الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدةد. مولاي الطاهر

كلية العلوم

قسم: الإعلام الآلي

Mémoire de Master

Spécialité :Modélisation informatique des connaissances et raisonnement (MICR)

Thème

Systèmes de recommandation basés sur les techniques de l'apprentissage profond

Présenté par :

HAMEL Wassila

HALIMI Amel Ikram

Dirigé par :

Dr .LATRECHE Abdelkrim

Promotion 2023 - 2024

Remerciements

Nous remercions le bon Allah tout-puissant, qui nous a donné la force, la volonté et le courage pour terminer ce modeste travail. Nous adressons notre profond remerciement à ***Dr LATRECHE Abdelkrim*** *pour son encadrement, son écoute, ses élucidations, ses conseils, ses directives et encouragements qu'il nous a afflué. Nous remercions Mrs les jurés pour l'intérêt qu'ils ont porté à ce travail en acceptant d'être examinateurs. Ainsi, nous adressons nos remerciements les plus chaleureux à toutes les personnes qui ont aidé de près ou de loin par fruit de leur connaissance pendant toute la durée de notre parcours éducatif. A tous les enseignants dans le département d'informatique Tout simplement à tous ceux et celles qui méritent nos remerciements.*

Dédicace

Tout d'abord, je remercier mon Dieu pour tout

Je dédie ce modeste travail à :

Les parents les plus chers au monde, mon père ***HOUARI*** *et ma mère* ***NACERA,*** *aucune dédicace ne saurait mon respect, mon amour et ma considération pour les sacrifices que vous avez consenti pour mon instruction et mon bien je vous remercie pour tout le soutien et l'amour que vous me portez depuis mon enfance, Je leur souhaite longue vie et que Dieu les protège et les protège.*

A mes chères sœurs ***FOUZIA*** *et* ***FATIMA,*** *et ma chère nièce* ***AYAT ERAHMANE CHIRAZ*** *, qui ont été toujours à mes côtés et m'ont toujours soutenu tout au long deces longues années d'études.*

À mon binôme ***AMEL IKRAM*** *ou je ne connus que des moments du bonheur.*

Et Je tiens à exprimer mes vifs remerciements envers mon encadreur ***Dr LATRECHE Abdelkrim***

HAMEL WASSILA.

Dédicace

Tout d'abord, je remercier mon Dieu de m'avoir donné la force et le courage de mener à bien ce modeste travail. Je dédie ce mémoire:

*« à **mon très cher père**, Tu as toujours été pour moi un exemple du père respectueux, honnête, de la personne méticuleuse, je tiens à honorer l'homme que tu es. Grâce à toi **ABI** j'ai appris le sens du travail et de la responsabilité. Je voudrais te remercier pour ton amour, ta générosité, ta compréhension... Ton soutien fut une lumière dans tout mon parcours. Aucune dédicace ne saurait exprimer l'amour, l'estime et le respect que j'ai toujours eu pour toi. Ce modeste travail est le fruit de tous les sacrifices que tu as déployés pour mon éducation et ma formation. Je t'aime papa et j'implore le tout-puissant pour qu'il t'accorde une bonne santé et une vie longue et heureuse. »*

*« À **ma chère mère**, qui a toujours cru en moi et m'a encouragé à poursuivre mes rêves, même lorsque les obstacles semblaient insurmontables. Sans toi, je n'aurais pas acquis la force et la résilience nécessaires pour aboutir à ce stade. Ta lumière et ton amour continueront d'éclairer mon chemin. »*

*« A l'ame bienveillante de mon **grand père**, qui reste mon héros et mon amour éternel. Chacune de mes réussites lui est dédiée. »*

*À mon cher frère « **Mohamed** » et à ma chère sœur*

« Aya Hiba »

Je vous souhaite une vie pleine de bonheur et de réussite, et nous demandons à Dieu Tout-Puissant de vous accorder le succès Il vous protège et vous garde.

« À ceux qui tendent la main dans les moments de faiblesse mon cousin et mon grand frère

dr.HALIMI Mustapha»

« À la compagne du premier et du dernier pas, ma compagne, à celle qui m'a tendu la main et m'a rejoint dans ce travail et sa gestion WASSILA. »

AMEL IKRAM

Résumé

Dans le contexte actuel de surcharge causée par l'important volume de données numériques accessibles, les systèmes de recommandation permettent de guider l'utilisateur dans ses activités d'apprentissages, d'achats, de loisir, de regarder des films, de lectures..., en lui suggérant des items personnalisés et de fournir à des utilisateurs des suggestions qui répondent à leurs exigences. Pour cela, ils prédisent ses préférences relativement aux items qu'il n'a pas encore évalués. Parmi les approches classiques de recommandation, le filtrage collaboratif (CF) est la méthode la plus importante et la plus utilisée qui reposent sur les données collectées par le biais de retours d'utilisateurs, généralement sous la forme d'une matrice de notes, et tentent d'y découvrir les informations pertinentes pour caractériser et prédire les goûts des utilisateurs.

Dans ce mémoire, notre travail s'inscrit dans le cadre de l'application des techniques de l'apprentissage profond (deeplearning) dans les systèmes de recommandations collaboratifs. Précisément, l'objectif de notre travail est d'implémenter, et de comparer divers méthodes basées sur l'apprentissage profond, tel que : Co-occurrence CNN pour recommandation (CoCNN), CNN pour recommandation (CNN), le filtrage collaboratif neuronal (NCF) et les méthodes basées sur le machine learning, tel que : k plus proches voisins(KNN).

Nous avons comparés ces modèles pour déterminer quelle méthode est la plus appropriée pour modéliser l'interaction complexe utilisateur-élément.

Abstract

In the current context of overload caused by the large volume of accessible digital data, recommendation systems make it possible to guide the user in their learning, shopping, leisure activities, watching films, reading, etc..., by suggesting personalized items and providing users with suggestions that meet their requirements. To do this, they predict his preferences relative to items that he has not yet evaluated. Among the classic recommendation approaches, collaborative filtering (CF) is the most important and widely used method which relies on data collected through user feedback, usually in the form of a rating matrix, and attempt to discover relevant information to characterize and predict user tastes.

In this dissertation, our work is part of the application of deep learning techniques in collaborative recommendation systems. Precisely, the objective of our work is to implement and compare various methods based on deep learning, such as: Co-occurrence CNN for recommendation (CoCNN), CNN for recommendation (CNN), neural collaborative filtering (NCF) and methods based on machine learning, such as: k nearest neighbors (KNN).

We compared these models to determine which method is most appropriate for modeling complex user-element interaction.

ملخص

في سياق الضغط الحالي الناجم عن الكم الهائل من البيانات الرقمية المتاحة، تساعد أنظمة التوصية في توجيه المستخدم في أنشطته التعليمية، والتسويقية، والترفيهية، ومشاهدة الأفلام، والقراءة...الخ، من خلال اقتراح عناصر ذات طابع خاص وتقديم اقتراحات للمستخدمين التي تلبي متطلباتهم. ولتحقيق ذلك، تتوقع هذه الأنظمة الأولويات للمستخدمين بالنسبة للعناصر التي لم يقيموها بعد. من بين أساليب التوصية الكلاسيكية، يُعد الترشيح التعاوني (CF) الطريقة الأكثر أهمية والأكثر استخدامًا والتي تعتمد على البيانات التي تم جمعها من خلال ملاحظات المستخدمين، عادة في شكل مصفوفة تقييمات، وتحاول اكتشاف المعلومات ذات الصلة لوصف وتوقع أذواق المستخدمين.

يتمثل عملنا البحتي في هذه المذكرة في تطبيق تقنيات التعلم العميق (deep learning) في أنظمة التوصية التعاونية. وعلى وجه التحديد، يهدف عملنا إلى تنفيذ ومقارنة طرق مختلفة تعتمد على التعلم العميق مثل شبكة الالتفاف المشتركة للتوصية(CoCNN) ، شبكة الالتفاف للتوصية (CNN)، الترشيح التعاوني العصبي(NCF) ، وأخرى تعتمد على تعلم الآلة مثل أقرب الجيران.(KNN)

و بالنتيجة قمنا بمقارنة هذه النماذج لتحديد الطريقة الأنسب لنمذجة التفاعل المعقد بين المستخدم والعنصر

Table de metiers

Chapitre II Les Systèmes de recommandation

Liste des tableaux

CHAPITR I

CHAPITRE II

CHAPITRE III

Table des figures

CHAPITRE I

CHAPITRE II

CHAPITRE III

Introduction générale

Les systèmes de recommandation sont devenus de plus en plus pertinents pour les entreprises offrant tout type de service aux utilisateurs. Des entreprises telles qu'Amazon (leader dans le monde du commerce électronique), sont capables de recommander un petit nombre d'articles que l'utilisateur pourrait apprécier en se basant sur le contexte actuel et les comportements passés à travers un catalogue de centaines de millions d'articles. De même, Spotify (le service de streaming musical le plus utilisé). Propose une playlist Découverte chaque semaine qui s'adapte aux goûts de l'utilisateur et lui permet de découvrir de la nouvelle musique en permanence. Netflix (leader mondial du streaming vidéo), voit sa base d'abonnés augmenter constamment et ses utilisateurs passent des heures à regarder des séries et des films.

La valeur générée pour ces entreprises par les systèmes de recommandation n'est pas négligeable. Selon une étude de McKinsey, jusqu'à 35 % du chiffre d'affaires d'Amazon.com provient de recommandations, ce qui a augmenté leurs ventes totales de 29 % à 136 milliards de dollars en 2016 et à 177,9 milliards de dollars en 2017. De même, 75 % du contenu visionné sur Netflix est issu de recommandations de produits basées sur des algorithmes et des modèles prédictifs qui analysent les données de transaction et les tendances des médias numériques. De plus, les dirigeants de Netflix, Carlos A. Gomez-Uribe et Neil Hunt, affirment qu'une combinaison de personnalisation et de recommandations de leurs propres systèmes de recommandation leur permet d'économiser plus de 1 milliard de dollars chaque année, ce chiffre étant appelé à augmenter depuis la publication de l'article en 2015. Un facteur commun à toutes les implémentations de systèmes de recommandation ci-dessus est que les préférences des utilisateurs sont souvent difficiles à deviner, surtout lorsqu'il s'agit de grandes bases de clients et de catalogues encore plus importants de produits à recommander. La difficulté de cette tâche est telle qu'une entreprise a soit besoin d'un groupe de scientifiques et d'ingénieurs en données coûteux pour développer et maintenir de tels systèmes, soit doit externaliser ses services de recommandation, comme le fait la BBC. Avec la diversité croissante et le volume de données disponibles, les approches standard des systèmes de recommandation ne suffisent plus, car elles manquent de capacité à évoluer au même rythme que la base utilisateur/catalogue, ainsi que de flexibilité pour apprendre des tendances changeantes des données sans reconcevoir toute la solution à partir de zéro.

Les systèmes de recommandation utilisent principalement deux méthodes de filtrage pour proposer des recommandations personnalisées aux utilisateurs, à savoir le filtrage collaboratif et le filtrage basé sur le contenu. Le premier regroupe des techniques qui visent à opérer une sélection sur les items à présenter aux utilisateurs (filtrage) en se basant sur le comportement et les goûts exprimés de très nombreux autres utilisateurs. Le deuxième consiste à déterminer quels items coïncident le mieux avec les préférences de l'utilisateur. Le filtrage collaboratif est considéré comme la méthode la plus populaire et la plus répandue dans les systèmes de recommandation qui reposent sur les données collectées par le biais de retours d'utilisateurs, généralement sous la forme d'une matrice de notes, et tentent d'y découvrir les informations pertinentes pour caractériser et prédire les goûts des utilisateurs. Les systèmes de recommandation basés sur le filtrage collaboratif recommandent à l'utilisateur actif les items que d'autres utilisateurs ayant des préférences similaires ont aimés par le passé. Cette méthode de recommandation repose sur l'idée que, si les utilisateurs sont d'accord sur la qualité de certains items, ils seront probablement d'accord sur d'autres.

L’application des techniques de l’IA dans les systèmes de recommandation est en plein essor, c'est pourquoi la plupart des systèmes de recommandation les plus récents et les plus efficaces sont basés sur des algorithmes d'apprentissage automatique. Récemment, il a radicalement changé l'architecture des recommandations et a apporté plus d'opportunités pour améliorer les performances des systèmes de recommandation. Les progrès récents dans ces systèmes basés sur l'apprentissage profond ont retenu l'attention en surmontant les obstacles des modèles conventionnels et en obtenant une qualité de recommandation élevée.

L’apprentissage profond (en anglais deep learning) est un type d'intelligence artificielle dérivé de l'apprentissage automatique, il peut généralement être considéré comme un sous-domaine de ce dernier. C’est un ensemble de méthodes d'apprentissage automatique qui a permis des avancées importantes en intelligence artificielle dans les dernières années, tentant de modéliser avec un haut niveau d’abstraction des données grâce à des architectures articulées de différentes transformations non linéaires. L’apprentissage profond s'appuie sur un réseau de neurones artificiels s'inspirant du cerveau humain. Le système apprendra par exemple à reconnaître les lettres avant de s'attaquer aux mots dans un texte, ou détermine s'il y a un visage sur une photo avant de découvrir de quelle personne il s'agit.

Récemment, des réseaux de neurones profonds (DNN), qui ont obtenu des succès remarquables en matière de vision par ordinateur et de traitement du langage naturel, ont été proposés pour combler certaines des lacunes évoquées afin d'améliorer le filtrage collaboratif. Perceptrons multicouches (MLP) et factorisation matricielle dans un cadre de filtrage collaboratif (NCF) basé sur un réseau neuronal pour modéliser des facteurs latents d'utilisateur et d'élément qui pourraient apprendre une fonction arbitraire à partir des données pour surmonter l'interaction complexe de l'utilisateur et de l'élément ; le cadre proposé a montré des améliorations significatives par rapport aux autres méthodes publiées sur deux ensembles de données du monde réel, MovieLens et Pinterest. Cependant, chaque neurone d'une couche est connecté à chaque neurone de la couche suivante dans MLP, cette connexion dense rend le modèle rigide en matière d'apprentissage de fonctionnalités, ce qui lui donne des difficultés pour apprendre de nouvelles fonctionnalités.

Le réseau de neurones convolutifs (CNN) est un réseau de neurones à réaction développé à l'origine pour la vision par ordinateur. Bien que cela ne soit pas couramment appliqué au domaine des systèmes de recommandation, des études ont été menées sur l'utilisation de CNN pour extraire des informations auxiliaires afin de créer des recommandations. CNN a été utilisé dans la recommandation musicale en analysant l'acoustique des chansons et en faisant des prédictions basées sur le modèle latent d'élément approché, et des recommandations vidéo en modélisant les facteurs latents d'élément sur la base des critiques de vidéos et en les intégrant avec Probabilistic Factorisation matricielle. Autoencoder, une architecture de réseau neuronal profond qui vise à apprendre la représentation de dimension inférieure des données brutes et à reconstruire les données à partir de cette représentation, est également appliquée avec succès dans les systèmes de recommandation. Ces modèles d'apprentissage profond améliorent considérablement le développement de systèmes de recommandation.

Dans ce mémoire, notre travail s'inscrit dans le cadre de l’application des techniques de l’apprentissage profond (deep learning) dans les systèmes de recommandations collaboratifs.

L'objectif de ce travail est d'étudier comment les modèles d'apprentissage profond peuvent être utilisés pour résoudre efficacement le problème de recommandation. Précisément, l’objectif de notre travail est d’implémenter, et de comparer divers méthodes basées sur l’apprentissage profond : Co-occurrence CNN pour recommandation (CoCNN), CNN pour recommandation (CNN) qui est une version simplifiée de CoCNN sans motifs de

cooccurrence, le filtrage collaboratif neuronal (NCF) et la méthode basée sur le machine learning : k voisins les plus proches (KNN).

Pour atteindre cet objectif, nous avons structuré notre mémoire comme suit :

Une introduction générale

Dans le premier chapitre nous allons faire un état de l'art qui donne une vision générale sur l'apprentissage profond (deep learning).

Le deuxième chapitre aborde les systèmes de recommandation en détaillant les différentes classifications, leurs objectifs, et les divers types de recommandations. Il comprend également une revue des travaux de pointe dans ce domaine, en mettant l'accent sur les approches basées sur le machine learning et le deep learning.

Dans le troisième chapitre Nous exposons les résultats de nos expérimentations, en présentant les différents outils utilisés pour la réalisation.

Enfin, une conclusion générale qui résume notre travail.

Chapitre 3

Apprentissage profond (Deep learning)

I.1 Introduction

Les termes intelligence artificielle (IA), apprentissage automatique (ML) et apprentissage profond (DL) sont largement utilisés de manière synonyme, mais ils n'ont pas les mêmes significations. Cela est dû au fait que beaucoup d'entre nous ne peuvent pas discerner l'un de l'autre.

Cette image tente d'illustrer comment ils diffèrent ou sont liés les uns aux autres :

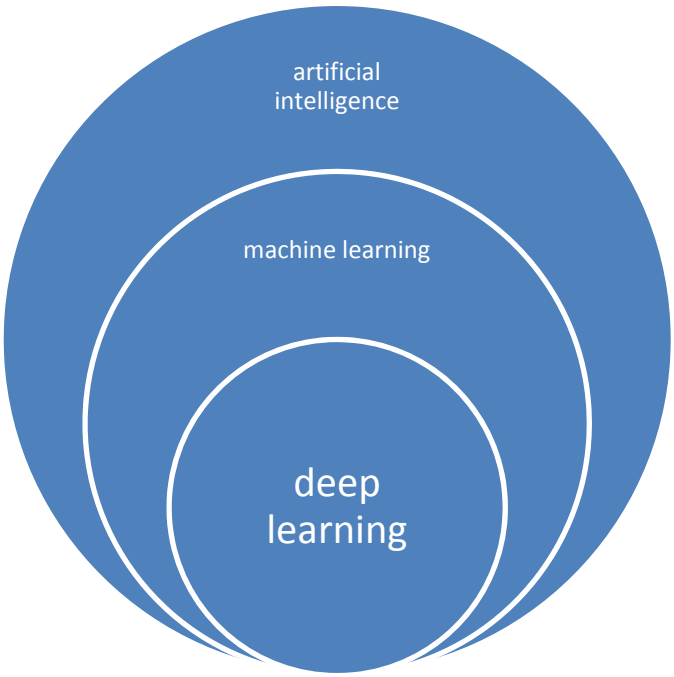


Figure I.1 : DL, ML et IA en relation les uns les autres

Les seuls problèmes que l'apprentissage automatique tente de résoudre sont tangibles. Nous avons également besoin de certains concepts liés à l'IA. Les réseaux neuronaux utilisés dans l'apprentissage automatique sont conçus pour ressembler aux processus de prise de décision humaine. L'apprentissage profond est l'objet exclusif de deux sous-ensembles limités d'outils et d'approches d'apprentissage automatique. Il doit être utilisé pour aborder tout problème qui nécessite de l'introspection, qu'il concerne les personnes ou les objets. Tout réseau neuronal profond aura trois types de couches différents :

- la couche cachée
- la couche d'entrée
- et la couche de sortie

Dans le domaine de l'apprentissage automatique, nous pouvons affirmer que l'apprentissage profond est la dernière expression. C'est une méthode de mise en pratique de l'apprentissage automatique. Pour analyser les données, en apprendre, et prendre des décisions défendables basées sur les informations acquises, nous employons un algorithme

machine. En essence, les couches de l'apprentissage profond sont utilisées pour construire un "réseau de neurones artificiels" autonome capable d'apprendre et de prendre des jugements éclairés. Il est possible de se référer à l'apprentissage profond comme un sous-domaine de l'apprentissage automatique.

Dans ce chapitre, tout d'abord, nous définissons l'intelligence artificielle, le Machine Learning et nous parlons du deep learning tel que sa définition, son histoire et ses différentes applications. Ensuite, nous présentons quelques notions sur les réseaux de neurones et ses types. Enfin, nous nous intéressons dans ce chapitre uniquement aux CNNs.

I.2 Intelligence Artificielle

I.2.1 Définition

L'intelligence artificielle est un ensemble de théories et de techniques qui développent des programmes informatiques complexes, ayant la capacité de mimiquer l'intelligence humaine en tant que capacité de raisonnement et d'apprentissage de manière machine (Fethi & Litim, 2021).

I.3 Apprentissage Automatique

I.3.1 Définition

L'apprentissage automatique implique de laisser l'ordinateur apprendre quel calcul effectuer, plutôt que de lui donner ce calcul (c'est-à-dire de le programmer explicitement). C'est du moins la définition de l'apprentissage automatique selon son inventeur Arthur Samuel, un mathématicien américain qui a développé un programme capable d'apprendre à jouer aux dames tout seul en 1959. Un autre Américain du nom de Tom Mitchell a donné une définition légèrement plus moderne de l'apprentissage automatique en 1998 en déclarant qu'une machine apprend lorsque sa performance dans l'exécution d'une certaine tâche s'améliore avec de nouvelles expériences. (Saint-Cirgue, 2019)

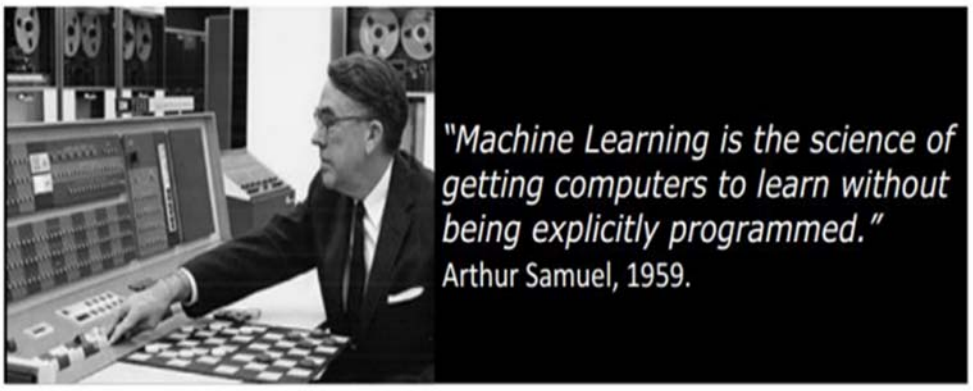


Figure I. 2 : Définition de l'apprentissage automatique*(Saint-Cirgue, 2019)*

I.3.2 Types d'Apprentissage

- **Apprentissage supervisé** avec un ensemble d'entraînement étiqueté.

Exemple de classification des e-mails avec des e-mails déjà étiquetés.

- **Apprentissage non supervisé** Découverte de motifs dans des données non étiquetées.

Exemple de regroupement de documents textuels similaires.

- **Apprentissage par renforcement** Apprendre à agir sur la base d'une récompense.

Exemple d'apprentissage pour jouer au Go, récompense gagner ou perdre.

