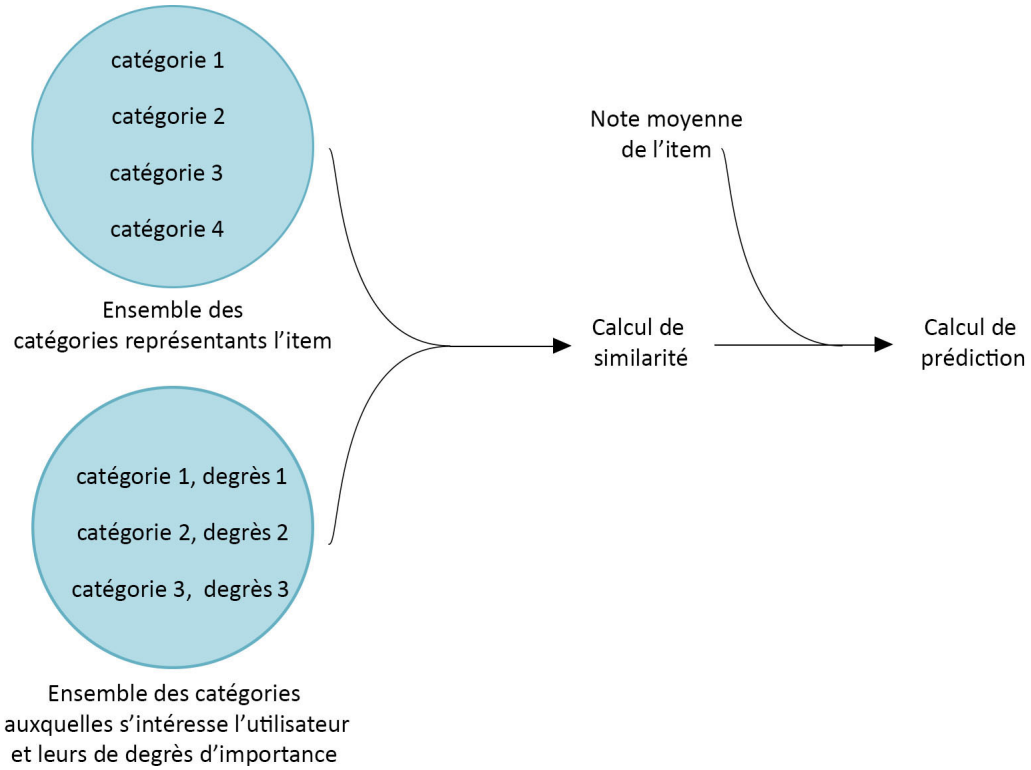

FIGURE 2.16 – Illustration de l'algorithme du filtrage sémantique

Calcul de similarité sémantique

En programmation, on ne peut pas dire si deux chaines de caractères ont une relation sémantique ou pas. Par exemple, « USTHB » et « Université » sont évidemment sémantiquement liés pour un être humain. Néanmoins pour une machine il s'agit de deux variables complètement différentes. Cependant, si ces deux concepts sont modélisés dans une ontologie, il devient possible à une machine de calculer une similarité sémantique entre ces deux concepts. Plusieurs formules de calcul de similarité sémantique ont été proposées. Nous présenterons ci-dessous la formule qui sera utilisée dans notre algorithme de recommandation.

- **Formule de similarité de Wu & Palmer**

  Cette formule [16] repose sur le principe de comptage d'arcs. Elle considère seulement les relations de subsomption et utilise l'ontologie comme une arborescence. Elle permet de calculer la similarité entre deux concepts en calculant le nombre d'arcs les séparant. La formule de Wu et Palmer s'énonce comme suit :

$$Sim_{WP}(C_1, C_2) = \frac{2N}{N_1 + N_2 + 2N} \tag{2.2}$$

  Notations :
  — C_1 et C_2 : sont deux concepts.
  — N_1 et N_2 : La distance séparant C_1 et C_2 de la racine.
  — N : Distance séparant l'ancêtre commun le plus proche de C_1 et C_2 de nœud racine.

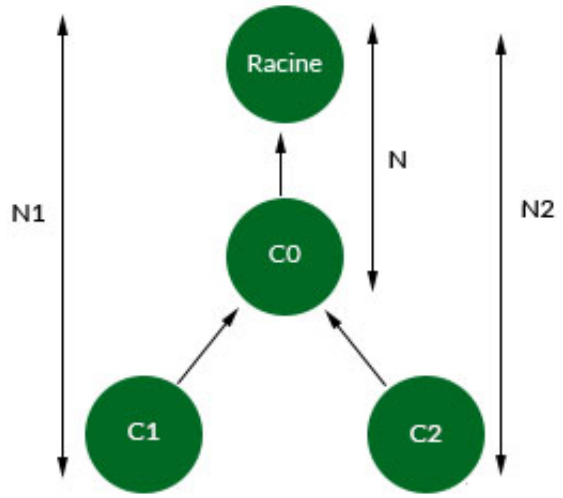

FIGURE 2.17 – Calcul de similarité sémantique par Wu et Palmer (1994)

Puis, pour calculer les similarités entre deux ensembles de catégories, (*i.e.* les catégories des services et les catégories auxquelles s'intéressent chaque utilisateur) nous utilisons la formule suivante :

$$Sim\_sem(u,i) = \frac{\sum_{y \in Y} max(Sim_{wp}(x,y)_{x \in X}) * n_{ux}}{\sum n_{ux}} \tag{2.3}$$

Notations :
— u : Un utilisateur.
— i : Un item.
— X : Ensemble des catégories auxquels s'intéresse u
— Y : Ensemble des catégories auxquelles appartient i
— n_{ux} : Le nombre de fois que l'utilisateur u a évalué la catégorie x (degrés d'importance)

Explication :
Pour chaque concept $x \in X$ on trouve le concept le plus similaire $y \in Y$ et on le multiplie par le degré d'importance de x. Enfin, on calcule la moyenne en divisant par la somme des poids.

Calcul de prédiction

Pour calculer la prédiction, nous avons pensé à la formule suivante :

$$prediction = Sim\_sem(u,i) * note\_moyenne(i) \tag{2.4}$$

Car la note attribuée à un item dépend de deux facteurs :
— L'intérêt porté à l'item : On considère qu'une grande similarité sémantique implique un grand intérêt porté à l'item.
— La qualité de l'item : La note moyenne attribuée reflète la qualité de l'item (satisfaction des utilisateurs).

2.6.4 Module du filtrage social

Notre approche repose sur le calcul de similarité sociale entre les utilisateurs selon leurs degrés d'amitié.

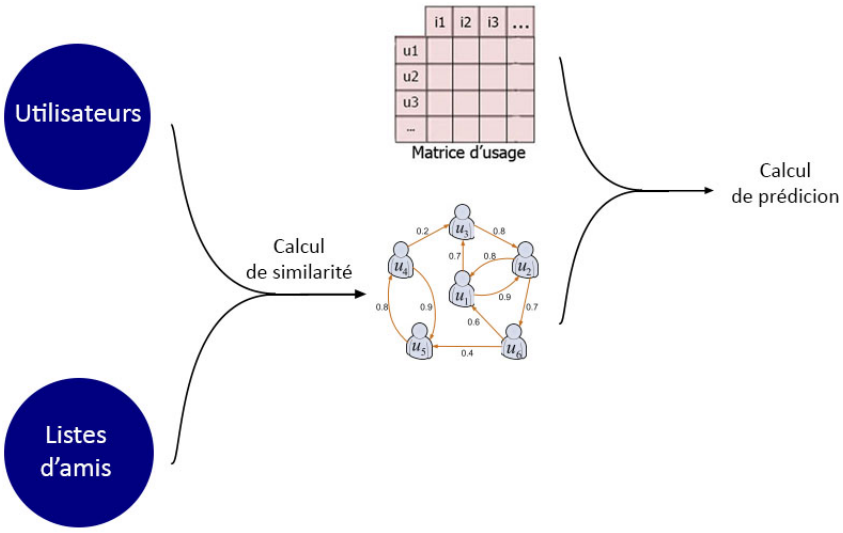

FIGURE 2.18 – Illustration de l'algorithme du filtrage social

Calcul de similarité sociale :

Afin de déterminer la similarité entre les utilisateurs, nous avons utilisé la formule de Jaccard qui s'énonce comme suit :

$$Amitié(u,v) = \frac{|F_u \bigcap F_v|}{|F_u \bigcup F_v|} = \frac{|F_u \bigcap F_v|}{|F_u| + |F_v| - |F_u \bigcap F_v|} \tag{2.5}$$

Notations :
— u et v : Deux utilisateurs.
— F_u et F_v : La liste d'amis de u et v respectivement.