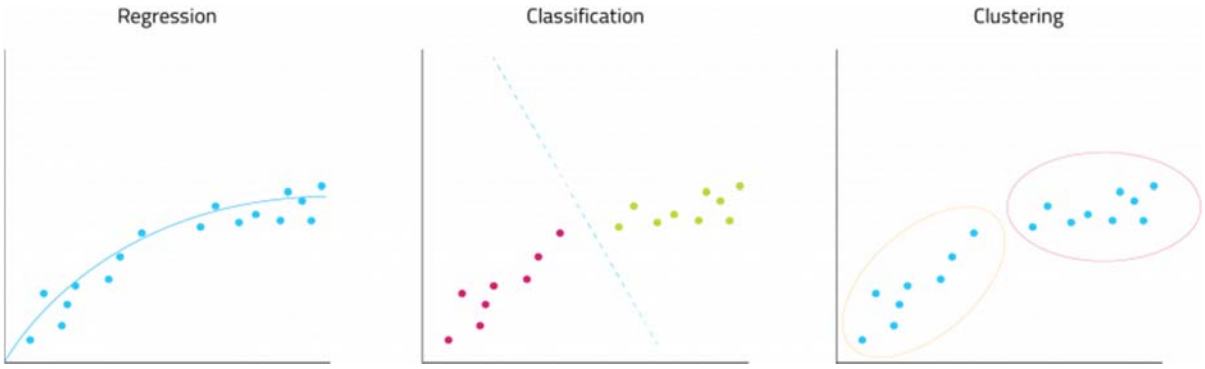


Figure I.3 ; exemple de régression, classification, clustering

I.4 Apprentissage profond

I.4.1 Définition

L'apprentissage profond est un type d'apprentissage automatique qui vise à entraîner et à enseigner à l'ordinateur à effectuer des fonctions humaines telles que la distinction des objets visuels et l'identification des sons et des images. Au lieu d'organiser les données, l'apprentissage profond définit ses propres paramètres de base, permettant à la machine d'apprendre par elle-même, et l'intérêt actuel pour l'apprentissage profond est en partie dû au bruit entourant l'intelligence artificielle. Les techniques d'apprentissage profond ont amélioré la capacité de classifier, de reconnaître, de détecter, etc.

L'apprentissage profond explore la structure complexe des grands ensembles de données en utilisant l'algorithme de rétroaction pour indiquer comment une machine doit modifier ses paramètres internes utilisés pour calculer la représentation de chaque couche à partir de la représentation de la couche précédente. Les réseaux convolutionnels profonds ont permis des avancées dans le traitement des images, de la vidéo, de la parole et de l'audio,

tandis que les réseaux récurrents ont été éclairés avec des données séquentielles telles que du texte et de la parole. (LECUN, Yoshua, & Geoffrey, 2015)

L'apprentissage profond est basé sur :

- Les réseaux neuronaux profonds.
- Un algorithme d'entraînement spécifique.
- L'apprentissage de représentations plus abstraites.
- Des ensembles de données plus grands.

I.4.2 Exemple sur l'apprentissage profond

Le réseau de neurones doit être entraîné pour cela. Pour ce faire, un ensemble d'images d'entraînement pour pratiquer l'apprentissage profond doit être compilé. Des milliers de photos de différents chats combinées avec des photographies d'autres objets seront regroupées dans cet ensemble. Ensuite, ces images sont transformées en données et envoyées sur le réseau. Ensuite, les neurones artificiels donnent du poids aux éléments. La couche finale de neurones rassemblera alors diverses informations pour déterminer s'il s'agit d'un chat ou non. Le réseau de neurones comparera ensuite cette réponse aux réponses humaines appropriées. Le réseau se souviendra de ce succès et l'utilisera plus tard pour reconnaître les chats si les réponses correspondent. D'autre part, le réseau note son erreur et ajuste le poids placé sur les différents neurones pour la corriger. Le processus est répété des milliers de fois jusqu'à ce que le réseau puisse reconnaître un chat sur une photo n'importe où. Cette méthode d'apprentissage est appelée apprentissage supervisé. L'apprentissage non supervisé est une méthode d'apprentissage supplémentaire. Cette méthode utilise des données non étiquetées. Pour savoir quels éléments dans une photo peuvent être pertinents, les réseaux de neurones doivent reconnaître les motifs au sein des ensembles de données . (lebigedata.fr)

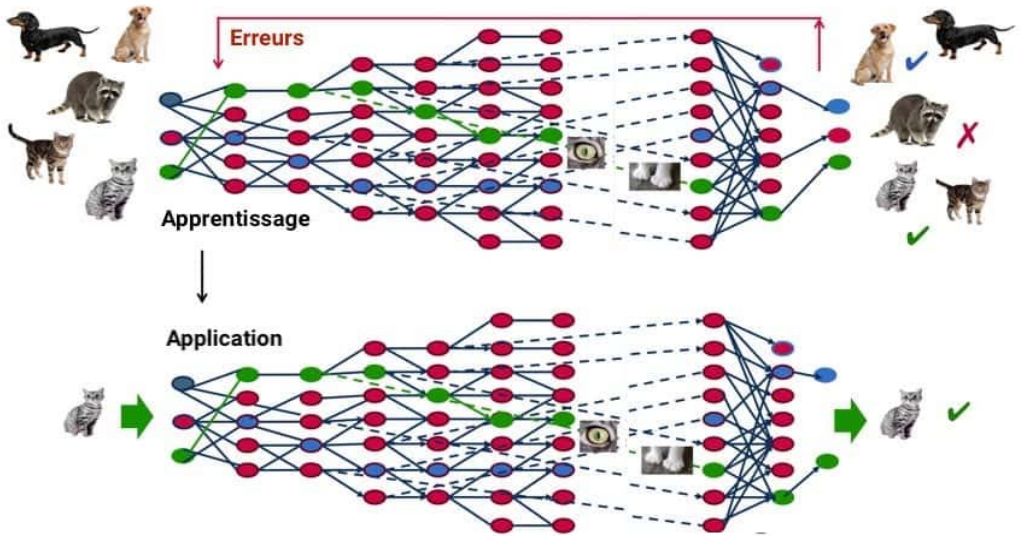


Figure I.4 : exemple d'apprentissage profond *(Futura)*

I.4.3 Historique du Deep learning

1943 - Le premier modèle mathématique d'un réseau de neurones

Au début de 1943, Walter Pitts et Warren McCullochont ont créé un modèle informatique basé sur les réseaux de neurones du cerveau humain, en utilisant un ensemble d'algorithmes mathématiques pour imiter la pensée humaine. Ce modèle a été appelé "logique de seuil". Le travail sur l'"apprentissage profond" a continué en lien avec une phase fluctuante de l'intelligence artificielle.

1950 - La prédiction de l'apprentissage machine

Turing, un mathématicien qui a déchiffré des codes pendant la Seconde Guerre mondiale, a prédit le développement de l'apprentissage machine en 1947. En 1950, il a inventé une machine similaire et a toujours parlé de "machines informatiques et d'intelligence", et a développé un "test de Turing" pour déterminer si l'ordinateur pouvait "penser".

1952 - Premiers programmes d'apprentissage machine

Arthur Samuel a continué à créer les premiers programmes d'apprentissage informatique. Et a commencé à montrer des caractéristiques d'apprentissage à travers le jeu de "dames", ce qui a permis à l'ordinateur de corriger ses erreurs et de trouver de meilleures façons de gagner. C'était l'un des premiers exemples d'apprentissage machine.

1957 - Pose des fondations des réseaux neuronaux profonds

En 1957, au laboratoire d'aviation de Cornell, Rosenblatt a écrit un article intitulé "La perspective de la perception et la reconnaissance automatique" dans lequel il déclarait qu'il créerait un système capable d'apprendre à reconnaître les similarités ou les identités entre les modèles d'information visuelle, électrique ou tonale de la même manière que les processus

cognitifs du cerveau. Cette idée a posé les bases de l'apprentissage ascendant et a été largement diffusée comme les bases des réseaux neuronaux profonds (DNN).

1959 - Découverte des cellules simples et complexes

En 1959, David Hubble et Torsten Wiesel ont découvert les cellules simples et complexes. Leurs activités biologiques ont été un facteur positif dans le domaine de l'apprentissage profond, de nombreux réseaux de neurones artificiels (RNA) étant inspirés par des observations biologiques.

1960 - Théorie du contrôle

En 1960, Henry J. Kelley a été reconnu pour avoir développé les fondements d'un modèle continu de rétro propagation. Deux ans plus tard, plus précisément en 1962, Stuart Dreyfus a développé une version simple basée uniquement sur la règle de la corde, mais le concept de rétro propagation, censé s'étendre au-delà des erreurs à des fins de formation, est resté inefficace jusqu'en 1985.

1965 - Les premiers réseaux d'apprentissage profond

Les efforts ont continué, y compris ceux d'Alexey GrigoryevichIvakhnenko, qui a travaillé sur les algorithmes d'apprentissage profond et développé une méthode de traitement des données. Valentin Grigor'evich Lapa a développé les techniques de cybernétique et de criminalistique, toutes datant de 1965. Ces modèles ont été utilisés avec des fonctions d'activation multidisciplinaires (équations complexes), puis analysés statistiquement.

Au cours des années 1970, un groupe de personnes a poursuivi ses recherches dans ce domaine, mais sans aucun financement.

1979-80 - Un ANN apprend à reconnaître les motifs visuels

Grâce à Fukushima, qui a créé un réseau de cellules neuronales Neocognitron, capable d'apprendre et d'identifier des motifs visuels et d'être utilisé dans la reconnaissance de caractères manuscrits et d'autres modèles, voire dans le traitement des langues naturelles. Son travail l'a amené à développer les premiers réseaux de neurones transformationnels, basés sur la régulation du cortex visuel chez les animaux.

1982 - Création des réseaux de Hopfield

Les réseaux de Hopfield sont un réseau de neurones fréquents qui fonctionne comme une mémoire rappelable qui reste un outil d'implémentation commun pour l'apprentissage profond en ce siècle.

1985 - Un programme apprend à prononcer des mots anglais

En 1985, Terry Segnovsky a créé le programme "NETtalk" qui permet la prononciation des mots anglais et parvient à s'améliorer avec le temps.

1986 - Amélioration de la reconnaissance de forme et de la prédiction de mots

En 1986, Romlhart, Hinton et Williams ont publié un article intitulé "Représentations de "l'apprentissage d'erreur rétrograde", qui a montré comment les réseaux de neurones existants pouvaient être considérablement améliorés pour de nombreuses tâches telles que la reconnaissance de forme, la prédiction de mots, etc. Il était considéré comme le père spirituel de l'apprentissage profond.

1989 - Q-Learning et machines lisant les numéros manuscrits

En 1970. C'est à cette époque que SeppoLinnainmaa a écrit une thèse de maîtrise comprenant le code FORTRAN pour la propagation ultérieure. Ce concept n'a été implémenté qu'en 1985. Rumelhart, Williams et Hinton ont démontré que le rejet du réseau neuronal pourrait fournir des représentations "intéressantes" de la distribution, ce qui a soulevé la question de savoir si la compréhension humaine était basée sur la logique symbolique ou la représentation distribuée.

En 1989, Yann LeCun a identifié la première présentation pratique de la dépression inverse, combinant les neurotransmetteurs et la propagation de chiffres "manuscrits". Ce système a permis de lire de nombreux chèques manuscrits.

1993 - Une tâche d'apprentissage très profond est résolue

En 1993, le informaticien allemand Schmidopfer a résolu la tâche de "l'apprentissage profond" qui nécessitait plus de 1 000 couches dans le réseau de neurones récurrent.

1995 - Machines à vecteurs de support

SVM est essentiellement un système d'identification et de planification de données similaires. Il peut être utilisé pour trier des textes, reconnaître des caractères manuscrits et trier des images pour l'apprentissage profond.

1997 - Proposition d'une mémoire à long terme.

1998 - Apprentissage du gradient.

2009 - Lancement d'ImageNet.

ImageNet a été lancé en 2009 par Fei Fei Lee, professeur et directeur du Laboratoire d'intelligence artificielle de l'Université Stanford.

2011 - Création d'AlexNet

Son succès a commencé à raviver le réseau neuronal dans la communauté de l'apprentissage profond.

2012 - L'expérience du chat

L'expérience du 'chat' a été une grande avancée. Un réseau de neurones réparti sur des milliers d'ordinateurs a été utilisé, et 1000000 d'images inattendues - capturées de

manière aléatoire sur YouTube - ont été détectées et analysées. Ils ont constaté que le programme avait appris à reconnaître et à identifier les chats et que la proportion de tentatives d'apprentissage avait considérablement augmenté.

2014 – DeepFace

Il a été développé en 2014 et le système de médias sociaux profond - DeepFace - utilise des réseaux neuronaux pour identifier les visages avec une précision de 97,35%.

2014 - Réseaux générateurs adverses (GAN)

Il a été introduit par une équipe de chercheurs dirigée par Ian Goodfellow en 2014. Le réseau GAN utilise deux réseaux concurrents : le premier prend les données et tente de créer des échantillons non caractéristiques, le second reçoit les données et les échantillons générés et doit déterminer si chaque point de données est correct ou créé.

2016 - Produits puissants d'apprentissage machine

Certaines entreprises, comme CrayInc, ont été en mesure de fournir des produits et des solutions puissants en matière de machines et d'apprentissage profond. Un exemple est l'utilisation du logiciel Microsoft Neural Networking sur des superordinateurs équipés de processeurs graphiques pour effectuer des tâches d'apprentissage de données en profondeur en très peu de temps.

En 2016, le programme AlphaGo, développé par Google DeepMind, a été sensationnel en battant dans le jeu de go, par le score de 4 à 1, le Sud-Coréen Lee Sedol, considéré comme le meilleur joueur du monde.

I.4.4 Application de l'apprentissage profond

Systèmes de recommandation

Les systèmes de recommandation utilisent l'apprentissage profond pour extraire des caractéristiques significatives pour un modèle à facteurs latents pour des recommandations de musique et de revues basées sur le contenu. L'apprentissage profond à plusieurs vues a été appliqué pour apprendre les préférences des utilisateurs dans plusieurs domaines. Le modèle utilise une approche collaborative hybride et est basé sur le contenu, améliorant ainsi les recommandations dans plusieurs domaines.

Reconnaissance faciale

La reconnaissance faciale est l'identification et la vérification des personnes sur une photographie par leur visage. C'est une tâche qui est effectuée de manière triviale par les humains. Les méthodes d'apprentissage profond sont capables d'apprendre des représentations riches et compactes des visages, permettant aux modèles de reconnaître les visages humains.

Traitement automatique du langage

Le traitement automatique du langage naturel (TALN) est la relation entre les ordinateurs et le langage humain. Plus précisément, le traitement automatique du langage naturel consiste à comprendre, analyser, manipuler et/ou générer un langage naturel par l'ordinateur. Les systèmes de TALN captent le sens d'une entrée de mot (phrases, paragraphes, pages, etc.) sous forme de sortie structurée (qui varie énormément en fonction de l'application). Le traitement automatique du langage naturel est un élément fondamental de l'intelligence artificielle (brownlee, 2019)

Traduction automatique

Grâce à l'apprentissage profond, nous avons accès à divers services de traduction. L'un des plus populaires, Google Translate, permet à son utilisateur de traduire facilement une langue. L'apprentissage profond a permis une amélioration considérable dans cette application (brownlee, 2019)

Voiture autonome

Des millions de données sont introduites dans le système pour construire un modèle, entraîner la machine à apprendre, puis tester le résultat dans un environnement réel. Les données proviennent de caméras, de capteurs et de la cartographie géographique, créant des modèles sophistiqués capables de naviguer dans la circulation et d'identifier les trajets et les panneaux

Marketing

L'application de l'apprentissage profond dans le domaine du marketing numérique aide les professionnels du marketing à évaluer l'efficacité de leurs campagnes. Les prédictions précises des algorithmes d'apprentissage profond aident à prédire la demande et la satisfaction des clients et les aident à créer un ciblage basé sur leur marque. Cela devient vraiment un atout précieux pour le professionnel du marketing moderne et lui permet de maintenir la compétitivité de ses services.

Santé

L'apprentissage profond joue un rôle important dans le diagnostic médical et la recherche. Il aide à diagnostiquer les maladies mortelles, les résultats pathologiques et le traitement conduisant à la standardisation et à la compréhension de la génétique pour prédire les risques de maladies futures.

I.4.5 Grille de neurones

I.4.5.1 Neurone biologique

Le système nerveux est un réseau de neurones biologiques composé de milliards de cellules. Les neurones sont en fait reliés les uns aux autres pour construire des réseaux de plus en plus complexes, ils ne sont pas séparés les uns des autres.

Il existe trois composants principaux d'un neurone biologique :

Le centre de contrôle, qui constitue le corps cellulaire, traite les données que les dendrites reçoivent. Les principaux conduits par lesquels les informations du monde extérieur circulent sont les dendrites. Le fil conducteur qui envoie le signal de sortie du corps cellulaire aux neurones voisins est appelé axone. (datascientest)

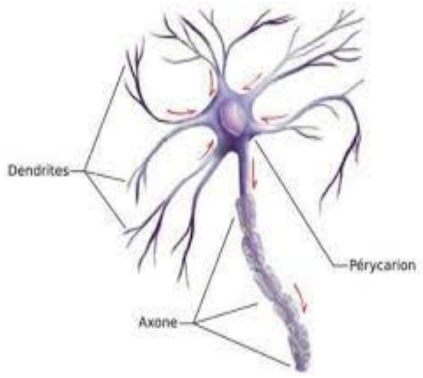


Figure I. 5 : neurone biologique *(ResearchGate)*

I.4.5.2 Du neurone biologique au neurone artificiel

Même avec tous les efforts des algorithmes et de l'intelligence artificielle à l'esprit, les applications sont actuellement à la limite de leur potentiel actuel. Les réseaux de neurones artificiels ont été développés sur la base de l'hypothèse que le comportement intelligent humain est le résultat de la structure et des éléments de base du système nerveux central (neurones).

Cette inspiration d'un modèle biologique vient du fait que le cerveau humain est un système d'apprentissage basé sur la structure contenant environ 100 milliards de neurones interconnectés par 10 000 contacts synaptiques. Tout comme le neurone biologique, le neurone artificiel est une unité de calcul simple. Ce "mini-processeur" calcule la somme pondérée des signaux entrants et renvoie une fonction de cette somme en sortie. Cette fonction f est définie lors de la création du neurone, elle dépendra de l'objectif souhaité concernant le futur réseau de neurones artificiels (la fonction la plus simple est le seuil : si $\sum x > 0, f = 1, f = 0$ alors). Les informations 'stockées' à l'intérieur d'un neurone ne sont finalement rien d'autre que ses poids.

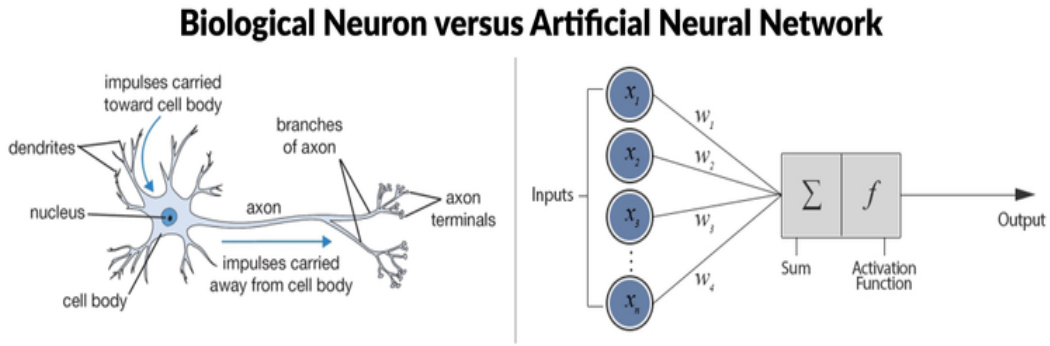


Figure I. 6 : Du neurone biologique au neurone artificiel *(datascientest)*

I.4.6 neurone artificiel

Les réseaux neuronaux artificiels (RNA) cherchent à simuler ces réseaux et à faire agir les ordinateurs comme des cellules cérébrales interconnectées. Différentes parties du cerveau humain sont responsables du traitement de différentes informations, et ces zones du cerveau sont disposées de manière hiérarchique, ou en couches. Ainsi, lorsque les informations entrent dans le cerveau, chaque niveau de neurones traite les informations, fournit des informations et les transmet à la couche supérieure suivante (Agatonovic & Beresford, 2000). C'est cette approche en couches du traitement de l'information et de la prise de décision que les RNA tentent de simuler. Dans sa forme la plus simple, un RNA ne peut avoir que trois couches de neurones :

- la couche d'entrée : où les données entrent dans le système.
- la couche cachée : où l'information est traitée.
- la couche de sortie : où le système décide quoi faire en fonction des données.

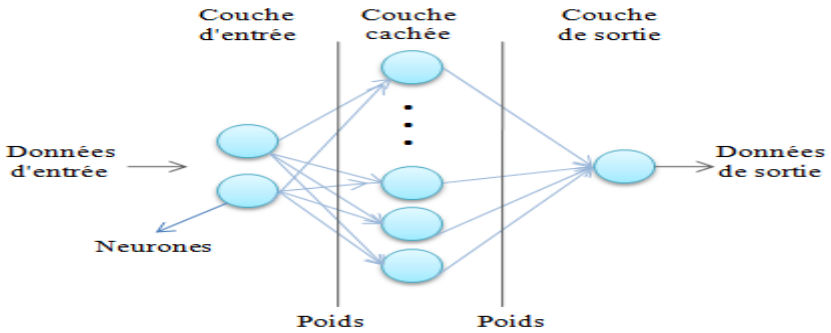


Figure I.7 Architecture de neurone artificiel

I.4.7 Réseaux neuronaux profonds

Un réseau neuronal profond (DNN) est un type de réseau neuronal artificiel (RNA) qui possède plusieurs couches entre les couches d'entrée et de sortie. Mathématiquement, le

DNN effectue des transformations sur les entrées, qu'elles soient linéaires ou non linéaires, pour produire une sortie. Le réseau se déplace à travers ces couches en calculant les probabilités pour chaque sortie (Rawal, et al., 2019).Les utilisateurs peuvent passer en revue les résultats et sélectionner les probabilités que le réseau doit afficher (par exemple, celles supérieures à un certain seuil), puis renvoyer l'étiquette proposée. Chaque manipulation mathématique est considérée comme une couche, et les DNN complexes peuvent avoir de nombreuses couches, d'où le nom de "réseaux profonds".

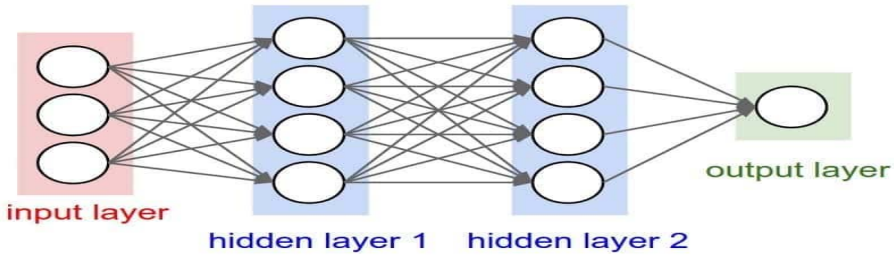


Figure I.8 : Architecture des réseaux neuronaux profonds *(lebigedata.fr)*

I.4.8 Types de réseaux neuronaux

I.4.8.1 Perceptron

Le Perceptron a été inventé en 1958 au Laboratoire d'aviation de Cornell par (Frank Rosenblat) financé par le Bureau de recherche navale des États-Unis. Le mot vient du verbe latin (Percipio) qui signifie en anglais understand, ce qui montre que le robot ou l'appareil pouvait apprendre et comprendre le monde extérieur. Un perceptron multicouche avec plusieurs couches cachées entre les couches d'entrée et de sortie est un réseau de neurones profonds (DNN) (Akram, 2020)

I.4.8.2 Réseau neuronal à propagation directe (FNN)

Le réseau de neurones à action directe est le premier type de réseau neuronal artificiel conçu. C'est aussi le plus simple. Dans ce réseau, l'information ne se déplace que dans une seule direction, vers l'avant, à partir des nœuds d'entrée, en passant par les couches cachées (le cas échéant) et vers les nœuds de sortie. Il n'y a pas de cycles ou de boucles dans le réseau.