Explication : La similarité sociale entre chaque deux utilisateurs est égale au nombre d'amis en communs qu'ils ont sur leur nombre d'amis distincts. Cette formule a pour propriété d'être égale à 0 si les deux utilisateurs n'ont aucun ami en commun et 1 s'ils sont parfaitement similaires (s'ils ont la même liste d'amis).

Calcul des prédictions :

Afin de calculer les prédictions, nous avons utilisé la formule suivante :

$$prediction(u,i) = \overline{r(u)} + \frac{\sum_{u_j \in U} Amitié(u,u_j).(r_{u_j,i} - \overline{r(u_j)})}{\sum_{u_j \in U} Amitié(u,u_j)} \tag{2.6}$$

Notations :
— u : Un utilisateur.
— i : Un item.
— U : Ensemble des utilisateurs.

— u_j : Utilisateur différent de u qui a une similarité différente de 0 avec ce dernier.
— $\overline{r(u)}$ et $\overline{r(u_j)}$: Les moyenne des évaluations que les utilisateurs u et u_j ont fait respectivement.
— $r_{u_j,i}$: Évaluation de l'utilisateur u_j sur l'item i.

Explication : Une note est prédite selon les utilisateurs ayant un degré de similarité avec u et ayant déjà noté l'item i. En effet, la différence entre la moyenne des évaluations d'un utilisateur et la note qu'il attribue à i est considérée comme l'avis de cet utilisateur, dans notre formule l'avis de chaque utilisateur u_j est pondéré selon son degré de similarité avec u et sera divisé par le nombre total des similarités de u cette valeur sera ajoutée à la moyenne des évaluations qu'il a faites pour avoir la note qu'il attribuera à i.

Algorithme 2 : Algorithme de prédiction

Data : Matrice d'usage, liste des utilisateurs similaires u_j et leur degré de similarité.
Result : Note que l'utilisateur u donnera à l'item i
initialization;
for *each* u_j **do**
  if u_j *a évalué* i **then**
    Avis = Matrice d'usage[u_j ,i] - Moyenne(Evaluations faites par u_j)
    Somme avis = Somme avis+Amitié(u,u_j) * Avis
    Somme amitié = Somme amitié + Amitié(u,u_j)
  else
  end
  prédiction = Moyenne(Evaluations faites par u) + Somme avis/Somme amitié
end
Return prédiction

2.6.5 Module d'hybridation

Le FC présente certaines limitations qui peuvent être réduite grâce aux algorithmes basés contenus (Fsem et Fsoc), notamment le problème du démarrage à froid expliqué précédemment. D'autre part, il est possible de palier à l'effet entonnoir du filtrage sémantique grâce au filtrage collaboratif, c'est pour cela que nous avons opté pour une hybridation entre les deux types de filtrage afin de minimiser les limitations de chaque algorithme.

Ceci dit, il existe différentes manières d'effectuer une hybridation. Pour notre projet, nous nous sommes intéressés aux deux combinaisons suivantes :
— Hybridation pondérée.
— Hybridation en cascade.

Hybridation pondérée

Un système de recommandation hybride pondéré calcule le score d'un item à partir des résultats de toutes les techniques de recommandation disponibles dans le système en utilisant, par exemple, une formule linéaire. Dans notre cas, nous utilisons la formule suivante pour effectuer des combinaisons entre les différents algorithmes développés :

$$\hat{r}_{ui} = (1 - \alpha)x_{ui} + \alpha * y_{ui} \tag{2.7}$$

Notations

— $\hat{r}_{ui}$: Prédiction de la note de l'utilisateur u pour l'item i.
— x_{ui} : Prédiction de la note de l'utilisateur u pour l'item i issu d'un algorithme.
— y_{ui} : Prédiction de la note de l'utilisateur u pour l'item i issu d'un autre algorithme.
— α : Degré d'importance où $0.1 \leq \alpha \leq 0.9$

Hybridation en cascade

Après l'obtention des prédictions issues des différents algorithmes précédents et de leurs hybridations, nous raffinerons les résultats obtenus grâce à l'algorithme de localisation que nous expliquons dans la section suivante.

2.6.6 Module de localisation

Dans notre contexte, nous recommandons des services, l'utilisateur sera donc obligé de se déplacer physiquement vers ce service, (ou que le service se déplace vers le client s'il s'agit d'un service à domicile). C'est pour cela que la distance séparant le client du service est un élément crucial dans notre système de recommandation. Pour cela, nous avons raffiné les résultats des items recommandés en considérant la distance séparant le client du service. Ce calcul sera effectué en cascade après la recommandation hybride. Pour calculer les distances entre un utilisateur et un service, nous avons utilisé la formule de Haversine qui s'énonce comme suit :

$$haversine\left(\frac{d}{r}\right) = haversine(\phi_2 - \phi_1) + cos(\phi_1).cos(\phi_2).haversine(\lambda_2 - \lambda_1) \tag{2.8}$$

Notations :

— $haversine(\theta) = sin^2\left(\frac{\theta}{2}\right) = \frac{1-cos(\theta)}{2}$
— d est la distance entre deux points.
— r est le rayon de la sphère.
— ϕ_1, ϕ_2 sont respectivement les latitudes des points 1 et 2.
— λ_1, λ_2 sont respectivement les longitudes des points 1 et 2.

d est alors donné par :

$$d = 2.r.arcsin\left(\sqrt{sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + cos(\phi_1).cos(\phi_2).sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right) \tag{2.9}$$

Calcule de prédiction

Puisque ce module de localisation s'exécute en cascade après les algorithmes précédents, pour effectuer la prédiction nous appliquons la formule suivante :

$$prediction = \hat{r}_{uv} * f(dist_{uv}) \tag{2.10}$$

tel que :

$$f(x) = \frac{2}{e^{0.03x} + e^{-0.03x}} \tag{2.11}$$

Notation
— u : Un utilisateur.
— v : Un item

— $\hat{r}_{uv}$: La prédiction de l'évaluation de u sur v issu de l'algorithme précédent.
— $dist_{uv}$: Distance en Kilomètres entre l'utilisateur et le service.

Explication : Voici la courbe de la fonction $f(x)$:

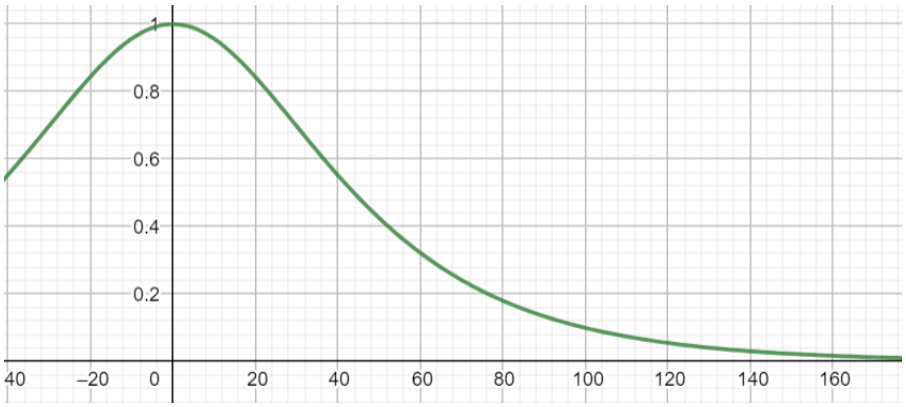

FIGURE 2.19 – Fonction de localisation f(x)

Plus la distance en Kilomètres est grande, plus la prédiction sera pénalisée.

2.7 Conclusion

Dans ce chapitre, nous avons, dans un premier temps, modélisé le système à l'aide du langage de modélisation UML afin de décrire l'aspect fonctionnel de notre application. Dans un second temps, nous avons proposé une approche pour la recommandation des services de notre application. Dans le prochain chapitre, nous présenterons la réalisation de notre application et du système de recommandation qui sera basé sur l'approche proposée.

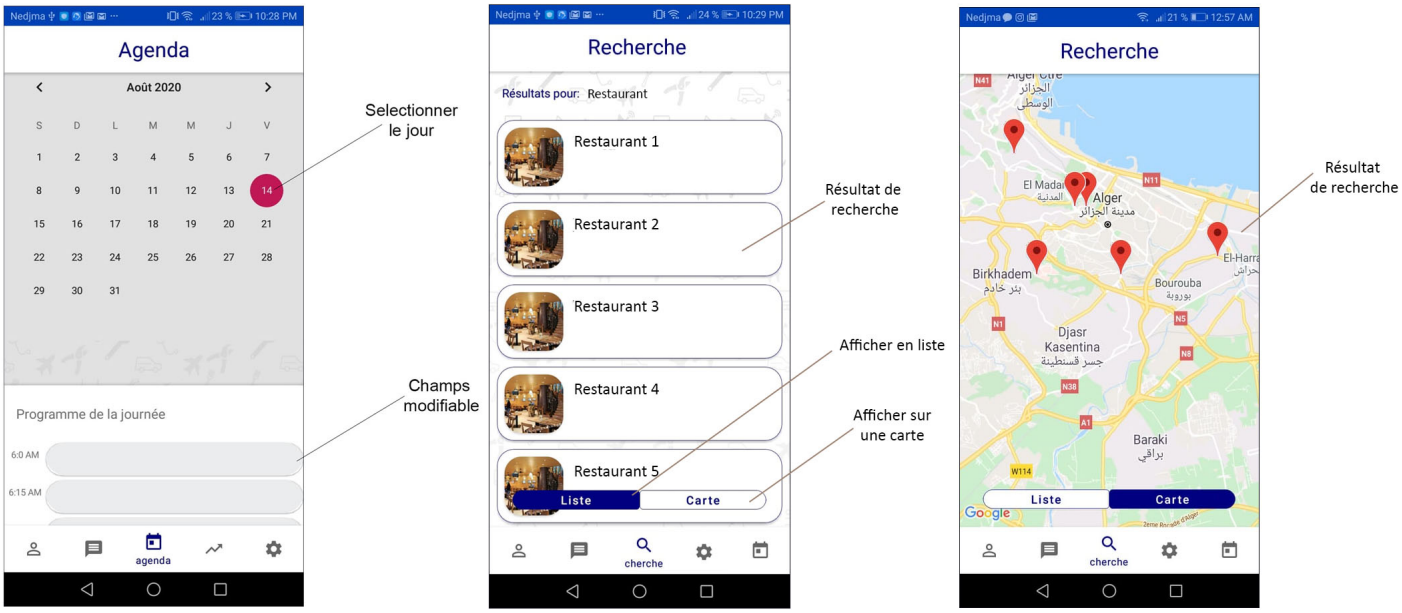

FIGURE 3.4 – Interfaces d'agenda et gestion de RDV, de résultats de recherche par liste et par carte respectivement.

3.3 Implémentation du système de recommandation

3.3.1 Langages de programmation

Nous présentons dans cette section tous les langages qui nous ont été nécessaires au développement de notre système de recommandation.

Python Python est un langage de programmation interprété, orienté objet et de haut niveau avec une sémantique dynamique. Ses structures de données intégrées de haut niveau, associées à un typage dynamique et à une liaison dynamique, le rendent très attractif pour le développement rapide d'applications, ainsi que pour une utilisation en tant que langage de script ou de collage pour connecter des composants existants entre eux. [25]

JAVA Java est un langage de programmation généraliste basé sur les classes, orienté objet et conçu pour avoir le moins de dépendances d'implémentation possible. [18]

3.3.2 Outils de programmation

Nous présentons dans cette section tous les outils qui nous ont été nécessaires au développement de notre système de recommandation.

Gensim Gensim est une bibliothèque Python pour la modélisation de sujets, l'indexation de documents et la recherche de similitudes avec de grands corpus. Le public cible est la communauté du traitement du langage naturel et de la recherche d'informations (RI).[26]

Eclipse Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.[27]

Librec La bibliothèque LibRec implémente une suite d'algorithmes de recommandation de pointe ainsi que les méthodes traditionnelles. En outre, une série de mesures d'évaluation sont mises en œuvre, y compris des mesures basées sur la diversité qui sont rarement activées dans d'autres bibliothèques. [28]

Colaboratory Colaboratory, ou «Colab» en abrégé, est un produit de Google Research. Colab permet à quiconque d'écrire et d'exécuter du code Python arbitraire via le navigateur. Colab est particulièrement bien adapté à l'apprentissage automatique, à l'analyse de données et à l'éducation. [29]

TensorFlow TensorFlow est une plate-forme Open Source de bout en bout dédiée au machine learning. Elle propose un écosystème complet et flexible d'outils, de bibliothèques et de ressources communautaires permettant aux chercheurs d'avancer dans le domaine du machine learning, et aux développeurs de créer et de déployer facilement des applications qui exploitent cette technologie.[30]

Owlready2 Owlready2 est un module de manipulation des ontologies OWL 2.0 en Python. Il peut créer, charger, modifier et enregistrer des ontologies et prendre en charge le raisonnement via HermiT (inclus). Owlready permet un accès transparent aux ontologies OWL. [31]

Protégé Protégé est un système pour la création d'ontologies. Il a été créé à l'université Stanford et est très populaire dans le domaine du Web sémantique et au niveau de la recherche en informatique.[32]

JSON JavaScript Object Notation (JSON) est un format de données textuelles dérivé de la notation des objets du langage JavaScript [2] . Il permet de représenter de l'information de façon structurée comme le permet XML par exemple.

3.3.3 Architecture technique de notre système de recommandation

La figure ci-dessous montre l'interaction entre les différents outils utilisés pour le développement de l'application.