I.4.8.3 Réseaux neuronaux convolutifs (CNN)

Un réseau neuronal convolutif est un type de réseau multicouche. Il se compose d'au moins cinq couches. Nous discuterons de cela en détail dans le titre suivant

I.4.8.4 Réseaux neuronaux récurrents (RNN)

Les réseaux neuronaux récurrents (RNN) sont une famille de réseaux neuronaux conçus pour le traitement de données séquentielles. Le chercheur allemand Jürgen Schmidhuberles compare à un grand réseau de rétroaction neuronale dans le cerveau humain, capable d'apprendre à traduire un flux d'entrées sensorielles en une séquence de sorties motrices utiles tout au long de la vie. Ce qui les distingue, c'est leur capacité à modéliser la dimension temporelle, ce qui leur permet de reconnaître les caractéristiques et

les motifs séquentiels dans les données, ce qui est crucial pour prédire le scénario le plus probable suivant. (Kabouche, 2022)

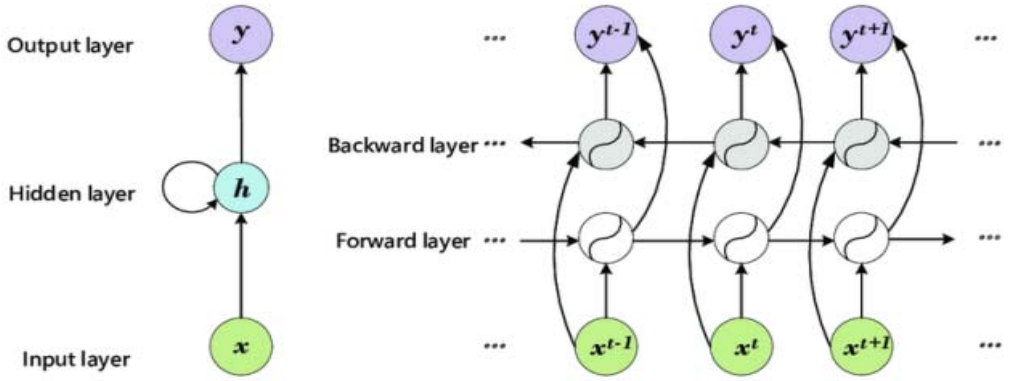


Figure I.9 : Réseaux neuronaux récurrents *(GEEKFLARE)*

I.4.8.5 Réseaux neuronaux résonants

L'appellation du réseau neuronal fait encore une fois référence à son fonctionnement. En effet, au sein des réseaux de neurones à résonance, l'activation de tous les neurones est renvoyée à tous les autres neurones au sein du système. Ce renvoi provoque des oscillations, d'où la raison du terme résonance.

Il va sans dire que ces réseaux de neurones peuvent prendre différentes formes avec des degrés de complexité plutôt élevés. Pour aller plus loin, je vous invite à vous intéresser à la Mémoire Associative Bidirectionnel qui permet d'associer deux informations de natures différentes ou encore le modèle ART (Adaptative Résonance Theory) qui fait interagir une information contextuelle avec la connaissance que l'on a déjà pour identifier ou reconnaître des objets (Belhaouci, 2019)

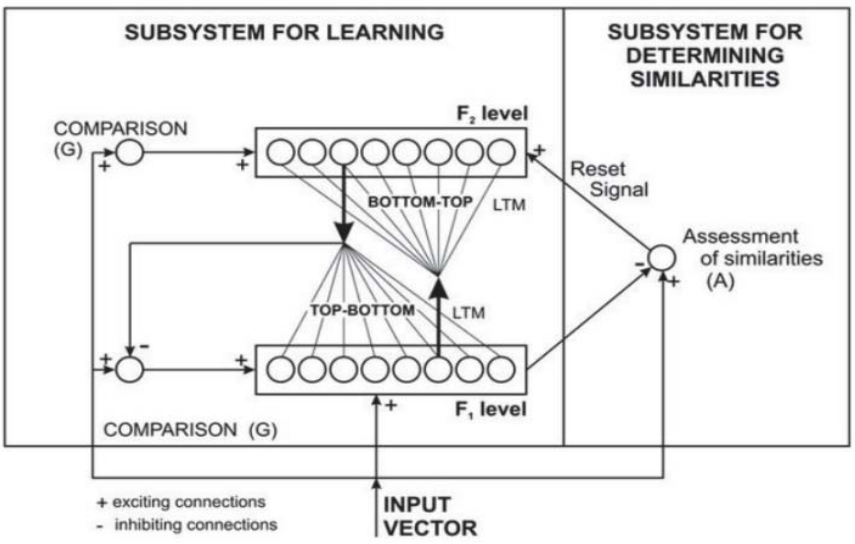


Figure I.10 : Réseaux neuronaux résonants *(ResearchGate)*

I.4.8.6 Réseaux neuronaux auto-organisés:

Les Réseaux de neurones auto-organisés sont surtout adaptés pour le traitement de d'informations spatiales. Par des méthodes d'apprentissage non-supervisé, les réseaux neuronaux auto-organisés sont capables d'étudier la répartition de données dans des grands espaces comme par exemple pour des problématiques de clusterisation ou de classifications.

Le modèle le plus connu de ce type de réseaux de neurones est sans doute la carte auto-organisatrice de Kohonen (Belhaouci, 2019):

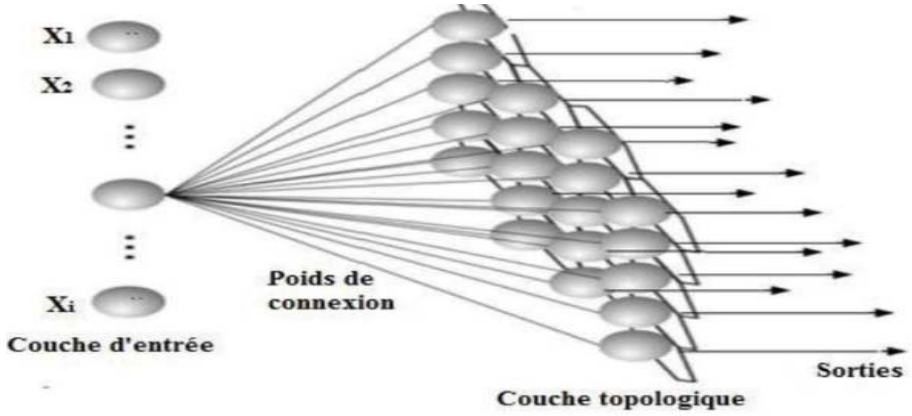


Figure I.11 : Réseaux de neurones auto-organisés(Carte auto organisatrice de Kohonen)
(ResearchGate)

I.5 Fonctionnement linéaire des neurones

I.5.1 Fonctions d'activation neuronale

Après avoir produit le produit entre ses entrées et ses poids, le neurone applique également une non-linéarité à ce résultat. La fonction d'activation est cette fonction non linéaire. La fonction d'activation est essentielle au réseau neuronal. Cette fonction détermine si le neurone est activé ou non. Elle ajoute le biais et calcule la somme pondérée des entrées. La transformation de la valeur d'entrée est non linéaire.

Cette sortie est envoyée à la couche suivante après la transformation. Dans les réseaux neuronaux, la non-linéarité est si importante qu'un réseau neuronal devient simplement un modèle linéaire sans la fonction d'activation. Il existe de nombreux types de ces fonctions, notamment:

I.5.1.1 Fonction sigmoïde

En mathématiques, la fonction sigmoïde (dite aussi *courbe en S*[1]) est définie par :

$$f(x) = \frac{1}{1 + e^{-x}} \qquad [I.1]$$

Pour tout réel x

Mais on la généralise à toute fonction dont l'expression est :

$$f_\lambda(x) = f(\lambda x) = \frac{1}{1 + e^{-\lambda x}} \qquad [I.2]$$

Elle représente la fonction de répartition de la loi logistique. La courbe sigmoïde génère par transformation affine une partie des courbes logistiques, ce qui en fait une représentante privilégiée.

La fonction sigmoïde est souvent utilisée dans les réseaux de neurones parce qu'elle est dérivable, ce qui est nécessaire pour l'algorithme de rétro propagation de Werbos, et parce que son codomaine est l'intervalle [0 ;1], ce qui permet d'obtenir des valeurs analogues à des probabilités. La dérivée de sa fonction inverse est extrêmement simple à calculer, ce qui permet d'améliorer les performances des algorithmes d'optimisation(WIKIPEDIA)

I.6. Fonctions de perte et de coût dans l'apprentissage profond

I.6.1 Fonction de perte

Chaque exemple d'apprentissage de l'ensemble d'apprentissage a sa propre fonction de perte. Une méthode d'évaluation de "comment votre algorithme modélise votre ensemble de données" est la fonction de perte. Votre fonction de perte produira un chiffre plus élevé si vos prédictions sont complètement fausses. Elle donnera un nombre plus bas si elles sont assez bonnes. Votre fonction de perte vous indique si vous vous améliorez ou non lorsque vous ajustez votre algorithme pour essayer d'améliorer votre modèle.

La "perte" nous aide à comprendre la différence entre la valeur réelle et la valeur prédite. Elle est divisée en trois catégories:

- Classification multi-classes
- Classification binaire
- Régression.

I.6.2 Fonction de coût

La fonction de coût, également appelée fonction objective, est une mesure globale de l'erreur du modèle sur tous les exemples d'entraînement. Elle est souvent notée J(θ), où θ représente les paramètres du modèle.

La fonction de coût est calculée en agrégeant les pertes sur tous les exemples d'entraînement. Pour un ensemble d'entraînement de taille m, la fonction de coût peut être définie comme la moyenne des fonctions de perte individuelles sur chaque exemple.

L'objectif de l'entraînement du modèle est de minimiser cette fonction de coût en ajustant les paramètres du modèle θ. Pour ce faire, des techniques d'optimisation telles que la descente de gradient sont souvent utilisées pour trouver les valeurs de θ qui minimisent la fonction de coût.

Exemple de fonction de coût utilisant la fonction de perte quadratique pour la régression:

$$J = \frac{1}{2m}\sum_{n=1}^{m}(\hat{y} - y)^2 \qquad [I.3]$$

I.7 Réseaux neuronaux convolutifs

I.7.1 Introduction

Actuellement, les réseaux CNN sont les modèles les plus efficaces pour classer et reconnaître les images. Ils utilisent une procédure mathématique linéaire unique connue sous le nom de convolution. Les deux composants principaux de ces réseaux sont la partie convolutive, qui collecte les caractéristiques de l'image, et la partie convolutive, qui crée de nouvelles images appelées cartes de caractéristiques. Certains filtres intermédiaires appliquent une opération locale maximale qui réduit la résolution de l'image. Pour catégoriser l'image, une deuxième section constituée de neurones entièrement connectés est couplée à la partie convolutive. La distribution de probabilité sur les classes est ensuite affichée en combinant la sortie de la section convolutive avec une deuxième partie, qui est une paire de neurones par neurone. Grâce à sa corrélation spatiale et à la création de liens spatialement corrélés entre les données par des couches convolutives, les CNN sont capables de traiter directement de gros volumes de données. Des couches de sous-échantillonnage (pooling) suivent généralement ces couches convolutives, réduisant ainsi considérablement la quantité de données.

I.7.2 Concept et fonctionnement de l'apprentissage profond

Votre façon de penser aux problèmes que vous résolvez avec l'analyse change avec l'apprentissage profond. Cela implique d'enseigner à l'ordinateur comment résoudre un problème, puis de le former à le résoudre lui-même.

Une méthode classique d'analyse consiste à utiliser les données disponibles pour créer de nouvelles variables, puis à choisir un modèle analytique et enfin à estimer les paramètres (ou inconnus) du modèle. Ces méthodes peuvent produire des systèmes prédictifs qui ne peuvent pas être généralisés car leur exhaustivité et leur précision dépendent de la qualité du modèle et de ses caractéristiques. Tout doit être redémarré si

davantage de données sont ajoutées. Remplacer la formulation et la spécification du modèle par des caractérisations hiérarchiques (ou des couches), qui apprennent à reconnaître les caractéristiques latentes des données à partir des régularités des couches, est une nouvelle approche basée sur l'apprentissage profond. Avec l'apprentissage profond, le changement de paradigme passe de l'ingénierie fonctionnelle à la représentation fonctionnelle.

Prenons un exemple simple pour mieux comprendre comment le faire, nous voulons identifier un poisson parmi un grand groupe de poissons capturés, quelle que soit leur méthode de capture.

Afin de former le réseau de neurones artificiels à la pratique de l'apprentissage profond, une grande collection d'images différentes de poissons et d'autres objets doit être assemblée. Après que les cellules nerveuses artificielles ont attribué un poids à différents éléments, ces images sont converties en données et transmises au réseau. La dernière couche de neurones collecte ensuite des informations pour déterminer s'il s'agit d'un poisson ou non.

Le réseau de neurones artificiels comparera ensuite ces résultats à la réponse donnée. S'il y a correspondance, le réseau enregistrera ce succès car il l'utilisera dans d'autres municipalités. Le réseau reconnaîtra le déséquilibre et ajustera le poids des différents neurones pour corriger son erreur si le résultat est négatif. Le processus est répété des milliers de fois jusqu'à ce que l'image soit identifiée. Ce type d'apprentissage est connu sous le nom d'apprentissage supervisé.

La deuxième méthode est l'apprentissage non supervisé, qui est basé sur des données non classifiées, c'est-à-dire que le réseau doit identifier des motifs dans des ensembles de données pour découvrir par lui-même que chaque élément de l'image peut être pertinent.

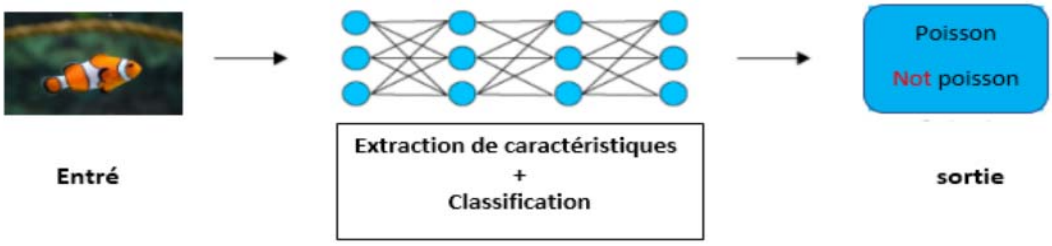


Figure I.12 : exemple de Concept et opération

I.7.3 Les différentes couches du réseau neuronal convolutionnel

I.7.3.1 Couche de convolution

Le cœur du réseau neuronal convolutionnel est la convolution. À l'origine, une convolution était un outil mathématique (également appelé produit de convolution) largement utilisé dans l'édition d'images car elle permet d'extraire les caractéristiques des images d'entrée, afin d'utiliser un bon filtre. En fait, une convolution prend simplement une image et un filtre (qui est une autre image), effectue un calcul et renvoie une nouvelle image (généralement plus petite) . (Azzoune & Khaldoun, 2019-2020)

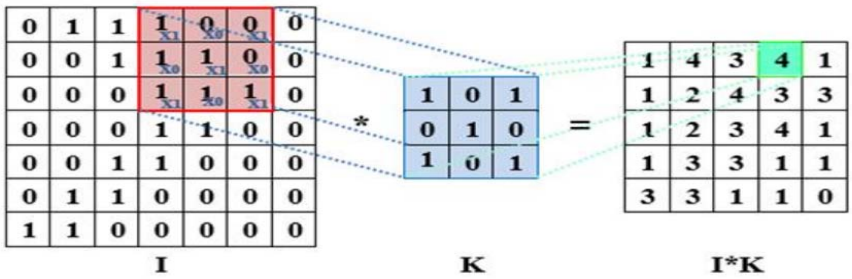


Figure I.13 : Illustration d'une convolution *(Kabouche, 2022)*

Le volume de la couche de convolution (également appelé volume de sortie) peut être mis à l'échelle en utilisant trois hyperparamètres : la profondeur, le pas et la marge. (Boughaba & Boukhris, 2016-2017)

- La profondeur de la couche : est mesurée par le nombre de noyaux de convolution, ou le nombre de neurones liés au même champ récepteur.
- Le pas : est utilisé pour gérer le chevauchement des champs récepteurs. Plus les champs récepteurs se chevauchent, plus le volume de sortie est grand.
- La marge (à 0) ou le rembourrage de zéro : il est parfois pratique de mettre des zéros à la limite profonde du volume d'entrée de l'apprentissage. Le troisième hyperparamètre est la taille de ce "rembourrage de zéro". Cette marge permet d'ajuster la dimension spatiale du volume de sortie. Parfois, il est préférable de garder la même surface que le volume d'entrée.

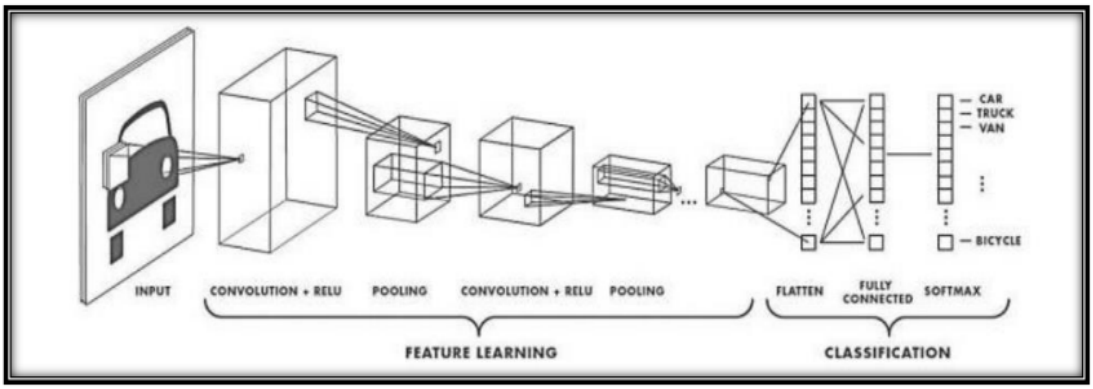


Figure I.14 : un exemple de convolution *(Kabouche, 2022)*

Chaque image utilisée pour l'apprentissage a des filtres appliqués à différentes résolutions, et la sortie de chaque image convoluée est utilisée comme entrée pour la couche suivante.

I.7.3.1.1 Les différentes convolutions

Il existe plusieurs types de convolutions, bien que nous utilisions généralement la convolution de base, il peut être utile de connaître les outils à notre disposition.

- La convolution classique : qui représente le décalage du noyau entre chaque calcul, et le padding qui est la manière dont on peut "dépasser" l'image pour appliquer la convolution.
- La convolution dilatée : identique à la convolution lorsque le noyau est brisé (nous prenons, par exemple, un pixel sur deux pour calculer la convolution). Il y a un paramètre supplémentaire : le taux de dilatation, qui est le nombre de pixels à ignorer.
- La convolution transposée : qui construit la sortie comme si vous inversiez une convolution sur l'image.
- La convolution séparable : qui est une convolution qui peut être décomposée en convolutions plus simples. (DAHAH, 2020)

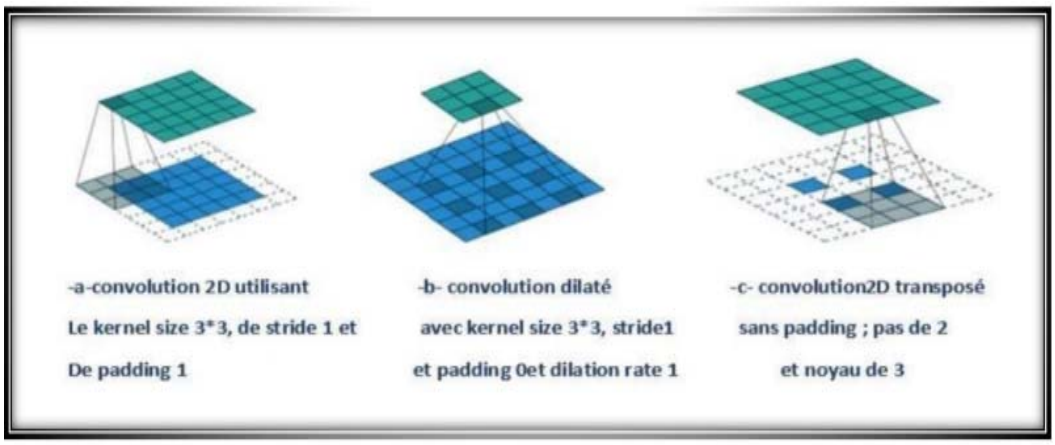


Figure I.15 : Les différentes convolutions *(Kabouche, 2022)*

I.7.3.2 Couche ReLU

C'est une couche qui s'intercale entre les couches de traitement pour améliorer l'efficacité du traitement et qui effectue une fonction mathématique (fonction d'activation) sur les signaux de sortie. Les neurones sont incités à retourner des valeurs positives grâce à la fonction ReLUF(x) = max (0, x) (DAHAH, 2020). Il arrive souvent que la fonction d'activation soit une fonction non linéaire. Leur objectif est de permettre aux réseaux de neurones d'acquérir des compétences plus complexes que la simple régression linéaire, car multiplier les poids d'une couche cachée est une transformation linéaire simple. (NACER, 2019)

Une unité linéaire redressée (ReLU) : Après chaque opération de convolution, ou lorsque toutes les valeurs de pixels négatives sont mises à zéro, elle est utilisée. Comme la majorité des données provient du monde réel et que la plupart des caractéristiques appliquées à l'une des cartes d'entrée produisent une carte de caractères redressés appelée carte de caractères redressés, le but de ReLU est d'introduire de la non-linéarité dans notre CNN. (Azzoune & Khaldoun, 2019-2020)

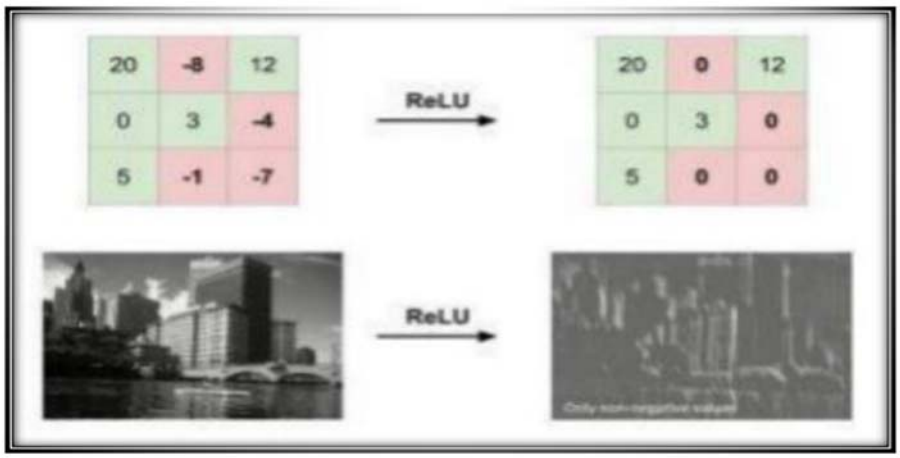


Figure I.16: Application of RELU from a matrix and an image *(Kabouche, 2022)*

I.7.3.3 Couche de Pooling

Ce type de couche est souvent placé entre deux couches de convolution : il reçoit plusieurs cartes de caractéristiques en entrée et applique l'opération de regroupement à chacune d'elles. L'opération d'assemblage, également appelée sous-échantillonnage, consiste à réduire la taille des images tout en préservant leurs caractéristiques essentielles. Pour cela, l'image est divisée en cellules régulières et la valeur maximale est conservée dans chaque cellule. Afin de ne pas perdre trop d'informations, de petites cellules carrées sont souvent utilisées en pratique. Les cellules adjacentes de 2 x 2 pixels qui ne se chevauchent pas ou des cellules de 3 x 3 pixels qui se chevauchent avec un pas de 2 pixels sont les options les plus courantes. Il y a le même nombre de cartes de caractéristiques en sortie qu'en entrée, mais elles sont considérablement plus petites. La couche de regroupement réduit le nombre de paramètres et de calculs du réseau. Par conséquent, l'efficacité du réseau est améliorée et le sur-apprentissage est évité. Il existe différents types de regroupement : Max, Moyenne, Somme, etc. (Azzoune & Khaldoun, 2019-2020)

- **Max pooling** : cela signifie prendre la valeur maximale de la sélection. C'est le type le plus couramment utilisé car il est rapide à calculer (immédiatement) et permet de simplifier efficacement l'image.