2. Langage de programmation de scripts

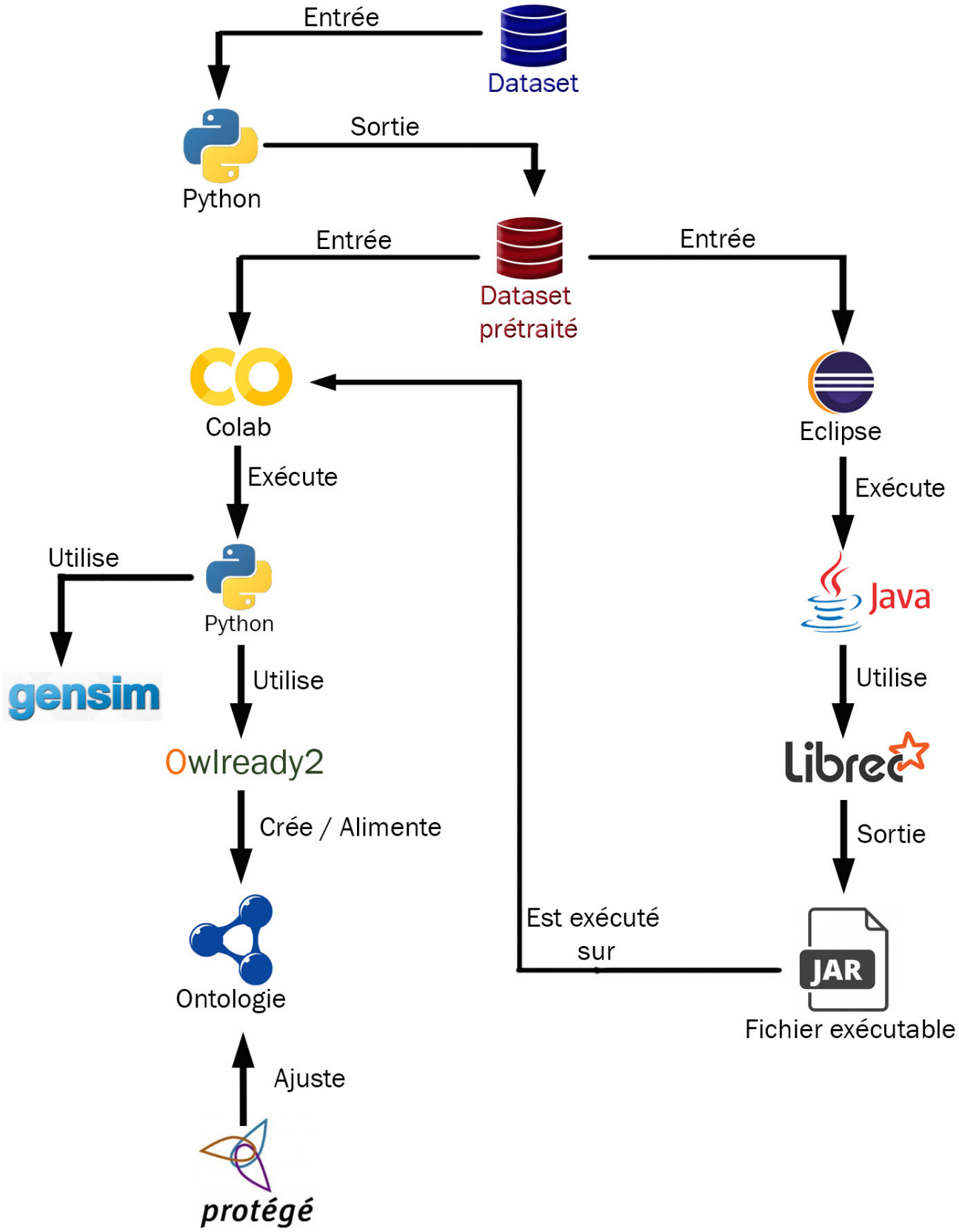

FIGURE 3.5 – Architecture technique du système de recommandation

3.3.4 Description du Système de recommandation

Pour avoir une vue d'ensemble sur notre approche, nous avons développé une application facilitant l'exécution et l'évaluation de nos algorithmes. Dans cette section nous allons présenter l'application ainsi que ses différentes interfaces graphiques.

Menu principal

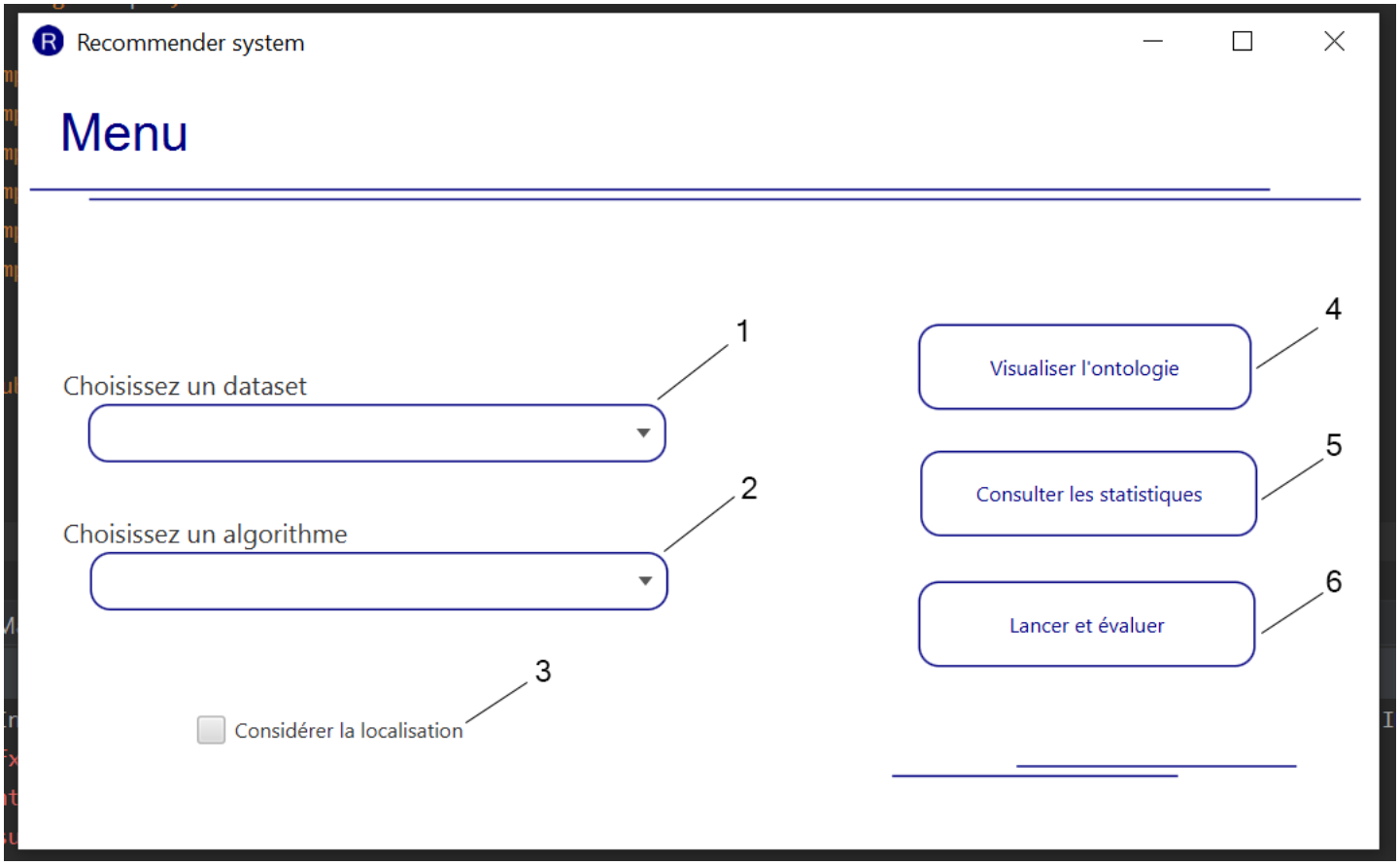

FIGURE 3.6 – Menu principal du système de recommandation

Légende :

- 1 : Menu déroulant permettant de choisir un dataset à utiliser dans nos évaluations.
- 2 : Menu déroulant permettant de choisir un algorithme de recommandation.
- 3 : Check-box « Considérer la localisation » permet la considération ou non de l'aspect de distance entre le client et le service.
- 4 : Bouton permettant de visualiser l'ontologie du dataset sélectionné en 1.
- 5 : Bouton permettant de Consulter les statistiques du dataset sélectionné en 1.
- 6 : Bouton permettant l'évaluation des paramètres précisés en 1, 2 et 3.

Visualisation de l'ontologie

Après avoir choisi un dataset et avoir cliqué sur « Visualiser l'ontologie », une page web interactive s'ouvre automatiquement permettant de visualiser l'ontologie. Il s'agit d'un diagramme permettant de comprendre la hiérarchie de l'ontologie, tel que chaque classe est représentée par un cercle, et chaque classe mère contient ses classes filles.

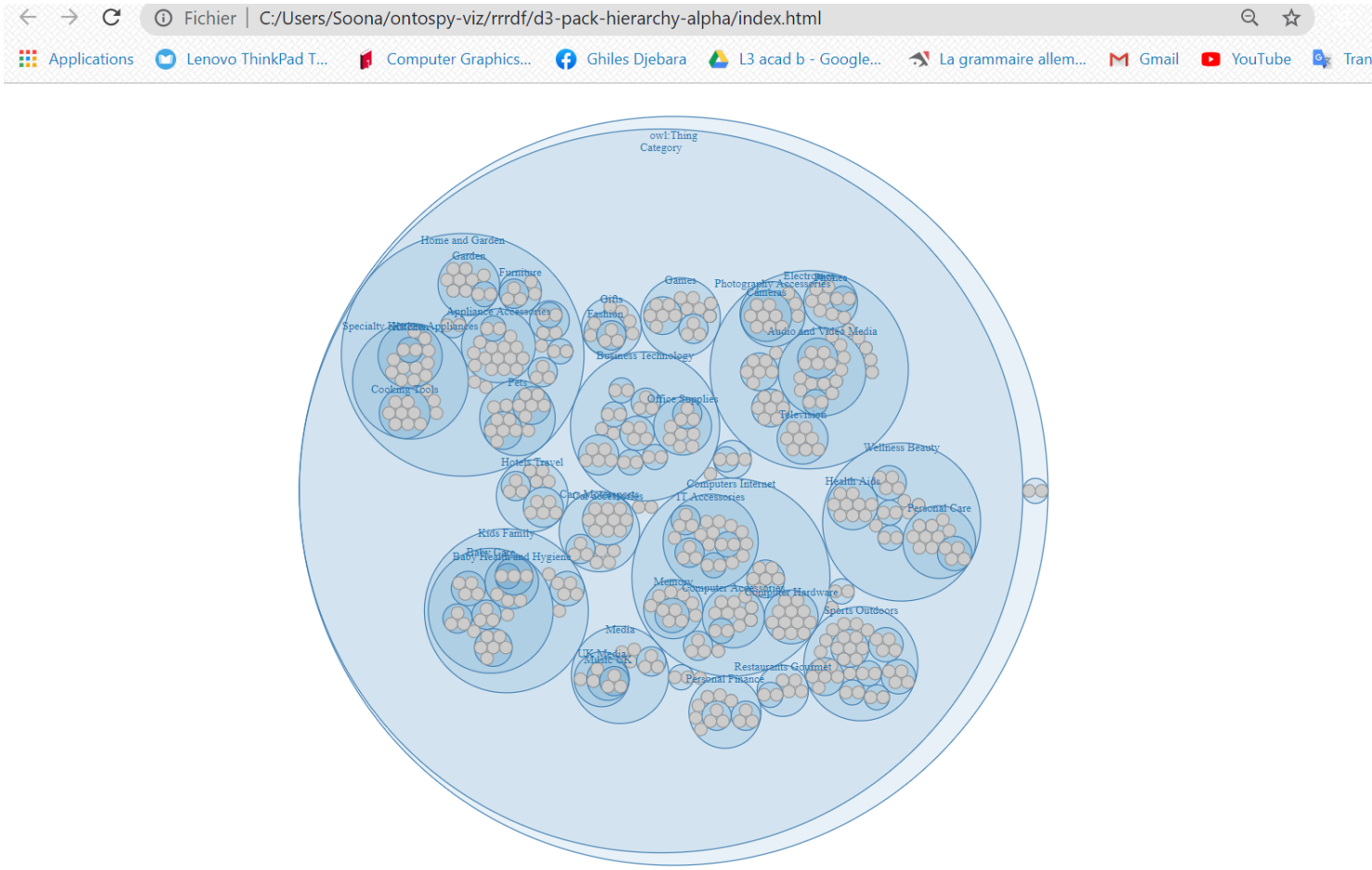

FIGURE 3.7 – Visualisation de l'ontologie

Interface d'évaluation

Cette interface permet de voir les traces d'exécution des algorithmes ainsi que les résultats des évaluations. *voir FIGURE 3.8.*

FIGURE 3.8 – Interface graphique de l'évaluation

3.3.5 Datasets

Dans cette section, nous allons présenter les différentes bases de données que nous avons utilisé pour évaluer les algorithmes que nous avons développés.

1 - Yelp Dataset

Yelp est une multinationale qui développe, héberge et commercialise Yelp.com et l'application mobile Yelp qui publient des avis participatifs sur les commerces locaux.[33]

L'ensemble de données Yelp est un sous-ensemble des entreprises, avis et données utilisateur que stocke Yelp. Elles sont disponibles sous forme de fichiers JSON accessibles au public pour une utilisation à des fins académiques. [34]

Description des données

Les données Yelp sont répartis dans plusieurs fichiers sous format JSON, tel que chaque fichier est composé d'un seul type d'objet ; un objet JSON par ligne. Parmi ces fichiers, nous en avons exploité quatre : user.json, business.json, review.json et categories.json.

- **user.json :** Ce fichier contient des informations sur les différents utilisateurs, parmi les attributs les plus importants décrivant un utilisateur on retrouve :

  - **user_id :** Une chaine de caractères unique identifiant l'utilisateur.
  - **review_count :** Le nombre d'évaluations effectuées par l'utilisateur.
- **business.json :** Ce fichier contient des information sur les différents commerces, parmi les attributs les plus importants décrivant un commerce on retrouve :
  - **business_id :** Une chaine de caractères unique identifiant le commerce.
  - **review_count :** Le nombre d'évaluations sur le commerce.
  - **latitude**
  - **longitude**
  - **stars** : Moyenne des évaluations des utilisateurs sur le commerce.
  - **categories** : Ensemble des catégories auxquelles appartient le commerce.
- **review.json :** Ce fichier contient les évaluations effectuées. Parmi les attributs les plus importants décrivant une évaluation on retrouve :
  - **user_id :** Identifiant de l'utilisateur.
  - **business_id :** Identifiant du commerce.
  - **stars :** Note attribuée par *user_id* à *business_id*.
- **categories.json :** Ce fichier contient la hiérarchie des catégories. Parmi les attributs les plus importants décrivant une catégorie on retrouve :
  - **alias :** Identifiant de la catégorie.
  - **title :** Nom de la catégorie.
  - **parent :** Identifiant du parent de la catégorie (catégorie mère).

Statistiques

Ci-dessous, quelques statistiques sur la base de données Yelp.

	Nb users	Nb Items	Nb evaluations	Nb Categories	Densité
Yelp	1,968,703	209,393	8,021,122	1580	0.0019%

TABLE 3.1 – Statistique du dataset Yelp

Remarque : La densité se calcule comme suit :

$$densite = \frac{NBe}{NBi * NBu}$$

Notations :
- NBe : Nombre d'évaluations.
- NBu : Nombre d'utilisateurs.
- NBi : Nombre d'items.

2 - MovieLens dataset

MovieLens est un système de recommandation en ligne permettant de suggérer des films à des utilisateurs selon leurs profils.

Description des données

La base de données s'organise sous plusieurs fichiers sous format CSV (Coma Separated Values). Parmi les fichiers on retrouve notamment :

— **u.user :** Contient les identifiants des utilisateurs.
— **u.item :** Contient les identifiants des items.
— **u.data :** Contient les différentes évaluations sous format « **id_user id_item note** ».

Statistiques

Voici quelques statistiques sur la base de données MovieLens :

	Nb users	Nb items	Nb évaluations	Densité
MovieLens	600	9000	100,000	1.9%

TABLE 3.2 – Statistique du dataset MovieLens

Comparaison des datasets

Le tableau suivant résume les atouts des différentes bases de données dans le but de les comparer.

Base de données	Évaluations	Catégories des items en hierarchie	Information sociale	Commentaires
Yelp	Oui	Oui	Oui	Oui
MovieLens	Oui	Non	Oui	Non

TABLE 3.3 – Comparaison des bases de données

On remarque que les différentes bases de données n'offrent pas les mêmes types d'information. Pour cela, l'évaluation des algorithmes se fera sur les bases de données qui présentent les informations nécessaires. Nous notons que Yelp dispose de toutes les informations nécessaires à l'évaluation de nos algorithmes tandis que la base MovieLens n'inclut pas toutes ces informations. Cependant, elle est la base la plus utilisée dans le domaine de la recommandation, c'est pour cela que nous avons jugé intéressant de l'exploiter pour évaluer nos algorithmes, notamment les algorithmes de FC.

Nous notons que « Catégories des items en hiérarchie » signifie si les catégories figurant dans la base de données peuvent être modélisées en ontologie avec assez de profondeur.