- **Averagepooling** : c'est le calcul de la moyenne des pixels de la sélection en divisant la somme de toutes les valeurs par le nombre de valeurs. Ainsi, une valeur intermédiaire est obtenue pour représenter cette quantité de pixels.

- **The sumpooling**: c'est la moyenne sans avoir divisé le nombre de valeurs (on ne calcule que la somme de leurs valeurs).

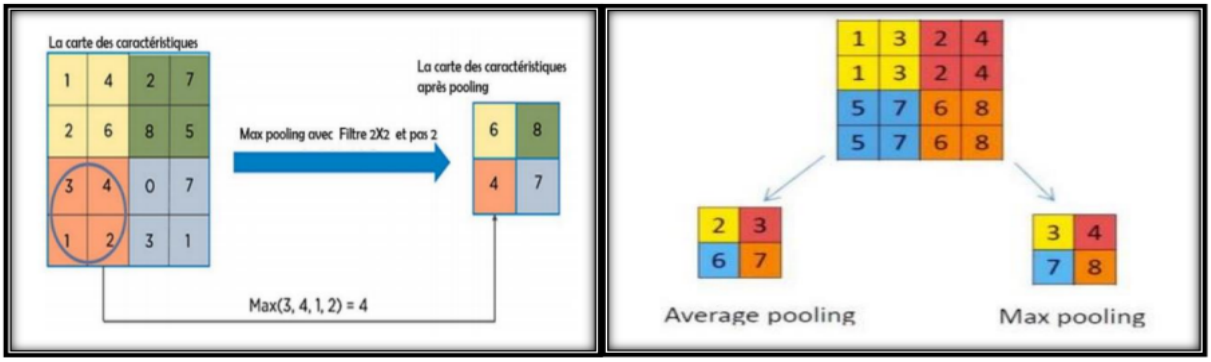


Figure I. 17 : Pooling avec un filtre 2*2 et un pas de 2 *(Kabouche, 2022)*

I.7.3.4Flatten

Le flattening est la dernière étape de la section "extraction d'informations", qui consiste simplement à combiner toutes les images (matrices) que nous avons pour en faire un (long) vecteur. Les pixels sont récupérés ligne par ligne et ajoutés au vecteur final, car ce ne sont plus des images ou des pixels, mais des matrices de nombres. (DAHAH, 2020)

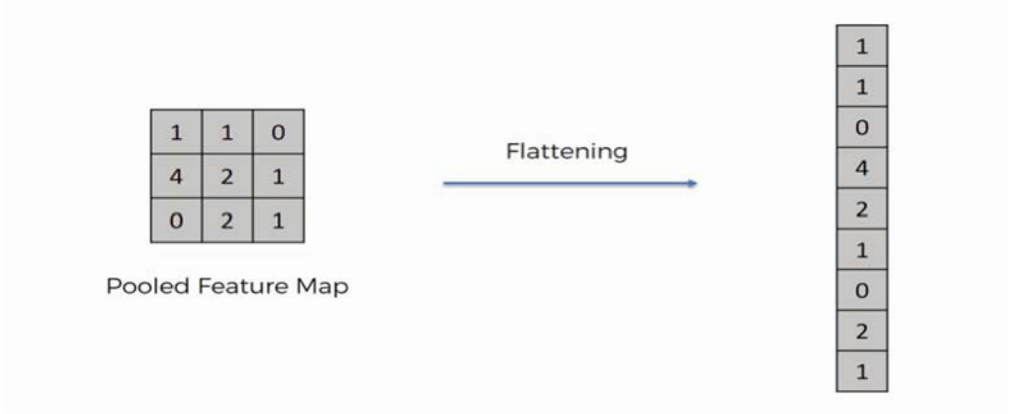


Figure I.18 : Exemple de Flatten*(SuperDataScience)*

I.7.3.5 Couche entièrement connectée

Le raisonnement de haut niveau dans un réseau neuronal se fait à travers les couches entièrement connectées après les couches de convolution et de regroupement. Chaque couche des réseaux de neurones convolutifs sert de filtre de détection pour détecter la présence de motifs ou de caractéristiques spécifiques dans les données d'origine. Les caractéristiques facilement identifiables et interprétables sont détectées dans les premières couches d'un réseau convolutif. Les caractéristiques plus abstraites sont découvertes dans les couches suivantes. En combinant toutes les caractéristiques spécifiques détectées par les couches précédentes dans les données d'entrée, la dernière couche du réseau convolutif est capable de faire une classification extrêmement précise. Les couches entièrement connectées chercheront à créer des notes de classe à partir des activations pour les utiliser dans la classification, effectuant les mêmes fonctions que les ANN conventionnels. Pour améliorer les performances, il est également recommandé d'utiliser ReLu entre ces couches. Le but de la couche entièrement connectée est de pouvoir utiliser ces caractéristiques pour classifier

l'image d'entrée dans différentes classes en fonction de l'ensemble de données d'apprentissage. (DAHAH, 2020)

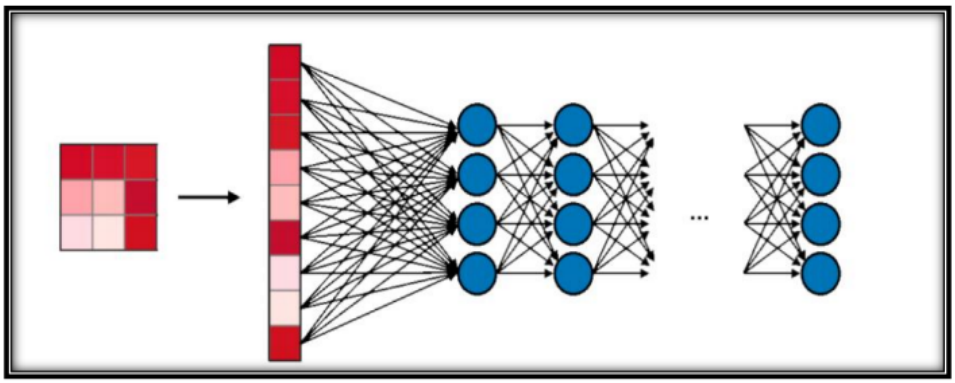


Figure I.19 : un exemple d'une opération entièrement connectée (FC) *(Kabouche, 2022)*

I.7.3.6 Couche de perte

La couche de perte spécifie comment le réseau pénalise l'écart entre le signal attendu et réel. Elle est généralement la dernière couche dans le réseau. Diverses fonctions adaptées à différentes tâches peuvent y être utilisées. La perte "Softmax" est utilisée pour prédire une seule classe parmi K classes mutuellement exclusives. La perte d'entropie croisée sigmoïde est utilisée pour prédire K valeurs de probabilité indépendantes dans [0,1]. La perte euclidienne est utilisée pour régresser vers des valeurs réelles. (Boughaba & Boukhris, 2016-2017)

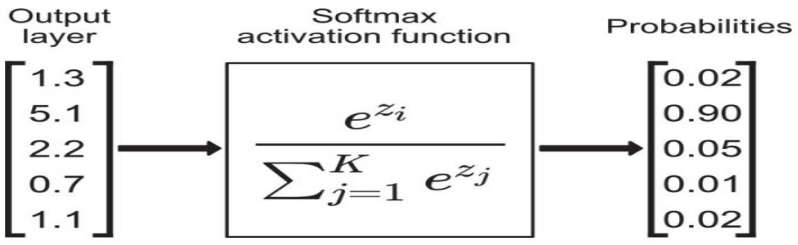


Figure I.20 : Exemple de fonction softmax*(Medium)*

I.7.4 Comprendre la complexité du modèle

Afin d'évaluer la complexité d'un modèle, il est souvent utile de déterminer le nombre de paramètres que comporte son architecture. Dans une couche donnée d'un réseau de neurones convolutifs, cela se fait de la manière suivante :

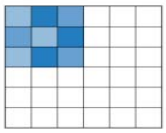
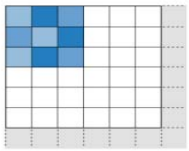
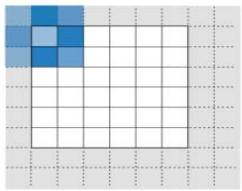
Configuration	Valide	Pareil	Total
Valeur	$P = 0$	$P_{\text{start}} = \left\lfloor \frac{S\lceil \frac{I}{S} \rceil - I + F - S}{2} \right\rfloor$ $P_{\text{end}} = \left\lceil \frac{S\lceil \frac{I}{S} \rceil - I + F - S}{2} \right\rceil$	$P_{\text{start}} \in [\![0, F-1]\!]$ $P_{\text{end}} = F - 1$
Illustration			
But	• Pas de padding • Enlève la dernière opération de convolution si les dimensions ne collent pas	• Le padding tel que la feature map est de taille $\lceil \frac{I}{S} \rceil$ • La taille de sortie est mathématiquement satisfaisante • Aussi appelé 'demi' padding	• Padding maximum tel que les dernières convolutions sont appliquées sur les bords de l'entrée • Le filtre 'voit' l'entrée du début à la fin

Tableau I.1 : Formules pour calculer le nombre de paramètres dans les couches CONV et FC *(Shervine Amidi)*

I.8 Overfitting et Underfitting

Les principales raisons pour lesquelles les modèles prédictifs des algorithmes d'apprentissage automatique fonctionnent mal sont le sur ajustement et le sous-ajustement. Nous examinerons les définitions de ces deux termes.

Underfitting

Lorsqu'un modèle statistique ou une méthode d'apprentissage automatique est trop basique pour représenter avec précision les subtilités des données, on dit qu'il y a un sous-ajustement. Une mauvaise performance sur les données d'entraînement et de test indique que le modèle est incapable d'apprendre l'ensemble d'entraînement de manière efficace. En d'autres termes, les modèles sous-ajustés sont erronés, en particulier lorsqu'ils sont utilisés sur des données non observées. Cela se produit principalement lorsque nous utilisons des modèles très basiques avec des présomptions trop simplifiées. Nous devons utiliser des modèles plus sophistiqués avec une meilleure représentation des caractéristiques et moins de régularisation pour surmonter le problème de sous-ajustement dans le modèle(geeksforgeeks).

Overfitting

Lorsqu'un modèle statistique ne parvient pas à produire des prédictions fiables sur des données de test, il est considéré comme sur ajusté. Un modèle commence à apprendre du

bruit et des entrées de données erronées dans notre ensemble de données lorsqu'il est entraîné avec une grande quantité de données. De plus, une variance élevée est le résultat des tests utilisant des données de test. Ensuite, en raison du bruit et d'une abondance de détails, le modèle ne parvient pas à classer correctement les données. Les techniques non paramétriques et non linéaires sont les causes du sur ajustement car elles offrent aux algorithmes d'apprentissage automatique une plus grande latitude dans la construction de modèles à partir de l'ensemble de données, ce qui leur permet de créer des prédictions extrêmement improbables. Si nous avons des données linéaires, nous pouvons éviter le sur ajustement en utilisant un algorithme linéaire, si nous utilisons des arbres de décision, nous pouvons utiliser des paramètres comme la profondeur maximale (geeksforgeeks).

I.9 Régularisation

Il est nécessaire pour un modèle d'apprentissage profond de généraliser ses prédictions à l'ensemble de données et d'améliorer son entraînement. L'outil pour cela s'appelle la régularisation. La régularisation est un ensemble de techniques qui aide à prévenir le sur ajustement (manque de généralisation) et à maximiser le processus d'apprentissage d'un modèle d'apprentissage profond. Plusieurs outils sont disponibles pour régulariser, nous parlerons de deux de ces outils:

- Batch Normalization
- Dropout

I.9.1 Batch Normalization

La normalisation par lots, qui "aide à combattre le gradient qui disparaît et facilite l'apprentissage". Il s'agit surtout d'une aide à l'apprentissage plutôt que d'une régularisation destinée à prévenir la sur apprentissage(Quora).

I.9.2 Dropout

Le dropout, qui force le modèle à généraliser en supprimant certains neurones à chaque lot. J'ai trouvé une excellente comparaison avec le fonctionnement d'une entreprise. Si chaque employé est considéré comme un neurone, alors une entreprise peut être considérée comme ayant un modèle sans dropouts, ce qui signifie que personne n'est jamais absent ou malade. Cela vous permet d'attribuer à chaque personne une tâche spécifique et très précise. En réalité, les travailleurs peuvent être malades ou en vacances. Si chaque travailleur est spécialisé dans une tâche particulière, il y aura un problème de production en cas d'absence puisque personne ne peut le remplacer. L'objectif du dropout est de "forcer" les travailleurs à manquer le travail afin que d'autres apprennent à prendre leur place. En raison de cela, l'entreprise n'est pas entravée et peut continuer à fonctionner en cas de maladie. Avec un réseau de neurones, cela fonctionne de manière similaire (Quora).

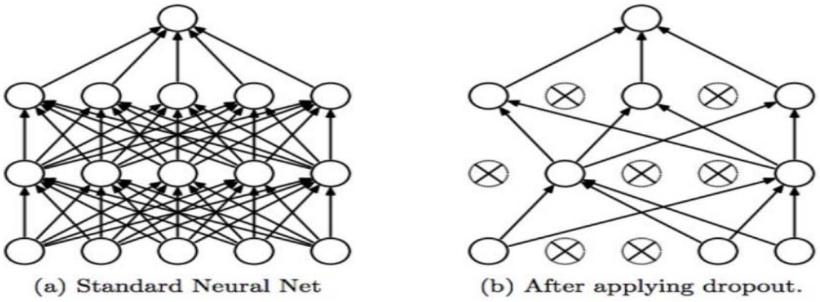


Figure I.21 : Avant et après l'application du Dropout *(ResearchGate)*

I.10 Transfert d'apprentissage

Comme l'apprentissage profond est devenu plus populaire, le transfert d'apprentissage a été extrêmement réussi. En effet, une quantité importante de temps de calcul et de ressources est souvent nécessaire pour les modèles utilisés dans ce domaine. Le transfert d'apprentissage peut créer efficacement des modèles solides et résoudre des problèmes complexes en vision par ordinateur et en traitement du langage naturel (NLP) en commençant par des modèles pré-entraînés (DataScientest).

I.11 Différents types d'architectures CNN

Architecture	Année	Caractéristiques Clés	Cas d'Utilisation
LeNet	1998	Premières applications réussies des CNN, 5 couches (alternant entre convolution et pooling), Utilisation des fonctions d'activation tanh/sigmoïde	Reconnaissance de caractères manuscrits et imprimés par machine
AlexNet	2012	Plus profond et plus large que LeNet, Utilisation de la fonction d'activation ReLU, Implémentation de couches de dropout, Utilisation de GPUs pour l'entraînement	Tâches de reconnaissance d'images à grande échelle
ZFNet	2013	Architecture similaire à AlexNet, mais avec différentes tailles de filtres et nombres de filtres, Techniques de visualisation pour comprendre le réseau	Classification ImageNet

Architecture	Année	Caractéristiques Clés	Cas d'Utilisation
VGGNet	2014	Réseaux plus profonds avec des filtres plus petits (3×3), Toutes les couches de convolution ont la même profondeur, Configurations multiples (VGG16, VGG19)	Reconnaissance d'images à grande échelle
ResNet	2015	Introduction de "connexions de saut" ou "raccourcis" pour permettre l'entraînement de réseaux plus profonds, Configurations multiples (ResNet-50, ResNet-101, ResNet-152)	Reconnaissance d'images à grande échelle, a remporté la 1ère place dans l'ILSVRC 2015
GoogleLeNet	2014	Introduction du module Inception, qui permet une computation plus efficace et des réseaux plus profonds, versions multiples (Inception v1, v2, v3, v4)	Reconnaissance d'images à grande échelle, a remporté la 1ère place dans l'ILSVRC 2014
MobileNets	2017	Conçus pour les applications de vision mobile et embarquée, Utilisent des convolutions séparables en profondeur pour réduire la taille et la complexité du modèle	Applications de vision mobile et embarquée, détection d'objets en temps réel

Tableau I.2Différents types d'architectures CNN *(Analytics Yogi)*

I.12 Conclusion

La définition, les architectures et d'autres termes clés associés à l'apprentissage profond ont été abordés dans ce chapitre. Ensuite, nous nous sommes concentrés sur les réseaux neuronaux convolutifs CNN, qui fournissant des outils puissants pour l'extraction de caractéristiques, l'apprentissage de représentations et la modélisation de types de données complexes. En tirant parti des CNN, les systèmes de recommandation peuvent offrir des recommandations plus précises, pertinentes et personnalisées, améliorant ainsi la satisfaction et l'engagement des utilisateurs.

Cependant, les réseaux neuronaux convolutifs présentent certains inconvénients. Tout d'abord, il peut être difficile d'évaluer les hyperparamètres du réseau à l'avance. Le nombre de couches, le nombre de neurones par couche ou les différentes connexions entre les couches sont, en réalité, des facteurs critiques principalement déterminés par une intuition solide ou par une série d'expériences et de calculs d'erreurs chronophages utilisés pour atteindre nos objectifs.

Dans le prochain chapitre, nous parlerons des systèmes de recommandation, des approches de filtrage, des mesures de similarité et de certains travaux clés dans le domaine.

Chapitre II

Les Systèmes de recommandation

II.1. Introduction

Le développement de l'internet nous a permis d'être au milieu d'une surcharge d'informations. Par exemple, une personne qui souhaite lire un cours se retrouve confronté à un grand nombre de propositions de cours. Cela rend le choix d'un cours très difficile. Des systèmes de recommandation ont émergé pour résoudre ce problème. Nous commençons ce chapitre en définissant ce qu'est un système de recommandation. Ensuite, nous présenterons les trois approches de filtrage qui permettent la recommandation. Nous décrirons et démontrerons les nombreuses mesures de similarité qui permettent aux systèmes de recommandation d'établir des liens entre les concepts. Enfin, nous mentionnerons quelques travaux significatifs dans le domaine ainsi que les limitations et les inconvénients des systèmes de recommandation.

II.2. Historique

Dans les premiers jours de l'informatique, les systèmes de recommandation étaient largement reconnus. Le "Système de Lentille d'Information" (Goldberg, Nichols, OKI, & Terry, 1992) est considéré comme le premier système de recommandation de ce type. À l'époque, la liste de distribution basée sur les groupes d'intérêt était la solution la plus populaire pour le partage d'informations dans un environnement de messagerie électronique.

La première définition du filtrage a également été donnée par Malone : "Bien que le terme ait une connotation littérale de laisser des choses de côté (filtrage négatif: suppression), nous l'utilisons ici dans un sens plus général, qui consiste à sélectionner des éléments dans un ensemble plus large de possibilités (filtrage positif: sélection)".

Le terme "filtrage collaboratif" par le système "Tapestry" a été introduit dans la littérature académique (Goldberg, Nichols, OKI, & Terry, 1992). Le système de recommandation a été créé en 1992 par le centre de recherche "Xerox" aux États-Unis et a été intégré à une application de messagerie électronique pour permettre aux utilisateurs de générer des recherches permanentes basées sur des annotations utilisateur (tags).

Quelques années plus tard, en 1994 et 1995, divers systèmes de recommandation académiques ont été créés, notamment le système de recommandation musicale Ringo et le système de recommandation de nouvelles et de films créé par Group Lens (Goldberg, Nichols, OKI, & Terry, 1992). Le filtrage collaboratif de livres (HAL thesis), de films, d'émissions de télévision, de pages web, d'articles et de liens Internet constitue également la base de ces deux plateformes.

Ensuite, afin d'améliorer sa position en tant que service de recommandation de films, Netflix a introduit le Prix Netflix en 2006. En conséquence, il propose désormais plus de 17 000 films dans sa bibliothèque. De nos jours, les systèmes de recommandation sont largement utilisés dans un large éventail d'applications web.

Les systèmes de recommandation ont ensuite gagné en popularité dans d'autres domaines d'application grâce au développement de l'Internet et des applications web. Nous pouvons citer :

- Les systèmes de recommandation de films, tels que Mobile et Eachmovie.
- Les systèmes de recommandation de livres (Bookcrossing).
- Les systèmes de recommandation musicale (LastFM6).
- Les systèmes de recommandation d'articles de presse.
- Les systèmes de recommandation de blagues.
- Les systèmes de recommandation introduits sur les sites de commerce électronique (Amazon).
- Les systèmes de recommandation de restaurants.
- Les systèmes de recommandation intégrés dans les extranets documentaires (Extranet documentaire du Crédit Agricole).
- Les systèmes de recommandation intégrés aux moteurs de recherche (le moteur de recherche AOL).
- Les systèmes de recommandation mis en œuvre sur les sites de recrutement (JobFinder).
- Les systèmes de recommandation bibliographique.

Dans chaque système de recommandation créé jusqu'à présent, la collecte de données sur les utilisateurs et/ou les objets est une étape cruciale du processus de personnalisation. Les types de données que les systèmes de recommandation peuvent utiliser et les problèmes qui surviennent lors de la collecte de données sont détaillés dans la section suivante.

II.3 Définition des systèmes de recommandation

Étant donné la variété des classifications proposées pour les systèmes de recommandation, il existe de nombreuses façons de les définir ; cependant, la définition générale de Robin Burke définit les systèmes de recommandation comme suit :

"Des systèmes capables de fournir des recommandations personnalisées pour guider l'utilisateur vers des ressources intéressantes et utiles au sein d'un grand espace de données."

Un système de recommandation, parfois appelé système de filtrage d'informations, est un système qui filtre les données entrantes en fonction des besoins uniques de chaque acteur. En d'autres termes, un système de filtrage rassemble, choisit, catégorise et recommande à l'utilisateur des informations susceptibles de correspondre à ses intérêts à

long terme afin de personnaliser sa recherche d'informations dans un certain domaine d'intérêt.

L'utilisateur et l'élément sont les deux entités fondamentales que l'on trouve dans tous les systèmes de recommandation. L'individu qui utilise un système de recommandation, fournit des retours d'informations sur différents produits et reçoit de nouvelles recommandations du système est appelé "utilisateur". Le nom de base de ce que le système suggère aux utilisateurs est un élément.

Les données d'entrée d'un système de recommandation sont déterminées par le type d'algorithme de filtrage utilisé. Ils tombent généralement dans l'un des groupes suivants :

• Les estimations, également appelées votes, transmettent les opinions des utilisateurs sur les articles (par exemple, de 1 très mauvais à 5 excellent).

• Les données démographiques : comprennent des détails sur le pays de l'utilisateur, son niveau d'éducation, son âge et son sexe. Ce type d'information est généralement spécifiquement collecté et est souvent difficile d'accès.

• Les données de contenu sont issues d'une analyse textuelle des documents relatifs aux éléments évalués par l'utilisateur. Pour déterminer un profil utilisateur, les caractéristiques qui ont été extraites de l'analyse sont entrées dans le processus de filtrage.

Le système de recommandation calcule les recommandations en utilisant des profils qui indiquent des préférences généralement stables des utilisateurs pour effectuer le filtrage. Pour cela, il prédit les scores que l'utilisateur est susceptible d'attribuer au contenu. Le système de recommandation utilise les retours d'information de l'utilisateur sur la pertinence du contenu (documents) qu'il reçoit pour ajuster ce profil au fil du temps. La fonction de jugement du système, par exemple, analyse le flux de documents entrants dans la figure 1.1 pour recommander les documents que l'utilisateur choisit en fonction de son profil.

Pour aider le système à mieux comprendre les besoins d'information de l'utilisateur et à fournir des recommandations plus pertinentes, l'utilisateur doit également évaluer régulièrement les recommandations.

Les trois parties suivantes constituent un système de recommandation :

• Les producteurs : Ce sont ceux qui feront des recommandations, ils "fourniront" les données aux utilisateurs.

• Le module de calcul : C'est l'algorithme lui-même. À l'entrée se trouvent toutes les données et la demande et à la sortie les différentes recommandations.

• Le consommateur : C'est celui qui demande la recommandation.

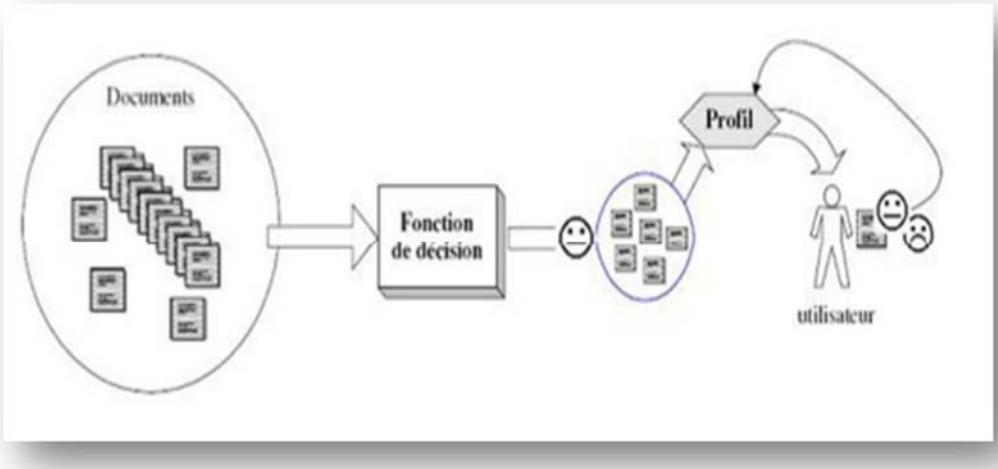


FIGURE II.1 : Le schéma général du filtrage d'informations (SENANTIC SCHOLAR).

II.4 Objectifs des systèmes de recommandation

Un système de recommandation cherche à offrir des ressources pertinentes à un utilisateur selon ses choix. Ainsi, ce dernier raccourcit son temps de recherche mais reçoit également des recommandations du système qu'il n'aurait pas pensé à rechercher par lui-même.

Des systèmes comme ceux de l'industrie du commerce électronique ont été établis en partie en raison de la croissance et de la popularité du Web. Par exemple, les sites de commerce électronique bien connus tels qu'Amazon ainsi que le moteur de recherche d'articles de référence CiteSeer.

Au début, les systèmes de recommandation peuvent être envisagés comme une solution pour les personnes qui ont du mal à décider lorsqu'elles utilisent un moteur de recherche d'informations "classique".

La base de la recherche d'informations est l'idée d'indexer les données pour faciliter les demandes des utilisateurs. La recherche documentaire, en particulier, relève du domaine de la recherche d'informations et consiste à interroger une base documentaire à l'aide de requêtes en langage naturel ou de recherches par mots-clés (également appelées requêtes ad hoc).

Les moteurs de recherche Web, tels que Google et Yahoo!, sont des exemples bien connus d'outils qui utilisent les principes de ce domaine. Les utilisateurs saisissent leur requête de recherche (ensemble de mots-clés) pour trouver les informations qu'ils recherchent. L'index de tous les documents de la base de données du moteur de recherche est ensuite croisé avec ces mots-clés. L'utilisateur se voit alors présenter un groupe de résultats quelque peu liés à sa requête initiale.

La surabondance d'informations que l'utilisateur doit trier est la difficulté fondamentale de ce type de technologie (résultats de recherche d'informations). Par exemple, la plupart du temps, les utilisateurs arrêtent de consulter des informations potentiellement pertinentes et se contentent de parcourir la première page des résultats des moteurs de recherche, en ignorant les autres. Les recommandations n'ont aucun impact sur cette situation, l'un des objectifs des systèmes de recommandation est de parcourir de manière transparente cette grande quantité de données pour l'utilisateur.

En ce qui concerne les recommandations, la création de profils d'utilisateurs est la première étape dans la recherche documentaire. En fait, comme le conseille spécifiquement Google, l'utilisateur est désormais perçu pour évaluer les résultats de sa requête à la lumière de ses recherches passées et des pages qu'il a visitées. C'est un système qui reconnaît d'abord l'utilisateur avant d'examiner son comportement. Cette méthode, également connue sous le nom d'analyse de traces, constitue l'un des principaux piliers du filtrage d'informations qui sous-tend les systèmes de recommandation.

Cela établit une distinction entre les systèmes de recommandation, où la participation de l'utilisateur n'est pas spécifiquement induite par les moteurs de surveillance, et les systèmes de recherche d'informations, où l'utilisateur demande expressément des conseils et de l'aide pour prendre des décisions appropriées.

II.5 Classification des systèmes de recommandation

De nombreuses stratégies et techniques ont été proposées pour classer les systèmes de recommandation. Plusieurs termes sont parfois utilisés pour désigner la même stratégie ou méthodologie. Pour cette raison, plusieurs chercheurs ont proposé une nomenclature unifiée, comme on peut le voir dans la Figure II.2 et ont exprimé un intérêt pour la classification de différentes Figure II.2 Les catégories suivantes sont répertoriées :

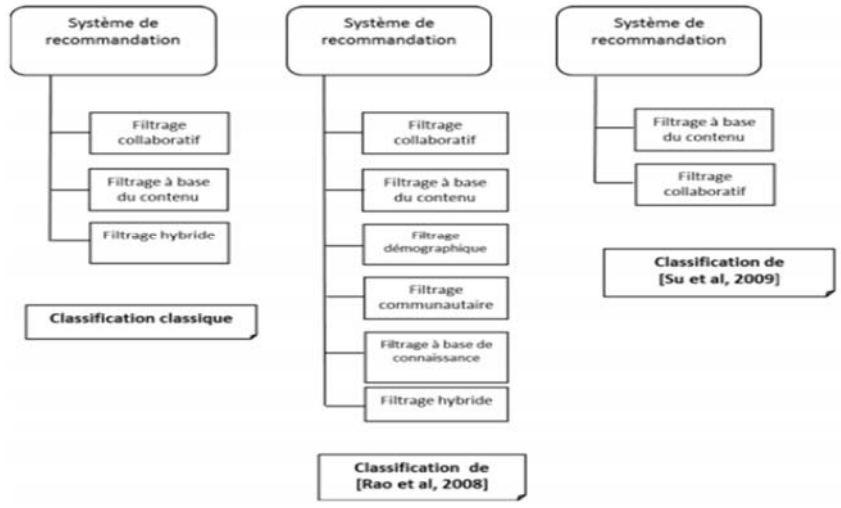


FIGURE II.2 : Principales classifications des systèmes de recommendation(SENANTIC SCHOLAR).

II.5.1 Classification classique

Cette classification de (Adomavicius & Tuzhilin, 2005), a créé un voyage extrêmement fascinant pour recueillir les nombreux points de vue. De plus, ce dernier forme la base d'une quantité croissante de travail. Trois catégories de filtrage ont été distinguées : le filtrage basé sur le contenu, le filtrage hybride et le filtrage collaboratif.

II.5.2 La classification de (Su & Khoshgoftaar, 2009)

Dans les systèmes coopératifs, c'est une classification. La classification des approches hybrides en méthodes de collaboration hybride est proposée par les auteurs comme une sous-classification. Le filtrage collaboratif est divisé en trois groupes par Su et al. (Su & Khoshgoftaar, 2009):

• Techniques de filtrage collaboratif basées sur la mémoire : pour les k voisins les plus proches.

• Méthodes de filtrage collaboratif basées sur un modèle comprenant une gamme de méthodes telles que la factorisation de matrices, le regroupement, les réseaux bayésiens et les processus de prise de décision de Markov.

• Filtrage collaboratif hybride : une méthodologie qui mélange une ou plusieurs autres approches avec une méthode de suggestion de filtrage collaboratif.

II.5.3 La classification de (Rao, 2008)

Cette classification repose sur la source d'information consultée. Rao et Talwar ont fourni une liste complète de systèmes de recommandation pour de nombreux domaines d'application, tant dans des environnements académiques qu'industriels. Basés sur les données utilisées, ils peuvent être largement divisés en six catégories :

- le filtrage coopératif
- le filtrage basé sur le contenu
- le filtrage hybride
- le filtrage démographique
- le filtrage basé sur la communauté
- le filtrage basé sur les connaissances.

II.6 Etude des algorithmes de recommandations

L'étude des algorithmes de recommandations a apporté des centaines d'algorithmes qui peuvent être classés sur base de leurs approches. C'est sur cette base hiérarchique que nous allons explorer les algorithmes les plus connus. Seul la section du deeplearning sera

étudiée séparément étant donné l'importance prise par ce domaine. Dans un premier temps, nous allons explorer le filtrage basé sur le contenu qui présente les algorithmes les plus simples. Nous étudierons ensuite l'approche basée sur le filtrage collaboratif qui représente la partie la plus importante de part ses performances et sa large adoption. Enfin, j'expliquerai brièvement l'approche hybride qui consiste à mélanger les recommandations des deux approches précédentes dans le but de pallier à certains défauts.

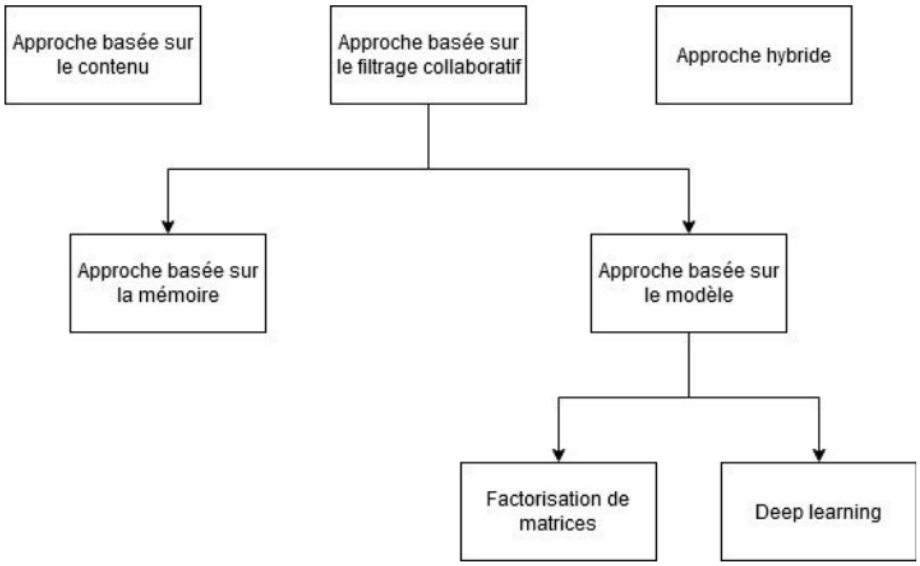


FIGURE II.3 : Les différentes approches des systèmes de recommandations.

II.6.1 Recommandation basée sur le filtrage collaboratif

Le filtrage collaboratif (Collaborative Filtering « CF ») a pour principe d'exploiter les évaluations faites par des utilisateurs sur certains documents (contenus), afin de recommander ces mêmes documents à d'autres utilisateurs, et sans qu'il soit nécessaire d'analyser le contenu des documents.

Tous les utilisateurs du système de filtrage collaboratif peuvent tirer profit des évaluations des autres en recevant des recommandations pour lesquelles les utilisateurs les plus proches ont émis un jugement de valeur favorable, et cela sans que le système dispose d'un processus d'extraction du contenu des documents. Grace à son indépendance vis-à-vis de la représentation des données, cette technique peut s'appliquer dans les contextes où le contenu est soit indisponible, soit difficile à analyser, et en particulier elle peut s'utiliser pour tout type de données : texte, image, audio et vidéo.

De plus, l'utilisateur est capable de découvrir divers domaines intéressants, car le principe du filtrage collaboratif ne se fonde absolument pas sur la dimension thématique des profils, et n'est pas soumis à l'effet « entonnoir ».

Un autre avantage du filtrage collaboratif est que les jugements de valeur des utilisateurs intègrent non seulement la dimension thématique mais aussi d'autres facteurs relatifs à la qualité des documents tels que la diversité, la nouveauté, etc.

Le CF souffre de plusieurs gros problèmes. Le problème principal étant le démarrage à froid : c'est le fait qu'un utilisateur doit voter sur beaucoup d'objet avant d'obtenir les recommandations.

II.6.1.1 Processus du filtrage collaboratif

Le processus du filtrage collaboratif suit les étapes données ci-dessous :

II.6.1.1.1 Evaluation des recommandations

Selon le principe de base du filtrage collaboratif, les utilisateurs doivent fournir leurs évaluations sur des documents afin que le système forme les communautés. Evaluer une recommandation peut se faire de façon explicite ou implicite, comme suit :

- **Explicite** : L'utilisateur donne une valeur numérique sur une échelle donné (par exemple de 1 à 5, ou de 1 à 10, etc.), ou bien, une valeur qualitative de satisfaction, par exemple, mauvaise, moyenne, bonne et excellente.
- **Implicite** : Le système induit la satisfaction de l'utilisateur à travers ses actions. Par exemple, le système estimera qu'une recommandation supprimée correspond à une évaluation tés mauvaise, alors qu'une recommandation imprimée ou sauvegardée peut être interprétée comme une bonne évaluation.

II.6.1.1.2 Formation des communautés

Le processus de formation des communautés est le noyau d'un système de filtrage collaboratif. Pour chaque utilisateur, le système doit calculer sa communauté, généralement cela se fait par la proximité des évaluations des utilisateurs. Pour ce faire, on peut calculer, dans un premier temps, la proximité entre un utilisateur donné et tous les autres. Ensuite, et afin de créer contrairement la communauté de l'utilisateur, on applique la méthode des voisins les plus proche en utilisant un seuil pour le niveau de proximité ou un seuil pour la taille maximale de la communauté, en raison de sa performance et sa précision.

II.6.1.1.3 Production des recommandations

Dans ce derniers processus, une fois la communauté de l'utilisateur créée, le système prédit l'intérêt qu'un document particulier peut présenter pour l'utilisateur en s'appuyant sur les évaluations que les membres de la communauté ont faites sur ce même document. Lorsque l'intérêt prédit dépasse un certain seuil, le système recommande le document à l'utilisateur.

II.6.1.1.4 Profils et communautés

Ici, nous discutons les profils bases sur l'historique des évaluations des utilisateurs, ainsi que les communautés, qui sont les deux facteurs clés d'SFC. Le problème de la surcharge d'information peut être pallié par la personnalisation de l'accès aux informations,

en utilisant des profils représentant des intérêts relativement stables des utilisateurs. En d'autres termes les profils des utilisateurs sont utilisés comme des critères persistant dans la recherche d'information.

a-) Profil de l'utilisateur :

Le profil utilisateur est composé de prédicats pondérés. Le poids d'un prédicat exprime son intérêt relatif pour l'utilisateur. Il est spécifié par un nombre réel compris entre 0 et 1. Le profil s'enrichit progressivement au fur et à mesure que l'utilisateur évalue des documents reçus. Outre l e s informations d'identification de base (par exemple, l'identifiant ou des éléments d'état civil), le profil de l'utilisateur peut regrouper des informations très diverses selon les besoins.

Parmi celles-ci, nous pouvons citer :

- Des caractéristiques personnelles pouvant influencer fortement l'interaction (âge,sexe, ….)
- Les intérêts et les préférences générales de l'utilisateur relatives à la tâche àaccomplir, qui permettent une adaptation à ses attentes.
- Qualité. Cette dimension contient tous les facteurs reflétant les préférencesrelatives à la qualité de l'information, comme la disponibilité de données, la concision, le style et la structure du document, etc. Dans cette dimension, nous nous intéressons en particulier à diversité de l'information.
- Sécurité. La dimension de sécurité dans le contexte du filtrage collaboratif, est le niveau de confidentialité concernant tous les autres critères.
- Un historique des interactions avec le service, qui peuvent permettre de modéliser les habitudes comportementales.

b-) Communautés

La notion de communauté dans un système de filtrage collaboratif est définie comme le regroupement des utilisateurs en fonction de l'historique de leurs évaluations, afin que le système calcule des recommandations. Selon cette optique, les profils sont un facteur interactif, alors que les communautés sont considérées comme un facteur interne du système.

II.6.1.1.2 Exemple

Dans la figure II.4, nous schématisons le principe du filtrage collaboratif. On suppose que l'on a des communautés formées par la proximité des évaluations des utilisateurs. Le document d sera recommandé à l'utilisateur u, car ce document est apprécié par la communauté G où se l'utilisateur.

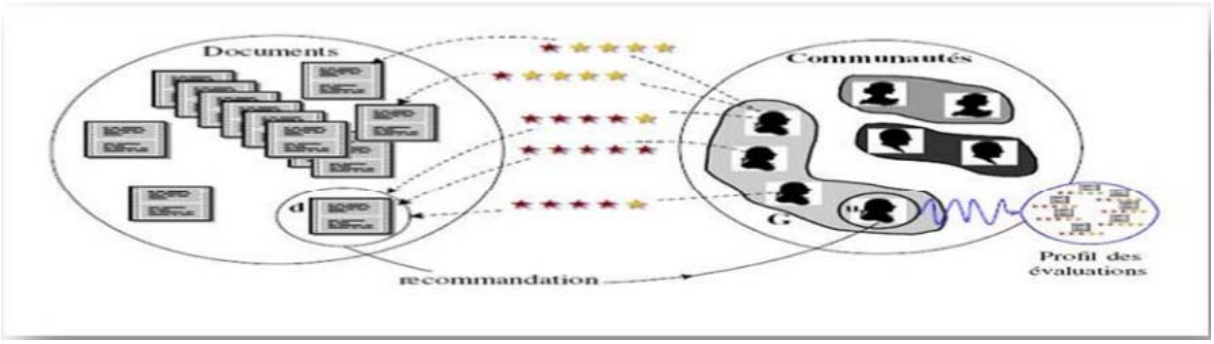


FIGURE II.4 : Principe général du filtrage collaboratif

II.6.1.2 Approche basée sur la mémoire

L'approche basée sur la mémoire utilise les interactions passées des utilisateurs pour calculer les similitudes entre ceux-ci ou entre les items. Pour trouver la note r qu'un utilisateur u donnerait à un item i , l'approche recherche les utilisateurs similaires à u qui ont noté l'item i et calcule la note r en fonction des notes des utilisateurs trouvés à l'étape précédente. Afin de trouver les U utilisateurs les plus similaires à l'utilisateur u , on calcule la similarité sur base des items communément notés avec l'utilisateur comparé en calculant leur distance ou leur similarité. (BASTIN, 2020)

NB : Notez que deux utilisateurs A et B peuvent être considérés comme absolument similaires dans la métrique de similarité cosinus malgré des évaluations différentes. Un exemple serait un utilisateur critique de cinéma qui attribue toujours des notes inférieures à la moyenne, mais dont le classement des éléments de sa liste serait similaire à celui des évaluateurs moyens comme B. Pour tenir compte de ces préférences individuelles des utilisateurs, il faut amener tous les utilisateurs au même niveau en supprimant leurs préjugés. Ceci peut se faire en soustrayant la note moyenne donnée par cet utilisateur à tous les items de chaque item noté par cet utilisateur.

Après avoir déterminé une liste d'utilisateurs similaire à un utilisateur u, on calcule la note r que u donnerait à un certain item i .

On considère que la note r d'un utilisateur pour un item i sera proche de la moyenne des notes attribuées à i par les U utilisateurs les plus similaires à u . La formule mathématique de la note moyenne donnée par U utilisateurs se calcule avec formule dont la version la plus simple est :

$$\hat{r}_{ui} = \frac{1}{n} * \sum_{u' \in U}^{n} r_{u',i} \quad [\text{II.1}]$$

r : représente la note donnée par un utilisateur u à un item i et U représente le groupe d'utilisateurs similaires à u.

Il est également possible de multiplier la note par le degré de similarité entre les deux utilisateurs afin de donner plus de poids aux notes d'utilisateurs fort similaires à u :

$$\hat{r}_{ui} = k * \sum_{u' \in U}^{n} sim(u, u') * r_{u',i} \text{ [II.2]}$$

k : est un facteur de normalisation.

Enfin, on peut également prendre en compte les notes moyennes de l'utilisateur u dans le calcul étant donné que les utilisateurs peuvent avoir tendance à voter différemment les uns des autres :

$$\hat{r}_{ui} = \bar{r}_u + k * \sum_{u' \in U}^{n} sim(u, u') * (r_{u',i} - \bar{r}_{u'}) \text{ [II.3]}$$

ru : est la moyenne des notes de l'utilisateur u pour tous les items notés par u.

Différence entre l'approche basée sur les utilisateurs et les items

Il existe deux approches pouvant être utilisées pour trouver des recommandations avec le filtrage collaboratif. Ces deux approches sont mathématiquement assez similaires, mais il existe une différence conceptuelle entre les deux. Voici comment elles se comparent :

- Basé sur les utilisateurs : pour un utilisateur u , le vote pour un item i , qui n'a pas encore été voté par l'utilisateur u est trouvé en prenant les U utilisateurs les plus similaires qui ont noté l'item i et en calculant le vote basé sur les votes de ces U utilisateurs.
- Basé sur les items : pour un item i , le vote par un utilisateur u , qui ne l'a pas encore voté est trouvé en prenant les I items les plus similaires qui ont été noté par l'utilisateur u en calculant le vote basé sur les votes de ces I items

Dans un système où il y a plus d'utilisateurs que d'items, le filtrage basé sur les items est plus rapide et plus stable que celui basé sur les utilisateurs. Il est efficace car en général, la note moyenne reçue par un item ne change pas aussi rapidement que la note moyenne attribuée par un utilisateur à différents éléments. Il est également connu pour être plus performant que l'approche basée sur les utilisateurs lorsque la matrice de notation est fortement éparse. (BASTIN, 2020)

II.6.1.3 Filtrage collaboratif basé sur un modèle

Comme son nom l'indique, ce type d'algorithme repose sur des modèles afin de minimiser la complexité. Ces modèles estiment ou apprennent un modèle qui est ensuite utilisé pour les prédictions en utilisant la base de données des notes des utilisateurs. Pour simplifier la complexité, ils peuvent être construits sur des classificateurs pour produire des classes.