3.3.6 Prétraitements

Nous avons effectué un ensemble de tâches de prétraitements sur les bases de données afin de :

— Supprimer les incohérences : Par exemple deux items avec le même identifiant mais avec des valeurs différentes pour les autres attributs.
— Garder les informations les plus importantes *i.e.* supprimer les utilisateurs ayant effectué un nombre d'évaluations inférieur à 15 (car nous pensons que c'est le seuil minimal pour considérer un utilisateur comme « engagé »).

— Traiter les informations manquantes : par exemple un attribut manquant d'un item.
— Supprimer les doublons.
Voici les statistiques des deux bases de données après ces prétraitements :

	Nb users	Nb items	Nb catégories	Nb évaluations	Densité
Yelp	7,456	32,596	1,088	826,753	0.34%
MovieLens	943	1,682		100,000	6.30%

TABLE 3.4 – Statistiques sur les bases de données prétraitées

Nous avons aussi généré plusieurs échantillons à partir de ces deux bases de données en faisant varier le nombre d'utilisateurs afin de tester les comportements de nos algorithmes avec l'augmentation de la taille de l'échantillon de données. Voici les échantillons générés :
— **E1** : Base de données avec 25% des utilisateurs, en considérant les items et les évaluations effectuées par ces utilisateurs.
— **E2** : Base de données avec 50% des utilisateurs, en considérant les items et les évaluations effectuées par ces utilisateurs.
— **E3** : Base de données avec 75% des utilisateurs, en considérant les items et les évaluations effectuées par ces utilisateurs.
— **E4** : Base de données avec 100% des utilisateurs, *i.e.* tous les utilisateurs, items et évaluations correspondantes.

3.3.7 Métriques d'évaluations

Dans cette section, nous allons décrire les métriques que nous avons utilisées pour évaluer nos algorithmes.

Remarque : Les algorithmes qui reposent sur un apprentissage automatique ont été entrainés sur 80% des donnés et évalués sur les 20% restants.

Mean absolute error (MAE)

MAE est la moyenne sur l'échantillon de vérification des valeurs absolues des différences entre la prévision et l'observation correspondante. Le MAE est un score linéaire, ce qui signifie que toutes les différences individuelles sont pondérées de manière égale dans la moyenne.[35] Ci-dessous, la formule de MAE :

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|r_i - \hat{r}_i| \tag{3.1}$$

Root mean square error (RMSE)

Le RMSE est une règle de notation quadratique, la différence entre les prévisions et les valeurs observées correspondantes est chacune mise au carré puis moyennée sur l'échantillon. Enfin, la racine carrée de la moyenne est prise. Puisque les erreurs sont mises au carré avant de faire la moyenne, le RMSE donne un poids relativement élevé aux erreurs importantes. Cela signifie que le RMSE

est le plus utile lorsque des erreurs importantes sont particulièrement indésirables.[35] Ci-dessous, la formule de RMSE :

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(r_i - \hat{r}_i)^2} \tag{3.2}$$

3.3.8 Évaluation du filtrage collaboratif

Dans cette section, nous allons évaluer les algorithmes PMF (FC-PMF) et l'algorithme à base de réseau de neurone (FC-RN) pour ensuite discuter les résultats obtenus.

1. FC-PMF

Le tableau ci-dessous résume les évaluations de PMF sur les différents échantillons des bases de données Yelp et MovieLens :

Algorithme	Base de données	Métriques	E1	E2	E3	E4
FC-PMF	Yelp	MAE	0.92	0.92	0.87	0.84
		RMSE	1.17	1.17	1.11	1.07
	MovieLens	MAE	0.75	0.74	0.73	0.72
		RMSE	0.95	0.95	0.93	0.91

TABLE 3.5 – Évaluation du FC-PMF

Discussion des résultats

On remarque d'abord que les résultats obtenus en utilisant la base de données MovieLens sont meilleurs que ceux obtenus avec Yelp. On peut expliquer cela par la densité de MovieLens qui est nettement supérieure à celle de Yelp.
On remarque aussi une différence entre les performances avec les échantillons E1, E2, E3 et E4 où les résultats sont de plus en plus meilleurs. On explique cela par le fait qu'il y ait de plus en plus d'utilisateurs dans ces bases de données respectivement (*cf.* section 3.3.6). Plus il y a d'utilisateurs, plus il y a d'évaluations et plus la densité de l'échantillon augmente. Ce qui explique la différence de performance.

2. FC-RN

Nous avons évalué le FC-RN de la même manière que le FC-PMF. Voici un tableau qui résume les résultats obtenus :

Algorithme	Base de données	Métriques	E1	E2	E3	E4
FC-RN	Yelp	MAE	0.82	0.79	0.77	0.77
		RMSE	1.07	1.02	1	0.99
	MovieLens	MAE	0.74	0.74	0,73	0.73
		RMSE	0.96	0.95	0.94	0.93

TABLE 3.6 – Évaluation du FC-RN

Discussion des résultats

À l'instar du FC-PMF, on remarque de meilleurs résultats avec MovieLens. L'explication réside toujours dans la différence de densité entre les deux bases de données. On remarque aussi une légère sensibilité au nombre d'utilisateurs dans l'échantillon évalué. Néanmoins, la sensibilité est moindre par rapport à celle de FC-PMF.

3.3.9 Évaluation du filtrage sémantique

Dans cette section, nous allons présenter puis discuter les résultats de l'algorithme de filtrage sémantique sur la base de données Yelp, car c'est la seule qui nous permet d'extraire des informations sémantiques. La figure ci-dessous résume les résultats obtenus en fonction du nombre d'utilisateurs.

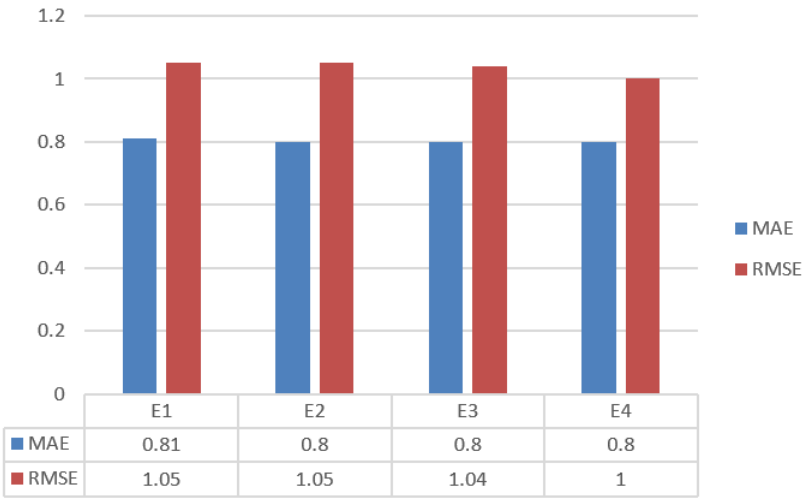

FIGURE 3.9 – Évaluation du filtrage sémantique

Discussion des résultats

On remarque que les résultats sont stables avec l'augmentation du nombre d'utilisateurs. On explique cela par le fait que ce ne soit pas un algorithme d'apprentissage automatique, mais un algorithme basé contenu, où dans ce cas, les résultats sont constants.

3.3.10 Évaluation du filtrage social

Dans cette section, nous allons évaluer les performances du filtrage social. Le tableau suivant résume les résultats obtenus :

Base de données	Métriques	E1	E2	E3	E4
Yelp	MAE	0.82	0.82	0.81	0.81
	RMSE	1.04	1.04	1.04	1.04
MovieLens	MAE	0.8	0.8	0.8	0.79
	RMSE	1	1	1	1

TABLE 3.7 – Évaluations du filtrage social

Discussion des résultats

Tout comme le filtrage sémantique, le filtrage social n'est pas un algorithme d'apprentissage mais un algorithme basé contenu. Il est donc moins sensible aux perturbations liées au nombre d'utilisateurs. C'est pour cela qu'on remarque qu'il présente des résultats constants avec la diminution du nombre d'utilisateurs.

3.3.11 Apport du TAL sur le filtrage collaboratif

Dans cette section, nous allons évaluer l'apport du TAL sur l'algorithme de FC-RN. Le tableau suivant est une comparaison entre les résultats de FC-RN et FC-RN-TAL sur la base de données Yelp.