II.6.1.3.1 La factorisation de matrices

La factorisation de matrices, ou décomposition de matrices est une méthode qui permet d'accélérer la recherche de recommandations. L'idée derrière la factorisation matricielle est de représenter les utilisateurs et les items dans un espace latent de dimension inférieure à celui de base en décomposant la matrice initiale en plusieurs autres matrices.

Cette réduction de dimensionnalité permet de faire face aux problèmes de scalabilité du traitement de ces matrices qui peuvent se révéler volumineuses et très éparses. Pour retrouver la matrice originale, il suffira de faire le produit de ces matrices entre elles. Cette décomposition d'une matrice creuse en deux matrices denses de dimensions inférieures nous permet d'économiser du stockage et d'accélérer les calculs. Ces avantages en ont fait une méthode très utilisée dans le domaine du filtrage collaboratif. (BASTIN, 2020)

II.6.1.3.2 Le modèle des facteurs latents

Le modèle des facteurs latents représente les items et les utilisateurs par des vecteurs de caractéristiques de même taille, les facteurs latents. Plus la correspondance entre les facteurs latents d'un utilisateur et d'un item est grande, plus il y a de chance pour que le film corresponde aux goûts de l'utilisateur. Bien que ces facteurs latents représente des caractéristiques, il ne faut pas faire l'erreur de vouloir mettre une étiquette dessus car nous ne pouvons faire que des suppositions sur leurs significations.

L'objectif principal des facteurs latents est d'approximer au mieux la matrice des relations user-item. L'estimation de cette matrice R est égale à la multiplication de la matrice des $\hat{R}$ facteurs latents des utilisateurs P par la matrice transposée des facteurs latents des items Q . (BASTIN, 2020)

$$R \simeq P * Q^{T} = R \text{ [II.4]}$$

Approximation de la matrice R par la multiplication de la matrice P et Q transposée.

Pour trouver la correspondance entre un utilisateur et un film, on multiplie leurs facteurs latents :

$$\hat{R}_{ui} = P_u^T * q_i = \sum_{k=1}^{k} p_{uk} * q_{ki} \text{ [II.5]}$$

Aproximation d'une relation user-item par la multiplication des facteurs latents de l'utilisateur u et de l'item i. k représente le nombre de facteurs latents.

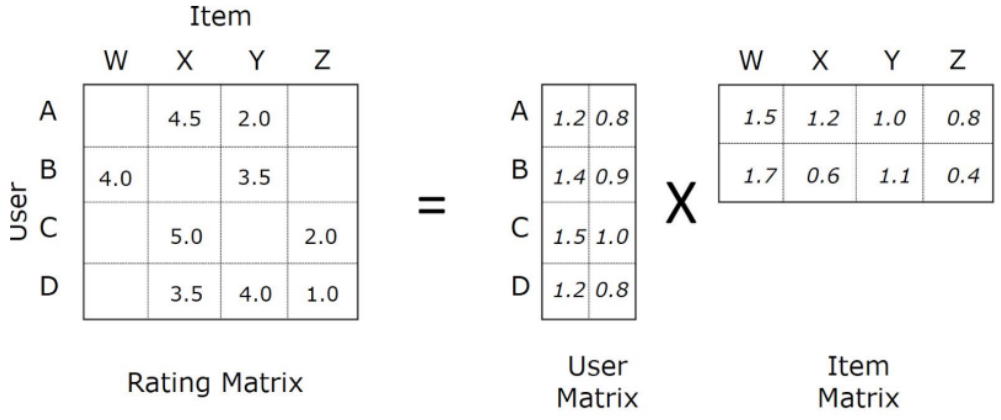


FIGURE II.5 : Représentation de la transformation d'une matrice creuse en matrices denses à 2 facteurs latents (BASTIN, 2020)

Méthodes d'apprentissage

Afin de trouver les facteurs latents qui estiment le mieux la matrice R̂, les matrices Q et P sont d'abord initialisées avec des valeurs aléatoires. Ensuite, on minimise l'erreur quadratique entre la relation estimée et la réelle grâce la formule :

$$min_{q_i,p_u} \sum_{u.i \in k} (r_{ui} - \hat{r}_{ui})^2 + \lambda \left(\|q_i\|^2 + \|p_u\|^2\right) [\text{ II.6}]$$

Formule de la minimisation de la différence entre la relation estimée et celle contenue dans la matrice originelle. Le deuxième terme est une régularisation qui permet de ne pas surestimer les données observées.

Afin de minimiser cette erreur, nous devons passer par un algorithme d'apprentissage automatique qui va progressivement diminuer l'erreur des estimations. Les deux techniques les plus utilisées sont l'apprentissage par descente de gradient stochastique et l'apprentissage par les moindres carrés alternés.

II.6.2. Recommandation basée sur le contenu :

Les évaluations des utilisateurs d'une collection de documents ou d'éléments servent de base aux systèmes de recommandation basés sur le contenu. L'objectif suivant est de comprendre la logique derrière votre décision de considérer un certain élément comme important ou non.

On peut le considérer comme un système de recherche d'informations exploitant le profil de l'utilisateur.

Les systèmes de recommandation basés sur le contenu présentent l'avantage de vous permettre de lier des produits au profil de l'utilisateur, notamment en appliquant des méthodes d'intelligence artificielle. L'utilisateur est autonome et peut recevoir des recommandations même lorsqu'il est le seul à utiliser l'application. Cependant, ce type de système présente plusieurs inconvénients :

- L'effet de "tunnel" : l'utilisateur ne peut pas avoir une large gamme de sujets car ses besoins deviennent de plus en plus spécialisés. Pire encore, puisqu'il n'est pas explicitement inclus dans le profil de l'utilisateur, un nouvel axe de recherche dans une certaine région pourrait ne pas être pris en compte.
- Problème de démarrage à froid : Un utilisateur novice du système trouve difficile de décrire ses intérêts et de créer un profil.

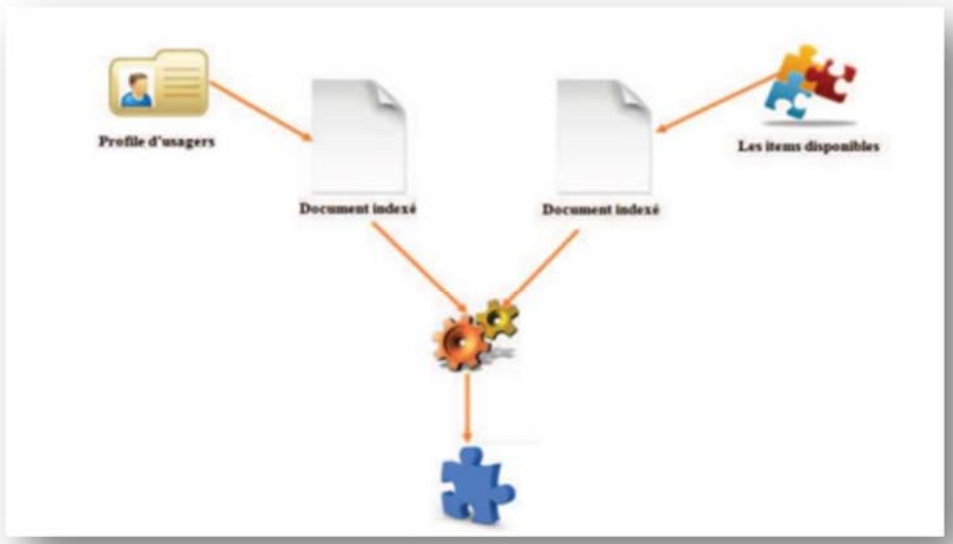


FIGURE II.6 : Recommandation basée sur le contenu (Ait Ahmed & Driss Khodja, 2018)

Il existe deux types de recommandations basées sur le contenu : basées sur des mots-clés et basées sur la sémantique :

II.6.2.1- Recommandation basée sur les mots-clés :

On peut utiliser l'approche de recommandation basée sur le contenu pour suggérer des sites web, des films, des articles de presse, des restaurants, et plus encore. En prenant comme exemple un système de recommandation d'articles scientifiques basé sur le contenu, le système suggérera des publications liées à la génétique à un utilisateur qui lit souvent des articles sur ce sujet. En effet, des termes comme "**ADN**", "**gène**" et "**protéine**" sont fréquemment utilisés dans ces textes.

II.6.2.2- Recommandation basée sur la sémantique :

De nombreuses techniques ont été utilisées pour incorporer la sémantique dans le processus de recommandation. Plusieurs facteurs sont pris en considération lors de l'adressage de ces méthodes :

- **Le type de source de connaissances impliqué (lexique, ontologie, etc.).**
- **Les techniques adoptées pour l'annotation ou la représentation des éléments.**
- **Le type de contenu inclus dans le profil utilisateur.**
- **La stratégie de correspondance entre les éléments et le profil.**

Les systèmes de recommandation basés sur la sémantique se développent en parallèle avec les techniques et les ressources disponibles dans l'espace du Web sémantique.

II.6.3 Filtrage hybride

Lorsque deux ou plusieurs systèmes de recommandation distincts sont combinés, on parle de système hybride. Les recommandations collaboratives et les recommandations basées sur le contenu sont souvent considérées comme complémentaires.

En général, l'hybridation se fait en deux étapes pour résoudre les lacunes de chaque approche lorsqu'elle est utilisée seule et pour capitaliser sur leurs forces :

- Appliquer séparément le filtrage collaboratif et d'autres techniques de filtrage pour produire des recommandations potentielles.
- Combinaison de ces ensembles de recommandations préliminaires en utilisant les méthodes suivantes : combinaison et amélioration des caractéristiques et du niveau moyen , pondération, mélange, cascading, commutation, etc. afin de fournir aux utilisateurs les recommandations finales.

En général, les systèmes hybrides gèrent les profils d'utilisateurs orientés contenu, et la comparaison de ces profils conduit à la création de communautés d'utilisateurs qui permettent le filtrage collaboratif

II.7 Avantages et inconvénients de la recommandation des méthodes système

Le tableau susmentionné énumère les avantages et les inconvénients des techniques conventionnelles employées par les systèmes de recommandation, dans cet exemple, le filtrage communautaire basé sur le contenu, démographique, collaboratif et de base de données.

Techniques	Avantages	Inconvénients
Filtrage Démographique	-Aucun historique d'estimation préalable n'est nécessaire.	- Préoccupation concernant la protection des données. - Utilisateur discernant. - Publication récente.
Filtrage À base de données communautaire	• Qualité s'améliorant proportionnellement au cercle d'amis. • Attribut d'adaptabilité en fonction du nombre de	- Arrivée de nouveaux utilisateurs. - Publication récente.

	relations.	
Filtrage à base du contenu	- Recommandations possibles même avec une petite communauté d'utilisateurs. - Génération de listes de commandes pour un seul utilisateur. - Amélioration progressive de la qualité avec le temps. - Absence de nécessité d'informations sur les autres utilisateurs. - Prise en compte des préférences individuelles des utilisateurs.	• Analyse de contenu nécessaire pour recommander. • Défi de recommandation pour les médias visuels (images et vidéos). • Nécessité d'un profil utilisateur pour la recommandation.
Filtrage collaboratif	- Indépendance vis-à-vis de la connaissance du contenu ou de sa sémantique. - Évaluation possible de la qualité de la recommandation. - Amélioration de la qualité de la recommandation avec un nombre croissant d'utilisateurs.	- Démarrage à froid. - Nouvel article. - Nouvel utilisateur. - Problème de confidentialité et complexité croissante : dans les systèmes avec un grand nombre d'éléments et d'utilisateurs, le calcul augmente de manière linéaire.

tableau II.1 : Avantages et inconvénients des méthodes de recommandation.

II.8.Les systèmes de recommandations basées sur le machine et deep learning

Nous présentons quelques travaux de l'état de l'art basé sur le machine et deep learning dans cette partie .

II.8.1 Filtrage collaboratif utilisant k plus proches voisins (KNN)

En général, les algorithmes de filtrage collaboratif basés sur les voisins se servent de l'ensemble de la matrice des notes des utilisateurs afin de formuler des recommandations. Afin d'utiliser l'algorithme de K plus proche des voisins, il est nécessaire que le système possède une mesure de similarité afin de distinguer les utilisateurs proches de ceux éloignés. On peut tirer cette mesure de similarité de la distance euclidienne, de la mesure du cosinus, de la corrélation de Pearson, et ainsi de suite. Par la suite, lorsqu'il est demandé une nouvelle information, le système va chercher les k utilisateurs les plus proches de l'utilisateur cible. Enfin, une consultation majoritaire est organisée afin de déterminer quel élément sera recommandé (filtrage basé sur les utilisateurs).

•Calcul de la similarité

Le calcul de la similarité a pour objectif de déterminer dans quelle mesure deux utilisateurs sont similaires. Il existe plusieurs façons de calculer cette similarité, cependant les méthodes les plus utilisées sont :

- La mesure de cosinus

$$Sim(A,B) = \frac{\sum_{i=1}^{n} v_{ai} \times v_{Bi}}{\sqrt{\sum_{i=1}^{n} v_{Ai}^2} \times \sqrt{\sum_{j=1}^{n} v_{Bi}^2}} \qquad [II.7]$$

$\boldsymbol{n}$: Nombre d'items communs entre A et B votés par v

$\boldsymbol{vAi}$: Vote de A pour l'item i

$\boldsymbol{vBi}$: Vote de B pour l'item

- Corrélation de Pearson

$$Sim(A,B) = \frac{\sum_j (v_{Ai} - \overline{v}_{Ai})\,(v_{Bi} - \overline{v}_{Bi})}{\sqrt{\sum_j (v_{Ai} - \overline{v}_{Ai})\,(v_{Bi} - \overline{v}_{Bi})^2}} \qquad [II.8]$$

$\boldsymbol{i}$: Nombre d'objets ayant voté à la fois par A et B.

$\boldsymbol{vAi}$: Vote de A pour l'item i.

$\boldsymbol{\overline{v}Ai}$: Moyenne des votes de A.

$\boldsymbol{vBi}$: Vote de B pour l'item i.

$\boldsymbol{\bar{v}Bi}$: Moyenne des votes de B.

Après avoir calculé la similarité, la prédiction est calculée afin de prédire la note de l'utilisateur cible pour un élément.

$$P_{A,j} = \frac{\sum_{i=1}^{n} Sim(A,i) \times v_{i,j}}{\sum_{i=1}^{n} sim(A,i)} \qquad \text{[II.9]}$$

n : Nombre d'utilisateurs présents dans le voisinage de A, ayant déjà voté sur l'item j.

, : Vote de l'utilisateur i pour l'objet j.

II.8.2 Perceptron multicouche

Un perceptron multicouche (multilayer perceptron ou MLP) est un dispositif de perception qui utilise des perceptrons supplémentaires, empilés en plusieurs couches, afin de resoudre des problemesspecefiques.

Pour les systèmes de recommandation, les perceptrons multicouches sont employés afin d'encoder les utilisateurs et les éléments individuellement avant de les fusionner avec un autre perceptron multicouches associé. Les interactions entre les utilisateurs et les éléments sont modélisées, et les éléments les plus représentatifs pour chaque utilisateur ainsi que les attributs d'éléments les plus importants pour chaque utilisateur sont sélectionnés.

Une représentation multicouche est utilisée pour représenter l'interaction entre l'utilisateur et les éléments, où la sortie d'une couche est utilisée comme entrée pour la couche suivante. La couche d'entrée initiale comprend deux vecteurs d'entrée Vu et Vi qui représentent l'utilisateur u et l'objet i. Suite à la couche d'introduction, il existe une couche d'intégration. Cette couche est entièrement reliée, ce qui permet de projeter la représentation séparée sur un vecteur compact. Ensuite, on intègre ces couches d'intégration dans une architecture neuronale multicouche afin de lier les vecteurs latents à des scores de prédiction. Il est également possible d'adapter chaque couche cachée afin de mettre au jour de nouvelles structures cachées à partir des interactions utilisateur-item. La couche finale fournit le score attendu(PINNAPAREDDY, 2018) .

II.8.3 S.R. utilisant le classement personnalisé bayésien (BPR)

En général, les utilisateurs effectuent des achats en ligne en ne parcourant que les premières pages des sites Web. De plus, de plus en plus de personnes se servent de tablettes et de smartphones pour effectuer leurs achats en ligne, il est donc essentiel de créer un classement sur mesure.

Dans la plupart des cas, lorsque l'utilisateur clique sur un objet pour consulter ses détails, cela indique que certaines caractéristiques de l'objet sont attirer. De cette manière, il est possible de supposer que les utilisateurs ont une préférence pour ces éléments par

rapport aux autres et les utilisent pour les classer après avoir parcouru le site Web pendant un certain temps. Le principal objectif du classement personnalisé consiste à offrir à un utilisateur une liste organisée d'éléments (Medium).

Le terme « U » désigne l'ensemble des utilisateurs et le terme « I » désigne l'ensemble des éléments. La figure II.7 suivante illustre la manière dont les données implicites sont traitées lorsqu'il s'agit de recommandations d'éléments.

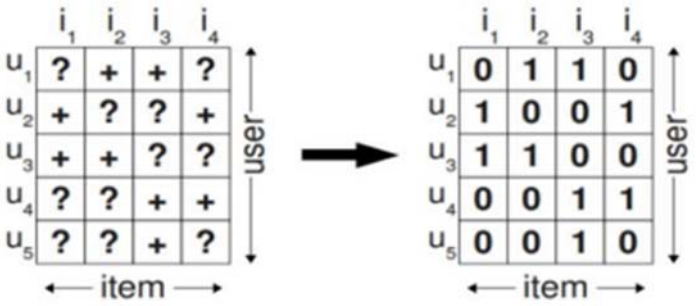


FIGURE II.7 : les interactions existantes entre l'utilisateur et l'item (Medium)

La méthode classique implique de prédire un score **Xui**sur mesure pour un objet, reflétant ainsi la préférence de l'utilisateur pour cet objet. Ensuite, les éléments seront triés en fonction de cette note. Comme illustré dans la figure ci-dessus, tous les échanges existants entre l'utilisateur et l'objet sont classés comme une classe positive (1), tandis que les autres échanges sont classés comme une classe négative (0).
Dans la méthode BPR (Medium), plutôt que de sélectionner un élément, les paires d'éléments sont perçues comme des données d'entraînement. Il serait préférable d'effectuer l'optimisation en se basant sur le classement de ces paires utilisateur-item plutôt que de se concentrer exclusivement sur l'interaction utilisateur-item. L'ensemble des données à prendre en compte est défini de la manière suivante :

$(u, i, j) \in D$s

La sémantique de (u, i, j) ∈Ds est que l'utilisateur u est supposé préférer i à j.

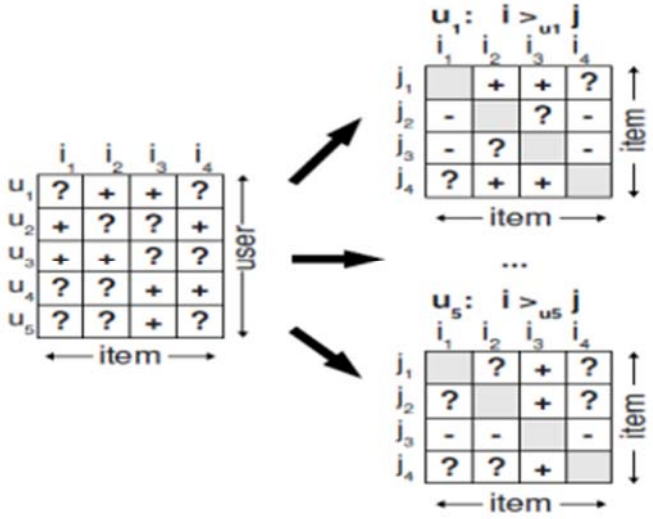


FIGURE II 8 : les préférences par paire spécifiques à l'utilisateur entre une paire d'items (Medium)

La figure II.8 ci-dessus montre que l'utilisateur u1 a observé l'élément i2 mais pas l'élément i1. Par conséquent, l'algorithme suppose que cet utilisateur préfère l'élément i2 à i1 (i 2> i1) et affiche un signe positif. Il n'y a pas de préjugés sur une préférence pour les éléments qui ont tous deux été consultés par un utilisateur et qui sont affichés comme «? ». C'est également le cas pour deux éléments qu'un utilisateur n'a pas encore aperçus (par exemple, l'élément i1 et i4 pour l'utilisateur u1). Au contraire, on peut constater un signe négatif pour (i1, j2) car l'utilisateur préfère l'élément 2 (i2) plutôt que l'élément 1 (i1).

Selon les auteurs (Medium), BPR est composé du critère d'optimisation BPR-OPT ainsi que de l'algorithme LearnBPR pour maximiser l'efficacité. Au lieu de prédire une note spécifique pour chaque élément, il nous suffit de prédire les préférences relatives de l'utilisateur pour toutes les paires (i, j).

BPR-OPT :

$$\sum_{(u,i,j)\in D_s} ln\sigma(\hat{x}_{uij}) - \lambda\|\theta\|^2 \qquad \text{[II.10]}$$

Ou σ est la Fonction logistique sigmoïde :

$$\sigma(x) := \frac{1}{1+e^{-x}} \qquad \text{[II.11]}$$

La fonction $\hat{x}_{uij}$ (θ) dans l'équation précédente est une fonction à valeur réelle qui représente la relation entre l'utilisateur u, l'item i et l'item j. Elle est généralement calculée en utilisant le modèle de factorisation matricielle.

désigne le vecteur de paramètres d'une classe de modèles variables (comme la factorisation matricielle par exemple).

Des paramètres de régularisation spécifiques au modèle sont λ .

La probabilité qu'un utilisateur préfère l'item i à l'item j peut être exprimée de la manière suivante :

$$p(i >_u j|\theta) := \sigma(\hat{x}_{uij}(\theta)) \qquad \text{[II.12]}$$

Dans ce cas, u correspond à la structure de préférence désirée mais latente pour l'utilisateur u. Il convient également de souligner que p (> u | θ) est une fonction de probabilité propre à l'utilisateur.

Fonction de vraisemblance :est une équation calculée à partir des paramètres d'un modèle statistique à partir de données observées.

$$P(\theta| > u)\alpha p(> u|\theta)p(\theta) \qquad \text{[II.13]}$$

Il est nécessaire de maximiser le nombre de prédictions correctes de toutes les paires (i,j), ce qui correspond à maximiser l'AUC, qui correspond à l'aire sous la courbe ROC.