Algorithme	Métriques	E1	E2	E3	E4
FC-RN	MAE	0.82	0.79	0.77	0.77
	RMSE	1.07	1.02	1	0.99
FC-RN-TAL	MAE	0.77	0.77	0,77	0.77
	RMSE	1	0.99	0.99	0.99

Table 3.8 – Comparaison entre FC-RN et FC-RN-TAL

Discussion des résultats

On remarque que les performances de FC-RN-TAL ne sont pas inversement proportionnelles au nombre d'utilisateurs contrairement à FC-RN, ses performances sont globalement stables et meilleures en fonction du nombre d'utilisateurs. Cela signifie que le TAL contribue grandement à améliorer les performances de l'algorithme FC-RN.

En plus d'améliorer les résultats de FC-RN, le FC-RN-TAL réduit grandement le temps d'apprentissage du modèle. Cela est dû au fait que seulement les vecteurs associés aux utilisateurs sont entrainés, alors que les vecteurs associés aux items sont issus du TAL et sont immuable durant l'apprentissage, comme on l'a expliqué dans le chapitre précédent. Voici un tableau mettant en évidence le temps d'exécution d'une epoch (*i.e.*un parcours complet sur les données d'entrainement) en fonction du nombre de paramètres et de la taille du batch (*i.e.* un sous-ensemble d'une epoch).

	Nombre de paramètres			
	150		300	
	Taille du batch			
	32	100	32	100
	Temps d'exécution d'une epoch			
FC - RN	**7min05s**	**2min25s**	**12min**	**3min42**
FC - RN -TAL	**2min28s**	**40s**	**2min26s**	**45s**

Table 3.9 – Temps d'exécution du FC-RN et FC-RN-TAL

3.3.12 Apport du filtrage social sur le filtrage sémantique

Dans cette section, nous allons évaluer l'apport du filtrage social sur le filtrage sémantique en effectuant une hybridation pondérée entre les deux algorithmes :

$$prediction = \alpha * Fsem + (1 - \alpha) * Fsoc \tag{3.3}$$

La figure ci-dessous résume les résultats obtenus sur la base de données Yelp en fonction de α :

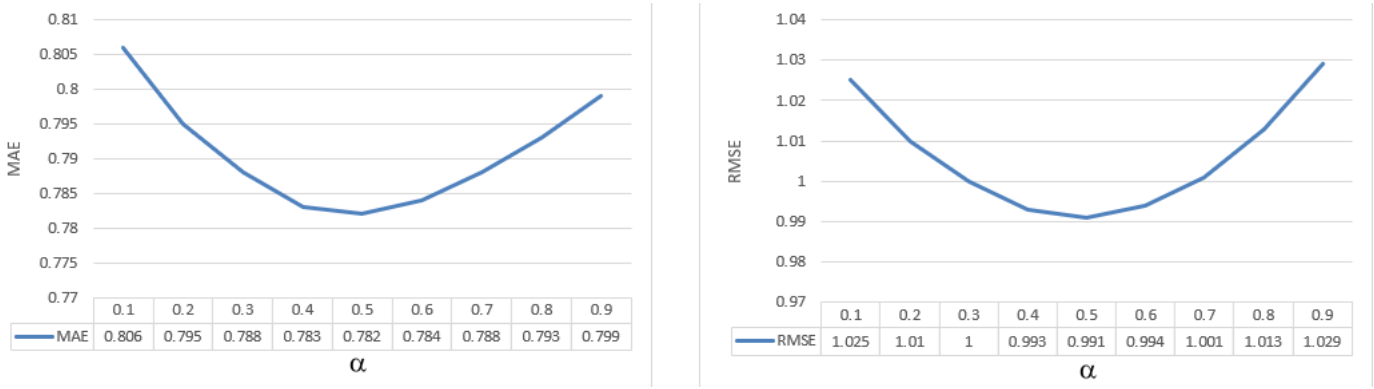

FIGURE 3.10 – Évaluation Fsoc-Fsem

Discussion des résultats

Nous remarquons que plus la valeur de α approche les alentours de 0.5, meilleures sont les métriques MAE est RMSE. Ce qui confirme l'apport du filtrage social sur le filtrage sémantique.

3.3.13 Apport des filtrages sémantique, social et du TAL sur le FC

Pour finir, nous avons évalué une hybridation pondérée entre FC-RN-TAL, le filtrage sémantique et le filtrage social. Pour cela, nous avons utilisé la fonction suivante :

$$prediction = \alpha * Sent + \beta * Sem + \gamma * Soc \tag{3.4}$$

tel que

$$\alpha + \beta + \gamma = 1$$

Notations :

- Sent : Prédiction issue de FC-RN-TAL.
- Sem : Prédiction issue du filtrage sémantique.
- Soc : Prédiction issue du filtrage social.

La figure suivante illustre les résultats de l'évaluation en fonction de α, β et γ sur l'échantillon E4 de la base de données Yelp :

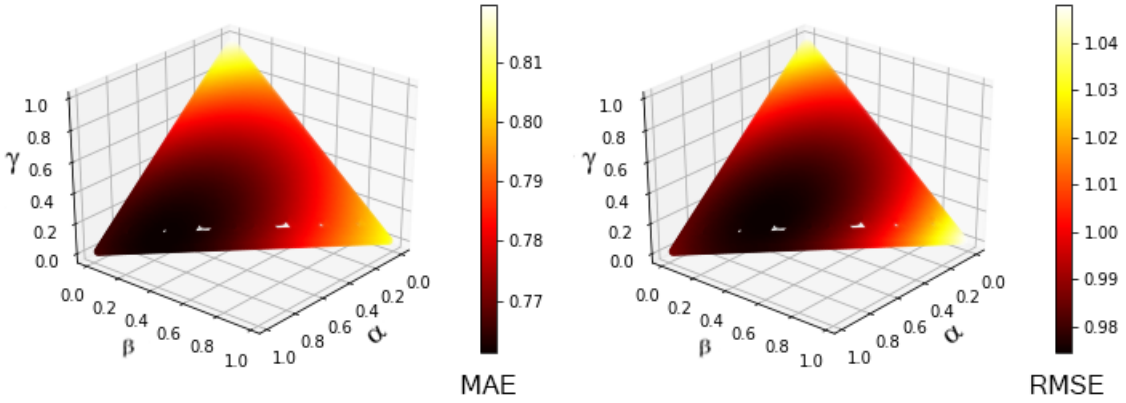

FIGURE 3.11 – Évaluation FC-RN-TAL / Fsoc / Fsem

Le tableau suivant présente quelques résultats obtenus sur l'échantillon E4 de la base de données Yelp :

alpha	beta	gamma	mae	rmse
0.6	**0.26**	**0.14**	**0.762**	**0.974**
0.75	**0.2**	**0.05**	**0.7612**	**0.975**
0.68	0.21	0.11	0.7614	0.975
0.31	0.34	0.35	0.769	0.97

TABLE 3.10 – Quelques résultats de l'évaluations

Le tableau suivant permet de comparer les différents algorithmes évalués sur les différents échantillons de la base de données Yelp :

Algorithme	Base de données	Métriques	E1	E2	E3	E4
PMF	Yelp	MAE	0.92	0.92	0.87	0.84
		RMSE	1.17	1.17	1.11	1.07
FC-RN		MAE	0.82	0.79	0.77	0.77
		RMSE	1.07	1.02	1	0.99
FC-RN-AS		MAE	0.77	0.77	0.77	0.77
		RMSE	1	0.99	0.99	0.99
FC-RN-TAL/FSOC/FSEM		**MAE**	**0.758**	**0.761**	**0.764**	**0.761**
		RMSE	**0.972**	**0.973**	**0.978**	**0.975**

TABLE 3.11 – Comparaison des différents algorithmes

Discussion des résultats

Dans la figure 3.11 et le tableau 3.10, on remarque que pour des valeurs basses à moyenne des poids β et γ et des valeurs fortes du poids de α, on obtient de meilleurs résultats, impliquant que

l'algorithme FC-RN-TAL est important dans la pondération.