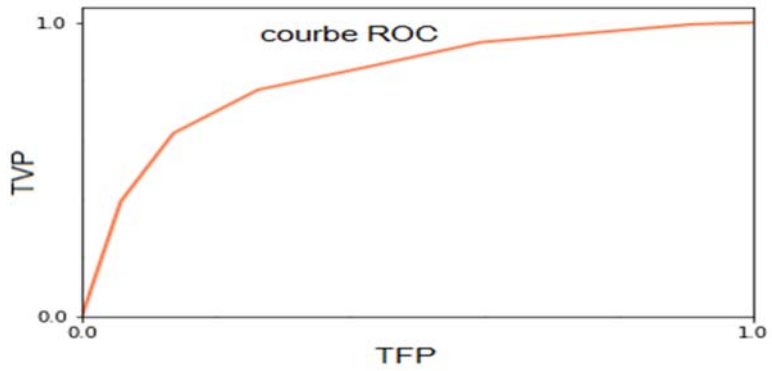


FIGURE II 9 : la courbe ROC (Jeremy Jordan)

Les performances d'un modèle de classification pour tous les seuils de classification sont représentées par une courbe ROC (receiver operating characteristic). Cette courbe représente la proportion de vrais positifs par rapport au taux de faux positifs :
L'équivalent du rappel est le taux de vrais positifs (TVP). C'est ainsi qu'il est défini :

$$TVP = \frac{VP}{VP + FN} \qquad \text{[II.13]}$$

Le taux de faux positifs (TFP) est défini comme suit :

$$TFP = \frac{FP}{FP + VN} \qquad \text{[II.14]}$$

Analogies à l'optimisation AUC (Area Under The Curve) :La formule de l'AUC est présentée dans l'équation ci-dessus, où I est l'ensemble de tous les éléments, et Iu + est l'ensemble des éléments sur lesquels l'utilisateur a cliqué :

$$AUC = \frac{1}{|U|}\sum_{u\in U}\frac{1}{|I_u^+||I/I_u^+|}\sum_{i\in I_u^+}\sum_{j\in|I/I_u^+|}\delta(\hat{x}_{uij} > 0) \qquad \text{[II.15]}$$

L'AUC représente la zone située sous la courbe ROC, représentée par un taux vrai positif comme axe y et un taux faux positif comme axe x pour différents seuils du modèle en question. Autrement dit, la courbe ROC peut être tracée pour un ensemble de recommandations d'articles en calculant le pourcentage de recommandations positives et de recommandations négatives pour chaque seuil.

II .8.4 Le filtrage collaboratif neuronal (NCF)

NCF abréviation de Neural Collaborative Filtering, a tenté d'atteindre les objectifs suivants :

- L'objectif de NCF est de décrire et de généraliser la factorisation matricielle (MF) dans son environnement.
- En utilisant un perceptron multicouche, NCF tente d'acquérir des connaissances sur les interactions entre les utilisateurs et les éléments.

Dans le cadre du filtrage collaboratif neuronal, MF est utilisé pour détecter la corrélation entre les éléments et les utilisateurs. L'analyse des évaluations des utilisateurs sur les éléments à l'entrée permet de prédire comment les utilisateurs évalueraient les éléments, permettant ainsi aux utilisateurs d'obtenir une recommandation de prédiction.

Le filtrage collaboratif neuronal (NCF) a pour objectif de créer une fonction d'interaction optimisée afin de simuler l'interaction des fonctionnalités cachées entre les utilisateurs et les éléments en :

1. Étudier l'interaction entre les fonctionnalités utilisateur et les éléments en utilisant l'architecture de réseau neuronal en employant un Perceptron multicouche (MLP). Cela représente une amélioration par rapport à MF car MLP peut théoriquement apprendre n'importe quelle fonction continue et présente un niveau élevé de non-linéarités (en raison de plusieurs couches).

2. En généralisant et en exprimant la MF en tant que cas spécifique de NCF. Le succès de MF dans le domaine des recommandations apportera une plus grande crédibilité à NCF.

II.8.5 Modèle NeuMF (Neural Matrix Factorization)

En raison de l'augmentation exponentielle des données sur le Web ces dernières années, les réseaux de neurones artificiels et l'apprentissage profond ont été largement utilisés dans divers domaines de l'intelligence artificielle, tels que le traitement d'images, la vision par ordinateur, la reconnaissance vocale et le traitement automatique du langage

naturel (LIU, WANG, LIU, & ALSAADI, 2017). Le domaine de la recommandation n'échappe pas à cette tendance et est également fortement influencé par ces avancées technologiques (FERRARI DACREMA, CREMONESI, & JANNACH, 2019; ZHANG & CHEN, 2020). L'apprentissage profond permet de capturer efficacement les relations non linéaires et complexes entre les utilisateurs et les items, et de découvrir des relations plus abstraites et complexes à partir des données d'interaction entre utilisateurs et items. De nombreuses approches de recommandation basées sur les réseaux de neurones ont ainsi vu le jour (SEDHAIN, MENON, & Sanner, 2015; He, Liao, Zhang, Nie, & Hu, 2017; GUO, TANG, & YE, 2017). Dans cette section, nous présentons en détail le modèle NeuMF(He, Liao, Zhang, Nie, & Hu, 2017), l'une des approches de recommandation basées sur les réseaux de neurones profonds les plus citées. Pour une vue d'ensemble des différents modèles de recommandation utilisant les réseaux de neurones et l'apprentissage profond, le lecteur peut se référer à (ZHANG & CHEN, 2020).

NeuMF est un modèle de filtrage collaboratif basé sur les réseaux de neurones profonds. Cette approche combine la technique de factorisation matricielle décrite dans les sectionsprécédentes avec les réseaux de neurones pour améliorer les performances des modèles basés sur les facteurs latents.

L'architecture générale du NeuMF est illustrée dans la figure 2.6. La couche d'entrée (Input), située en bas du schéma, représente deux vecteurs décrivant respectivement l'utilisateur u et l'item i. Les auteurs de (He, Liao, Zhang, Nie, & Hu, 2017) ont adopté l'encodage one-hotvector pour modéliser u et i. Cet encodage repose sur l'identifiant de l'utilisateur u et de l'item i (par exemple, un vecteur de dimension |U| avec un 1 à l'emplacement correspondant à l'identifiant de l'utilisateur cible et des 0 pour les |U|−1 autres dimensions). Il est à noter que les vecteurs de description des utilisateurs et des items peuvent également être basés sur d'autres informations que les évaluations, si celles-ci sont disponibles, comme les contenus des items.

Au-dessus de la couche d'entrée se trouve la couche de plongement (Embedding). Il s'agit souvent d'une couche entièrement connectée (fullyconnected layer) qui projette la représentation initiale (généralement très creuse, par exemple, un vecteur de |U| dimensions avec |U|−1 zéros et un 1 pour un utilisateur donné) dans un vecteur plus dense (généralement de 64 ou 128 dimensions). Les plongements obtenus respectivement pour l'utilisateur u et pour l'item i sont analogues aux vecteurs latents du modèle des facteurs latents discuté précédemment.

La couche Neural CF, qui suit la couche Embedding, est une architecture de neurones multi-couche utilisée pour aligner les vecteurs latents aux scores de prédiction. Enfin, la dernière couche de sortie (Output) produit la prédiction r^u,i. L'entraînement du modèle est généralement réalisé en minimisant l'erreur (Loss) entre la prédiction r^u,i et la valeur réelle de ru,i.

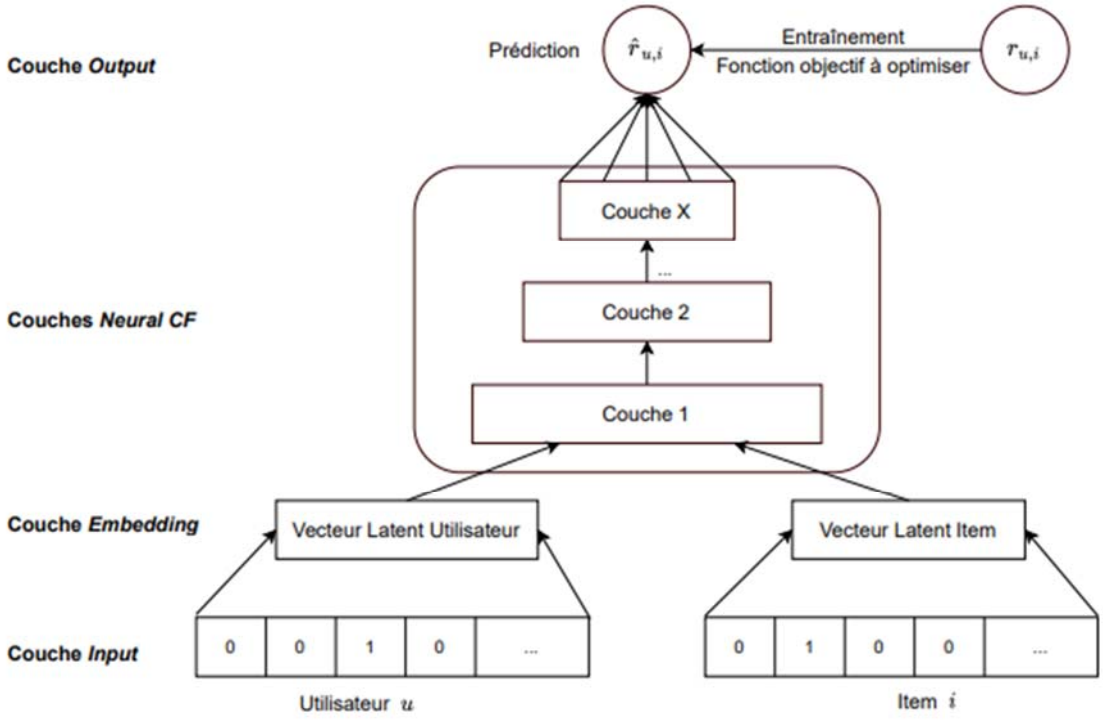


FIGURE II 10 : Architecture générale du modèle NeuMF (ZHANG & CHEN, 2020)

Dans le contexte du filtrage collaboratif, comme expliqué dans la section précédente relative au modèle SVD++ , deux modélisations sont possibles, basées respectivement sur les feedbacks explicites ou implicites. L'architecture de neurones présentée ci-dessus peut s'adapter automatiquement en fonction du type de données (explicites ou implicites) utilisé pendant l'entraînement. Il suffit de choisir la fonction objectif appropriée selon le contexte.

Pour le feedback explicite, la fonction objectif peut être définie pour minimiser l'erreur quadratique entre la prédiction r^u,iet la valeur réelle du rating ru,i, comme dans le cas des modèles de facteurs latents. Dans (He, Liao, Zhang, Nie, & Hu, 2017), les auteurs ont choisi d'apprendre les paramètres du modèle dans le contexte des feedbacks implicites. Autrement dit, $r_{u,i}$= 1 si l'interaction entre l'utilisateur u et l'item i est observée dans le jeu de données, et $r_{u,i}$ = 0 sinon. Ainsi, on peut considérer $r_{u,i}$comme un label : 1 signifie que l'item i est pertinent pour l'utilisateur u, et 0 sinon. La prédiction $\hat{r}_{u,i}$représente ainsi la probabilité que l'item i soit pertinent pour l'utilisateur u.On note R′ l'ensemble des instances de ratings positives, c'est-à-dire les interactions (u,i) observées dans le jeu de données, et R′- l'ensemble des items avec lesquels l'utilisateur u n'a pas interagi, c'est-à-dire les instances négatives. L'objectif est que les ratings prédits pour les éléments de R′− soient proches de 1 et ceux pour les éléments de R′− soient proches de 0. L'utilisation de la fonction logarithmique (log) permet de pénaliser fortement les écarts importants par rapport à ces prédictions attendues. Ainsi, dans le contexte des feedbacks implicites, la fonction objectif J à minimiser pour apprendre les paramètres du réseau de neurones est généralement l'entropie croisée (cross-entropyloss).

$$Minimise J = -\sum_{(u,i)\in R'}^{N} log\hat{r}_{u,i} - \sum_{(u,j)\in R'-}^{N} log(1-\hat{r}_{u,j}) \text{[II.16]}$$

$$= - \sum_{(u,i)\in R' \; UR'-}^{N} r_{u,i} log\hat{r}_{u,i} + (1-r_{u,i}) log(1-\hat{r}_{u,i})$$

De manière plus spécifique, le modèle NeuMF proposé dans (He, Liao, Zhang, Nie, & Hu, 2017) combine deux architectures de réseaux de neurones : Generalized Matrix Factorization (GMF) et Multi-Layer Perceptron (MLP), comme illustré dans la figure II.12 L'idée principale de l'architecture de neurones de GMF est de généraliser les modèles basés sur la factorisation matricielle, c'est-à-dire pu ·qi ,où i) les facteurs latents sont traités de la même manière (avec les mêmes poids pour chaque facteur latent) et ii) une fonction linéaire (le produit scalaire) est utilisée pour modéliser l'interaction entre l'utilisateur et l'item. Le modèle GMF peut automatiquement apprendre différents poids pour chaque facteur latent et envisager des liens non linéaires entre les utilisateurs et les items. La non-linéarité peut être obtenue par une fonction d'activation non linéaire (par exemple, sigmoïde) appliquée au produit scalaire.Le réseau de neurones MLP, contrairement à GMF qui utilise un produit fixe de pu ·qi, permet d'apprendre des interactions plus subtiles entre utilisateurs et items. Cela est rendu possible par l'ajout de couches cachées (hiddenlayers) sur le vecteur concaténé de pu et de qi. Enfin, les dernières couches cachées de GMF et MLP sont concaténées pour générer la prédiction finale du rating d'un item par un utilisateur.

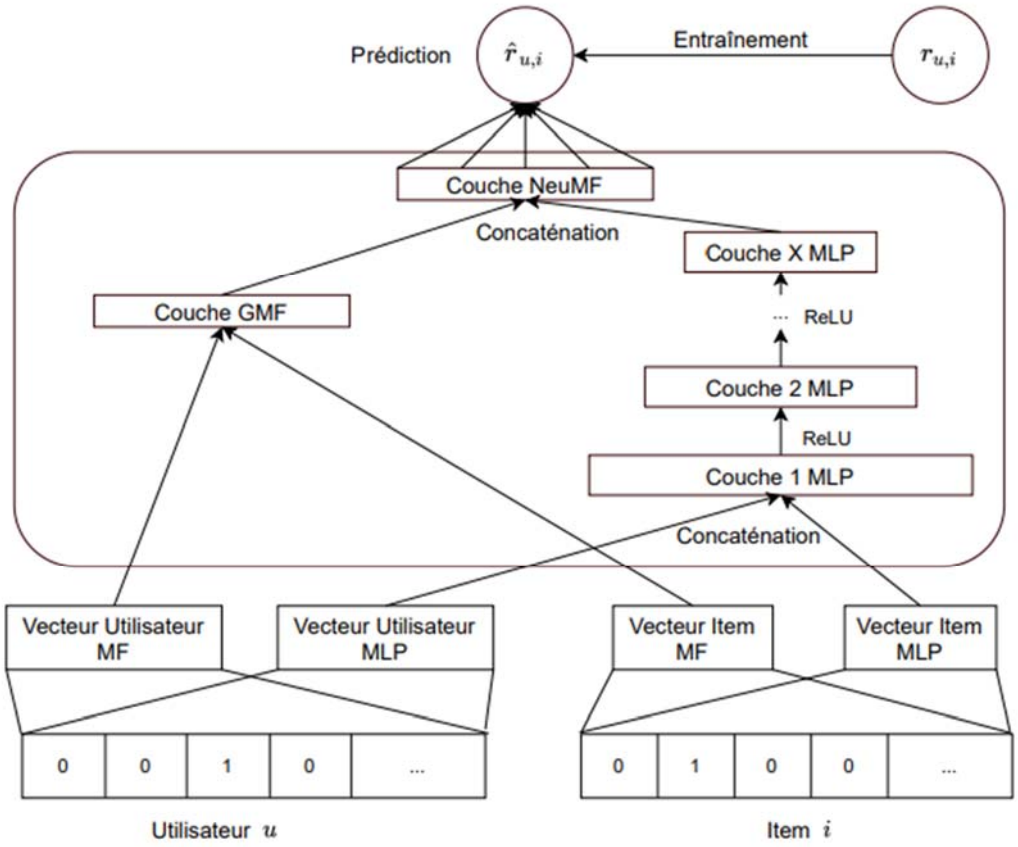


FIGURE II.11 : Architecture complet du modèle NeuMF(He, Liao, Zhang, Nie, & Hu, 2017)

II.8.6 Le filtrage collaboratif basé sur un auto-encodeur (ACF)

Les auto-encodeurs sont une méthode d'apprentissage non supervisée qui utilise les réseaux de neurones pour l'apprentissage de la représentation en cherchant à reconstituer les données d'entrée dans la couche de sortie. Ils sont constitués d'une couche d'entrée, d'une couche dissimulée et d'une couche de sortie. Les informations d'entrée sont envoyées à la couche d'entrée. La couche d'entrée et la couche cachée forment un encodeur. Un décodeur est construit par la couche cachée et la couche de sortie. Les informations de sortie sont extraites de la couche de sortie (SCHMITT, 2018).

Il existe deux façons générales d'appliquer l'auto-encodeur au système de recommandation :

- Apprendre des représentations d'entités de dimension inférieure à la couche de goulot d'étranglement en utilisant l'auto-encodeur.
- Procéder à la remplissage des vides de la matrice d'interaction directement dans la couche de réalisation. Il est possible d'utiliser presque toutes les versions d'autoencodeur pour la tâche de recommandation.

Il y a actuellement plusieurs types de codeurs automatiques utilisés dans les systèmes de recommandations. Ces quatre sont les plus fréquents :

a)Autoencodeur de débruitage (Le Denoising Autoencoder DAE)

Les entrées sont corrompues avant d'être intégrées dans la représentation masquée, puis l'entrée d'origine est reconstruite à partir de sa version corrompue. L'objectif est de contraindre la couche cachée à développer des caractéristiques plus solides et d'empêcher le réseau d'acquérir simplement la fonction d'identité.

b)Auto-encodeurs à réduction de bruit empilés (Stacked Denoising Autoencoder SDAE)

Diffuse plusieurs auto-encodeurs de débruitage les uns sur les autres afin d'obtenir des représentations de niveau supérieur des entrées. En général, la formation est optimisée en utilisant des algorithmes sophistiqués, qui évoluent à chaque étape. Ces désavantages sont le coût de calcul élevé de la formation et le manque d'évolutivité pour les fonctionnalités de grande taille.

c) autoencodeurs de redaction de bruits marginalises (Marginalized Denoising Autoencodeur MDAE)

Il est possible d'éviter le coût de calcul élevé du SDAE en excluant la corruption des fonctionnalités stochastiques. Il possède donc une vitesse d'apprentissage rapide, une mise en place facile et une flexibilité pour s'adapter à des données de grande taille.

d) Les auto-encodeurs variationnels (Variational Autoencoder VAE)

Il s'agit d'un modèle non supervisé de variable latente qui acquiert une représentation approfondie à partir de données de grande taille. Le concept consiste à encoder l'entrée comme une distribution de probabilité plutôt qu'une estimation ponctuelle, comme c'est le cas dans l'autoencodeur vanille. Par la suite, VAE se sert d'un décodeur afin de reconstruire l'entrée initiale en utilisant des échantillons de cette distribution de probabilité.

II.8.7 Co-occurrence CNN for recommendation (CoCNN)

Co-occurrence CNN for recommendation (CoCNN) (CHEN, MA, & ZHOU, 2022), est un nouveau modèle de recommandation, qui combine un modèle de cooccurrence et un modèle CNN pour un filtrage collaboratif avec feedback implicite. L'idée clé du modèle de cooccurrence est que certains éléments apparaissent toujours entre paires sur la liste des favoris d'un utilisateur. Dans le réseau CoCNN, les relations de cooccurrence agissent comme un pont entre les paires utilisateur-élément et les paires élément-élément, qui ne sont pas observées directement.

Un exemple du modèle de cooccurrence de film est illustré dans la figure II.14. Lorsqu'un utilisateur (qui est un fan de Tom Hanks) regarde les célèbres films de Tom Hanks, tels que "Forrest Gump", "The Terminal'' et ''CastAway'', il/elle est très susceptible de leur attribuer une note élevée. Ainsi, pour capturer la relation entre les éléments, les informations de cooccurrence élément-élément sont utilisées pour construire le modèle.

Pour la modélisation de cooccurrence de leurs modèle, les auteurs supposent également que plus deux éléments apparaissent fréquemment ensemble, plus ils sont proches l'un de l'autre. Prenez la figure 1.1 par exemple : dans l'historique des utilisateurs, les films de Tom Hanks, tels que « Forrest Gump », « The Terminal » et « CastAway », apparaissent toujours ensemble. Ainsi, tous les films de cooccurrence sont étroitement liés. Ils supposent que deux éléments hautement corrélés, co-appréciés par les utilisateurs ayant des états similaires, sont proches l'un de l'autre. Ainsi, le modèle de cooccurrence est utilisé pour explorer les relations entre les éléments. Au lieu de calculer la matrice de cooccurrence des éléments, ils reconstruisent les interactions brutes selon la matrice de cooccurrence, dont l'élément est un lorsque l'utilisateur interagit avec deux éléments différents. Dans CoCNN, CNN est appliqué directement sur les facteurs latent (embedding), plutôt que via des étapes intermédiaires par des opérations de produits externes. Dans leur modèle basé sur CNN dans lequel un filtre de taille 2 × 2 capture le lien entre un utilisateur et un élément, et un filtre de taille 3 × 3 capture les liens entre un utilisateur et ses éléments pour améliorer performance des recommandations.

FIGURE II.12 : Un exemple de modèles de cooccurrence dans un film.

II.8.7.1 Matrice des ratings

Dans un système de recommandation, les retours des utilisateurs, qu'ils soient explicites ou implicites, peuvent être représentés par une matrice dont les lignes représentent les utilisateurs et les colonnes représentent les items. Formellement, pour un SdR avec$|U|$, utilisateurs et $|I|$ items, la matrice des ratings, $R_{|U|x|I|}$ est systématiquement construite. $r_{u,i}$, i.e. la valeur de rating assignée par l'utilisateur u pour l'item i , est stockée dans la case située à la u-ième ligne et à la i-ième colonne de la matrice R. La figure 2 illustre un exemple de deux matrices de ratings qui représentent les retours de 4 utilisateurs sur 4 items.