Dans le tableau 3.11, on remarque que l'algorithme FC-RN-TAL / FSEM / FSOC présente de bonnes performances quelques soit l'échantillon utilisé. Par exemple, avec l'échantillon E1 on obtient 0.75 pour MAE contre 0.92 avec FC-PMF. On remarque aussi que les résultats obtenus sont meilleurs que tous les autres résultats obtenus par les autres algorithmes confondus. On peut donc conclure que l'algorithme FC-RN-TAL / FSem / FSoc est plus performant que le reste des algorithmes. Ce qui confirme l'apport du TAL, de l'aspect social et de l'aspect sémantique sur le filtrage collaboratif.

3.3.14 Discussion de l'apport de l'aspect de localisation

L'apport de l'algorithme de localisation en cascade ne peut pas être évalué par les métriques MAE et RMSE, pour la simple raison qu'on ne dispose pas des retours d'expérience des utilisateurs en ce qui concerne le rapport entre la qualité et l'intérêt qu'ils portent à un service et la distance les séparant de ce service. Cependant, son apport est intuitif, la considération de l'aspect de localisation permet de réduire les chances qu'un service trop éloigné d'un utilisateur lui soit recommandé inutilement car il ne pourrait pas s'y déplacer. Toutefois, si un service est assez bon (évalué positivement par les autres utilisateurs), alors l'algorithme recommandera ce service à l'utilisateur car il pourrait s'y intéresser en particulier lorsque la nécessité est élevée (exemple un médecin d'une spécialité donnée). Ainsi, un service avec une prédiction de 3/5 à 20 km changera la note à 2.53/5, alors ce service ne sera pas recommandé. Par contre si ce dernier avait une prédiction de 5/5 sa note changera à 4.21/5. Alors il serait recommandé malgré sa distance car il en vaut la peine.

3.4 Implémentation du système de recommandation dans l'application

Le démarrage à froid est le plus grand problème que l'on rencontre lors du déploiement d'un système de recommandation. Car au début, nous ne disposons pas de données pour exécuter les algorithmes. Nous avons pensé à la stratégie suivante pour intégrer notre système de recommandation à notre application mobile en limitant la contrainte du démarrage à froid :

- Recommander les services les mieux évalués en attendant l'enrichissement de la base de données.
- En perspective, améliorer l'application pour obliger les clients à définir une catégorie de service qui est sensible de les intéresser. Cela nous permettrait d'utiliser l'algorithme de filtrage sémantique.
- En perspective, améliorer l'application pour permettre aux utilisateurs clients de créer des liens d'amitiés, Cela nous permettrait d'utiliser l'algorithme de filtrage social.
- En perspective, demander aux clients l'accès à leur liste de contacts (répertoire de contacts de leurs téléphones) afin de pouvoir calculer les liens d'amitiés en se basant sur les numéros de téléphones de chaque client.
- En perspective, effectuer de la récolte de données en ligne (web-scrapping) afin d'acquérir des informations sur les différents services et alimenter l'algorithme FC-RN-TAL.
- Commencer à utiliser le meilleur algorithme obtenu, à savoir (FC-RN-TAL/FSOC/FSEM) avec l'algorithme de localisation en cascade dès qu'on disposera d'assez de données.

Bibliographie

[1] Reinsel & Gantz& John Rydning (2018).The Digitization of the World.

[2] Unified Modeling Language User Guide, 2^{nd} édition.

[3] Burke, R. (2002). Hybrid Recommender Systems : Survey and Experiments. User Modeling and UserAdapted Interaction, 12(4), 331-370.

[4] Gruber, T. (2009). Ontology. Dans L. Liu, & T. M. Özsu, Encyclopedia of Database Systems. SpringerVerlag

[5] Bouzeghoub, M., & Kostadinov, D. (2005). Personnalisation de l'information : Aperçu de l'état de l'art et définition d'un modèle flexible de définition de profils. In Actes de la seconde

[6] Salton, G., Wong, A., & Sang, C. S. (1975). A vector space model for automatic indexing.

[7] Hanani, Uri & Shapira, Bracha & Shoval, Peretz. (2001). Information Filtering : Overview of Issues, Research and Systems. User Model. User-Adapt. Interact.. 11. 203-259. 10.1023/A :1011196000674.

[8] Kumar, M., Yadav, D., Singh, A.K., & Gupta, V.K. (2015). A Movie Recommender System : MOVREC. International Journal of Computer Applications, 124, 7-11.

[9] Sulieman, D. (2014). Systèmes de recommandation sociaux et sémantiques. Université de Cergy

[10] Newman, M. E. (2010). Networks An Introduction. Oxford University Press.

[11] Zhang, J., Guo, G., & Yorke-Smith, N. (2014). Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems. Knowledge-Based Systems, 14-27.

[12] https ://www.investopedia.com/terms/n/neuralnetwork.asp

[13] What is NLP (Natural Language Processing) ? [en ligne], https ://towardsdatascience.com/introduction-to-natural-language-processing-for-text-df845750fb63. page consulté le 20/07/2020

[14] https ://radimrehurek.com/gensim/auto_examples/tutorials/run_doc2vec_lee.html

[15] Andriy Mnih & Ruslan Salakhutdinov, Matrix factorization methods for collaborativ filtering, Université de Toronto

[16] Wu, Z. & Palmer, M. (1994), « Verbs semantics and lexical selection », InProceedings of the 32nd annual meeting on Association for Computational Linguistics, pp. 133-138. Association for Computational Linguistics.

[17] Structured Query Language [en ligne] , https ://fr.wikipedia.org/wiki/Structured_Query_Language , page consultée le 15/07/2020

[18] Java (langage) [en ligne] , https ://fr.wikipedia.org/wiki/Java_(langage) , page consultée le 15/07/2020

[19] Extensible Markup Language (XML) [en ligne] , https ://www.w3.org/XML/ , page consultée le 15/07/2020

[20] What is SQLite ? [en ligne], https ://www.sqlite.org/index.html , page consultée le 15/07/2020

[21] Ratchet [en ligne] , http ://socketo.me/ , page consultée le 15/07/2020

[22] Ratchet, WebSockets for PHP [en ligne] , https ://fr.wikipedia.org/wiki/WebSocket , page consultée le 15/07/2020

[23] What is an API ? [en ligne] , https ://www.redhat.com/en/topics/api/what-are-application-programming-interfaces , page consultée le 15/07/2020.

[24] SQL Injection [en ligne] , https ://www.w3schools.com/sql/sql_injection.asp , page consultée le 15/07/2020

[25] What is Python ? [en ligne] , https ://www.python.org/doc/essays/blurb/ , page consultée le 15/07/2020

[26] Gensim [en ligne] , https ://pypi.org/project/gensim/ page consultée le 15/07/2020

[27] Eclipse (projet) [en ligne] , https ://fr.wikipedia.org/wiki/Eclipse_(projet) , page consultée le 15/07/2020

[28] GUO, Guibing, ZHANG, Jie, SUN, Zhu, et al. LibRec : A Java Library for Recommender Systems. In : UMAP Workshops. 2015.

[29] Qu'est-ce que Colaboratory ? [en ligne] , https ://research.google.com/colaboratory/faq.html , page consultée le 16/07/2020

[30] Why tensorflow, [en ligne],https ://www.tensorflow.org/ ?hl=fr , page consultée le 16/07/2020.

[31] Welcome to Owlready2's documentation [en ligne] , https ://owlready2.readthedocs.io/en/latest/ , page consultée le 16/07/2020

[32] Protégé (logiciel) [en ligne] , https ://fr.wikipedia.org/wiki/Prot%C3%A9g%C3%A9_(logiciel) , page consultée le 16/07/2020

[33] Yelp [en ligne] , https ://fr.wikipedia.org/wiki/Yelp , page consultée le 18/07/2020

[34] Yelp Open Dataset [en ligne] , https ://www.yelp.com/dataset , page consultée le 18/07/2020

[35] Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) [en ligne] , http ://www.eumetrain.org/data/4/451/english/msg/ver_cont_var/uos3/uos3_ko1.htm , page consultée le 18/07/2020