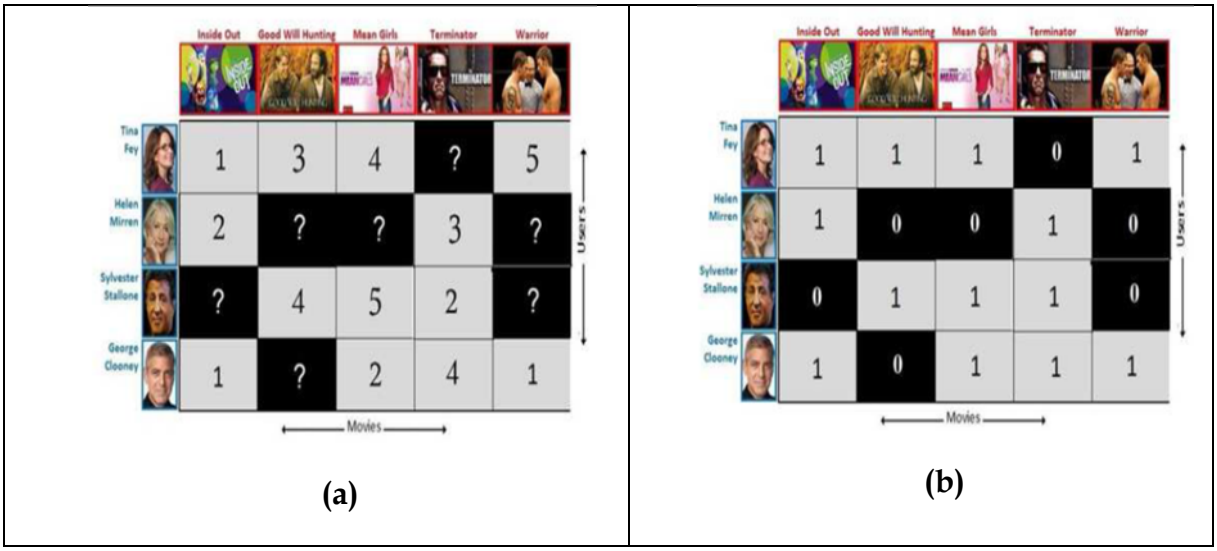


FIGURE II.13 : Exemples de matrices de ratings pour des feedbacks explicites (a) et implicites (b)

II.8.7.2 Définition du problème

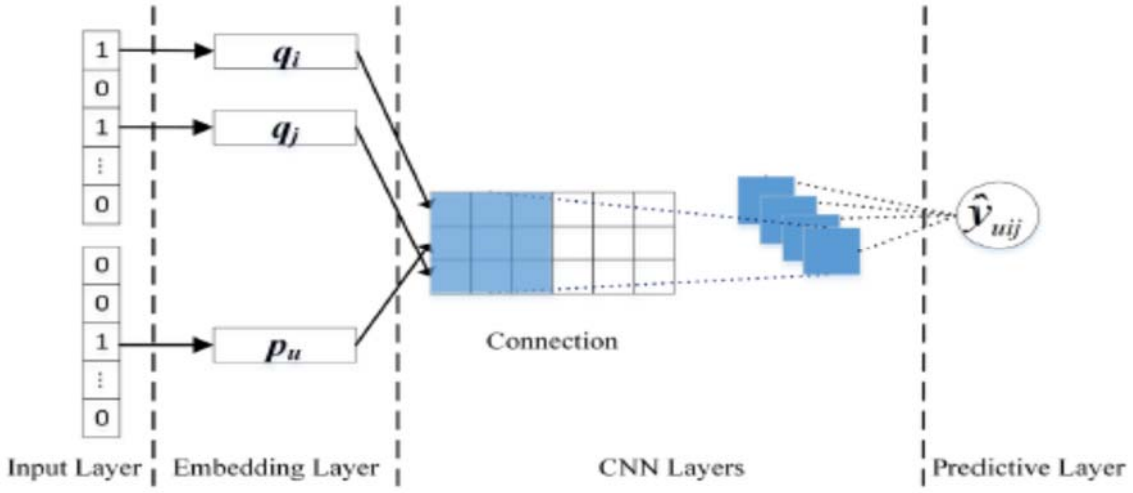
Dans les tâches de recommandation, étant donné deux ensembles : U avec m utilisateurs, I avec n éléments, leurs interactions sont notées $R \in R^{m \times n}$. Dans R, l'élément, $r_{uj} = 1$ désigne une interaction observée entre u et i ; sinon $r_{uj} = 0$.

Ensuite, une matrice de préférence par paire de l'utilisateur, $Y \in R^{m \times n \times n}$, est générée à partir de R. Dans, l'élément $yuij = 1$ indique que pour l'utilisateur, u, les éléments i et j se produisent simultanément , $yuij = 0$ indique que les éléments i et j ne sont jamais apparus ensemble.

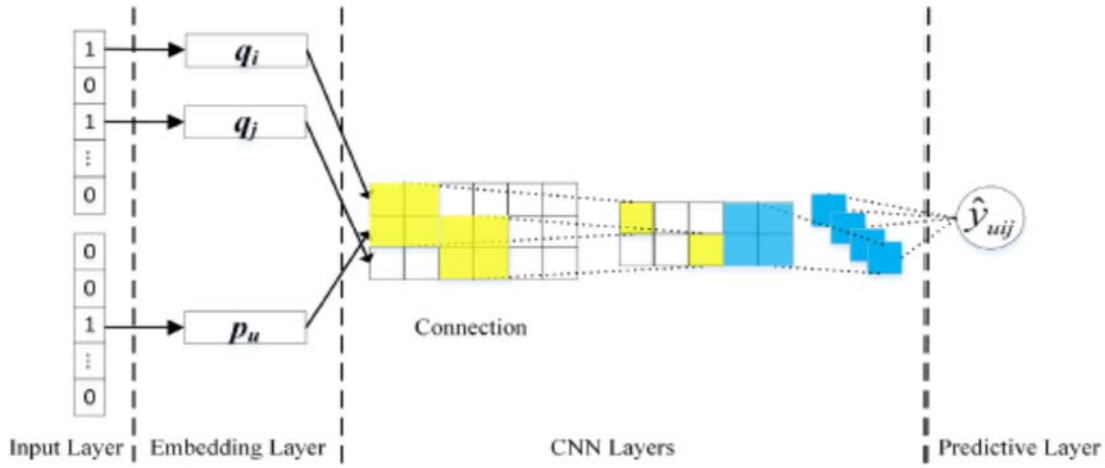
$$y_{uij} = \begin{cases} 1, & if\, r_{ui} = 1 \; and \; r_{uj} = 1 \\ o, & otherwise \end{cases} \qquad [II.17]$$

II.8.7.3 Le modèle CoCNN

La méthode proposée vise à apprendre le classement des recommandations personnalisées à partir des informations d'interaction utilisateur-élément. Il existe deux conceptions clés dans leurs modèles : un filtre de taille 2 × 2 capture le lien entre un utilisateur et l'élément, et un filtre de taille 3 × 3 capture les liens entre un utilisateur et ses éléments. Pour capturer efficacement la relation entre la paire utilisateur-élément avec CNN, une nouvelle architecture est proposée. Dans CoCNN, le vecteur latent de l'utilisateur (User embeddings) u et sa préférence par paire pour les éléments i et j sont combinées dans une matrice. Ensuite, la matrice est transmise à CNN pour apprendre le comportement par paire. Pour modéliser les relations utilisateur-élément et élément-élément, ils ont conçu deux méthodes, et leurs architectures sont présentés à la FigureII.16.



a) CoCNN avec un filtre de taille 3 × 3



b) CoCNN avec un filtre de taille 2 × 2

FIGURE II.14 : Architecture de CoCNN

Couches entrée et embedding (Input layers and embedding layer).

La première couche est la couche d'entrée, où deux vecteurs de caractéristiques décrivant l'utilisateur u et l'item i sont utilisés en entrée. Puisque l'accent principal de cette étude était mis sur un cadre de filtrage collaboratif pur, les identifiants de l'utilisateur u et de l'élément i sont fournis. Comme la plupart des modèles de réseau de neurones pour la recommandation, des représentations one-hot sont utilisées pour représenter chaque utilisateur et chaque item, dont les caractéristiques éparses sont mappées de l'entrée à une représentation dense par la couche d'embedding. Les embeddings, $pu \in R_1 \times k$ et $qi \in R_1 \times k$, générés à partir de tables de recherche, sont définis comme suit :

$$pu = flookup(u)$$

$qi = flookup(i)$

Couches CNN (CNN Layers). Cette couche contient des couches de convolution et de pooling. Le CNN est puissant pour apprendre des caractéristiques profondes à partir de la matrice. La matrice de connexion est alimentée à la couche CNN pour apprendre les corrélations non linéaires entre les différents embeddings.

Connexion. Pour apprendre les relations utilisateur–item et item–item, un nouveau réseau de neurones est conçu. Étant donné un triple (pu, qi, qj), une matrice de connexion les intégrant est concaténée comme suit : c = [qi, pu, qj]∈ R3×k. Ensuite, pour connecter les embeddings avant de les alimenter à la couche CNN, deux manières différentes sont conçus.

Filtre 3 × 3. Dans CoCNN avec un filtre 3 × 3 (CoCNN3), un champ réceptif complet fournit au modèle les relations globales entre les utilisateurs et leurs éléments de co-occurrence. Un champ réceptif plus grand tend à avoir une capacité plus puissante pour capturer plus d'informations à partir de l'entrée.

$$h = ReLU\ (w_3 * c + w_3)[\text{ II.17}]$$

où : $*$ désigne l'opération de convolution ; et $\boldsymbol{w_3}$désigne un filtre de taille 3 × 3 dans l'opération de convolution, et le stride est de 1.

Filtre 2 × 2. Dans CoCNN avec un filtre 2 × 2 (CoCNN2), les différences par rapport au CoCNN3 sont le mode de connexion, la taille du filtre pour la matrice de connexion et la profondeur des couches CNN. Par rapport à CoCNN3, CoCNN2 utilise d'abord un filtre de taille plus petite, 2 × 2, pour capturer les corrélations utilisateur–élément :

$$h' = ReLU\ (w_2' * c + b_2')[\text{ II.18}]$$

où : $\boldsymbol{w_2'}$ désigne un filtre de taille 2 × 2 dans l'opération de convolution, et le stride est de 1.

Ensuite, dans la couche plus profonde, un filtre de taille 2 × 2 continue d'être utilisé pour apprendre la corrélation entre les utilisateurs et leurs éléments de co-occurrence. CoCNN2 apprend des caractéristiques plus profondes grâce à sa structure CNN profonde comme suit :

$$h = ReLU\ (w_2 * h' + b_2)[\text{ II.19}]$$

où : w_2 désigne un filtre de taille 2 × 2, et le stride est de 1.

- **Couche prédictive (Predictive Layer).** Enfin, pour prédire la probabilité, $\hat{y}uij$, une couche entièrement connectée est utilisée pour mapper le résultat de la couche CNN au résultat final, formulée comme suit :

$$\hat{y}_{uij} = \sigma\ (w_P^T h + b_p)[\text{ II.20}]$$

OùWp etb_p désignent respectivement le poids et le biais de la couche prédictive ; $\sigma(\cdot)$désigne la fonction $sigmoid$.

Dans cette tâche, nous visons à apprendre les intérêts des utilisateurs à partir de leur préférence relative pour différentes paires d'éléments. La fonction d'erreur quadratique moyenne (MSE) est utilisée pour évaluer la perte, définie comme suit :

$$l_{co} = -\sum_{u=1}^{n} \sum_{(i,j)\in C \cup C^-} (y_{uij} - \hat{y}_{uij})^2 \text{[II.21]}$$

Où C_u et C^-_u désignent respectivement l'ensemble des instances positives et négatives de u.

II.9 Conclusion

Les systèmes de recommandation sont une nouvelle technique puissante et deviennent rapidement un outil crucial notamment pour le commerce électronique sur le Web.

Dans ce chapitre, nous avons présenté les systèmes de recommandation qui sont devenus omniprésents ces dernières années dans de nombreux domaines. Ces systèmes sont conçus pour aider les utilisateurs à trouver des ressources qui les intéressent et qui sont adaptées à leurs préférences, parmi le nombre important des choix qui s'offrent à eux.

Nous avons d'abord défini la notion des systèmes de recommandation. Ensuite, Nous avons présenté ses composants principaux qui sont les utilisateurs et les items. Basé sur les préférences d'utilisateur et l'exploitation du profil d'utilisateur, un système de recommandation peut proposer des items aux utilisateurs.

Plus précisément, dans ce chapitre nous avons passé en revue les techniques (approches) de recommandations existantes : le filtrage collaboratif, le filtrage à base de contenu, le filtrage contextuel, le filtrage hybride. Nous avons aussi discuté d'un ensemble de travaux de l'état de l'art qui exploitent les techniques de machine et deeplearning dans les systèmes de recommandation.

Dans le prochain chapitre nous décrivons l'implémentation, l'évaluation et la comparaison de quelques algorithmes de recommandation. Un de ces algorithmes sont basés sur le machine learning qui sont les modèles k-voisins les plus proches (KNN); et trois modèles basé sur le deeplearning qui est le modèle de filtrage collaboratif neuronal NCF, CoCNN et CNN.

Chapitre III

Implémentation et évaluation

III.1 Introduction

L'objectif de ce chapitre est de présenter l'implémentation, l'évaluation et la comparaison des algorithmes des modèles : CoCNN, CNN qui est une version simplifiée de CoCNN sans motif de cooccurrence, NCF et KNN. Les expérimentations ont été réalisées sur une collection de test «MovieLens», plusieurs mesures d'évaluation ont été utilisées pour évaluer les différents algorithmes.

III.2 Collection de test (Dataset)

MovieLens, c'est l'un des plus populaires ensembles de données qui a été créé en 1997 par GroupLensResearch, un laboratoire de recherche sur les interactions homme-machine du Département d'Informatique et d'Ingénierie de l'Université du Minnesota, afin de recueillir des données de recherche sur des recommandations personnalisées. Il fonde ses recommandations sur les informations fournies par les utilisateurs du site Web. Les classements dans MovieLens peuvent survenir à tout moment, en fait, cela peut arriver des années plus tard après avoir regardé un film. Les utilisateurs saisissaient souvent de nombreuses évaluations à la fois dans l'espoir d'obtenir des recommandations plus personnalisées ou simplement pour leur satisfaction.

Trois ensembles de données ont été utilisées : MovieLens100K, MovieLens1M et Lastfm. Les deux premiers, collectés par GroupLens (Harper & Konstan, 2015), sont disponibles sur son site web[1]. Le dernier, collecté à partir de Last.fm, est disponible sur le site web de GroupLens[2]. Certaines statistiques sont présentées dans le Tableau III.1.

	# of Users	# of Items	# of Interactions	Density
MovieLens100K	943	1682	100,000	6.3%
MovieLens1M	6040	3706	1,000,209	4.47%
Lastfm	518	3488	46,175	0.26%

Tableau III. 1 Statistiques sur les datasets

Notre jeu de données contient deux fichiers textuels ; movies.csv et ratings.csv. Le premier fichier "movies.csv" contient la table des films et le deuxième fichier "ratings.csv" contient la table des votes des utilisateurs pour les films.

[1] *https://grouplens.org/datasets/movielens/*
[2] *https://grouplens.org/datasets/hetrec-2011/*

	movieId	title	genres	year
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1995
1	2	Jumanji (1995)	Adventure|Children|Fantasy	1995
2	3	Grumpier Old Men (1995)	Comedy|Romance	1995
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	1995
4	5	Father of the Bride Part II (1995)	Comedy	1995
...	...	...	...	...
9737	193581	Black Butler: Book of the Atlantic (2017)	Action|Animation|Comedy|Fantasy	2017
9738	193583	No Game No Life: Zero (2017)	Animation|Comedy|Fantasy	2017
9739	193585	Flint (2017)	Drama	2017
9740	193587	Bungo Stray Dogs: Dead Apple (2018)	Action|Animation	2018
9741	193609	Andrew Dice Clay: Dice Rules (1991)	Comedy	1991

9742 rows × 4 columns

FIGURE III.1 : Table des films provenant du fichier "movies.csv".

	userId	movieId	rating	timestamp
0	1	1	4.0	964982703
1	1	3	4.0	964981247
2	1	6	4.0	964982224
3	1	47	5.0	964983815
4	1	50	5.0	964982931
...	...	...	...	...
100831	610	166534	4.0	1493848402
100832	610	168248	5.0	1493850091
100833	610	168250	5.0	1494273047
100834	610	168252	5.0	1493846352
100835	610	170875	3.0	1493846415

100836 rows × 4 columns

FIGURE III.2 : Table des votes provenant du fichier "ratings.csv".

III.3 Mesures d'évaluation utilisées

Les mesures d'évaluation mesurent la capacité de l'algorithme de recommandation à prédire correctement les évaluations connues des utilisateurs. Elles utilisent des évaluations correctes d'un ensemble de donnée de test et les comparent avec des prédictions proposées par l'algorithme de recommandation en se basant sur un ensemble de donnée

d'apprentissage. La majorité de ces métriques proviennent du domaine de la recherche d'information. Nous présentons ci-dessous les métriques que nous avons utilisées.

III.3.1 Le gain cumulé actualisé normalisé NDCG

Pour comprendre NDCG, nous devons comprendre ses prédécesseurs : Gain Cumulatif (CG) et Gain Cumulatif actualisé (DCG).

a) Le gain cumulé CG

Chaque recommandation est associée à un score de pertinence. Le gain cumulé est la somme de tous les scores de pertinence d'un ensemble de recommandations.

$$CG = \sum_{i=1}^{n} rel_p \qquad [III.1]$$

rel_p : Représente la liste des items pertinents dans la collection jusqu'à la position p.

b) Le gain cumulé actualisé DCG

Le gain cumulé actualisé (DCG) est souvent utilisé pour mesurer l'efficacité des algorithmes. À l'aide d'une échelle de pertinence graduée des items dans un ensemble de résultats, DCG mesure l'utilité ou le gain d'un item en fonction de sa position dans la liste de résultats. Le gain est accumulé du haut de la liste de résultats vers le bas.

$$DCG = \sum_{i=1}^{|rel_p|} \frac{2^{rel}i - 1}{log_2(i+1)} \qquad [III.2]$$

IDCG est le gain cumulé actualisé (DCG) idéal avec ordre.

c) NDCG

La comparaison des performances d'un algorithme à l'autre ne peut pas être obtenue de manière cohérente en utilisant uniquement DCG, de sorte que le gain cumulé à chaque position pour une valeur choisie de P doit être normalisé. Cela se fait en triant tous les items pertinents du corpus par leur pertinence relative, en produisant le maximum de DCG possible par position P, également appelé Ideal DCG (IDCG) à travers cette position. Pour calculer NDCG, nous devons d'abord calculer :

- DCG de l'ordre recommandé
- DCG de l'ordre idéal (IDCG).

NDCG est alors le rapport entre DCG d'ordre recommandé et IDCG d'ordre idéal.

$$NDCG = \frac{DCG_p}{IDCG_p} \qquad [III.3]$$

Le principal avantage du NDCG est qu'il prend en compte les valeurs de pertinence notées. Lorsqu'elles sont disponibles dans l'ensemble de données, le NDCG est un bon ajustement.

III.3.2. Hit Ratio (HR)

Le "HR of Top-K items (HR)" désigne le pourcentage d'éléments recommandés aux utilisateurs qui sont réussis. Si hit_u indique le nombre d'éléments corrects dans la liste, HR est défini comme suit :

$$HR = \frac{\sum_{u=1}^{m} hit_u}{m.k} \qquad [III.4]$$

Dans les systèmes de recommandation, le Taux de Réussite (HR) et le Gain Cumulatif Pondéré Normalisé (NDCG) sont populaires pour évaluer les capacités des modèles de recommandation. Le premier se concentre sur la précision des prédictions , le second se concentre sur les éléments en tête de la liste de recommandation.

III.3.3 MAE (Mean Average Error)

Le "Mean Absolute Error" (MAE) est une mesure de la différence entre deux variables continues. Le MAE est la moyenne des différences absolues entre la sortie obtenue et la sortie cible. Il est calculé comme suit :

$$MAE = \frac{1}{N}\sum_{i=1}^{N}(p_{ui} - r_{ui}) \qquad [III.5]$$

Où$\mathbf{p_{ui}}$est la note prédite pour l'utilisateur u et l'élément i, r_{ui} la note réelle et N le nombre de prédictions.

III.3.4 RMSE (Root Mean Squared Error)

La racine carrée de l'erreur quadratique moyenne (RMSE) est une mesure statistique qui représente l'écart-type entre un ensemble de valeurs estimées et les valeurs réelles. Dans les systèmes de recommandation, elle est utilisée pour mesurer dans quelle mesure un ensemble de prédictions s'éloigne des valeurs réelles. Le RMSE est calculé comme suit :

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_{ui} - r_{ui})^2} \qquad [III.6]$$

Où$\mathbf{p_{ui}}$ est la note prédite pour l'utilisateur u et l'élément i, r_{ui} la note réelle et N le nombre de prédictions.

III.4 Les méthodes comparées

Nous avons comparé trois méthodes de recommandation basée sur l'apprentissage profond (deep learning) : CoCNN, CNN et NCF avec une méthode de recommandation basée sur les techniques de machine learning : KNN. Puisque les méthodes des réseaux de neurones évaluent les interactions utilisateur-item, nous avons mis en concurrence la méthode avec des méthodes de filtrage collaboratif utilisateur-item plutôt qu'avec des modèles item-item.

Le filtrage collaboratif, aussi connu sous le nom de collaborative filtering (CF) en anglais, est une technique de recommandation utilisée pour prédire les préférences des utilisateurs en se basant sur les évaluations ou les comportements d'autres utilisateurs similaires. Cette méthode repose sur l'idée que si deux personnes ont eu des goûts similaires dans le passé, elles continueront probablement à avoir des préférences similaires à l'avenir.

Il existe deux principales approches de filtrage collaboratif :

- **Filtrage collaboratif basé sur l'utilisateur (user-based collaborative filtering**): Cette approche recommande des éléments à un utilisateur en se basant sur les préférences d'utilisateurs similaires. Par exemple, si un utilisateur A a des préférences similaires à un utilisateur B sur un certain nombre d'articles ou de produits, alors les articles appréciés par l'utilisateur B mais non encore consommés par l'utilisateur A pourraient lui être recommandés.

- **Filtrage collaboratif basé sur les éléments (item-based collaborative filtering)**: Dans cette approche, les recommandations sont basées sur la similarité entre les éléments eux-mêmes plutôt que sur la similarité entre les utilisateurs. Par exemple, si un utilisateur aime un certain film, le système recommandera d'autres films qui sont similaires à celui-ci en fonction des évaluations données par d'autres utilisateurs.

III.4.1 Filtrage collaboratif utilisant la méthode des k plus proches voisins (KNN)

Il s'agit de l'une des techniques de filtrage collaboratif de voisinage populaires basées sur l'utilisateur (SARWAR, KARYPIS, & KONSTAN, 2001). Nous avons adapté cette méthode pour apprendre à partir des données d'interactions utilisateur-item implicites. En général, les algorithmes de filtrage collaboratif basés sur les voisins se servent de l'ensemble de la matrice des notes des utilisateurs afin de formuler des recommandations. Afin d'utiliser l'algorithme de K plus proche des voisins, il est nécessaire que le système possède une mesure de similarité afin de distinguer les utilisateurs proches de ceux éloignés. On peut tirer cette mesure de similarité de la distance euclidienne, de la mesure du cosinus, de la corrélation de Pearson, et ainsi de suite. Par la suite, lorsqu'il est demandé une nouvelle information, le système va chercher les k utilisateurs les plus proches de l'utilisateur cible. Enfin, une consultation majoritaire est organisée afin de déterminer quel élément sera recommandé (filtrage basé sur les utilisateurs). La valeur de k est fixée par l'utilisateur.

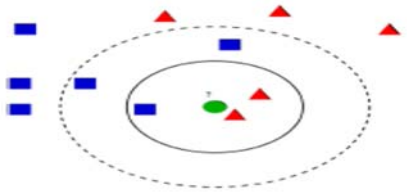


FIGURE III.3 : Exemple de classification k-NN.

III.4.2 Le filtrage collaboratif neuronal (NCF)

NCF est un nouveau modèle de factorisation de matrice neuronale, qui regroupe la factorisation matricielle généralisée (GMF) et le perceptron multicouche (MLP) pour unifier les forces de linéarité de MF et de non-linéarité de MLP pour modéliser les structures latentes utilisateur-item. De manière plus spécifique, le modèle de NeuMF proposé dans (He, Liao, Zhang, Nie, & Hu, 2017) combine deux architectures de réseaux de neurones : Generalised Matrix Factorization (GMF)et Multi-Layer Perceptron (MLP), comme illustré dans la figure III.4. L'idée principale de l'architecture de neurones de GMF est de généraliser les modèles basés sur la factorisation matricielle, i.e., pu.qi, dans laquelle i) les facteurs latents sont traités de la même manière (avec les même poids pour chaque facteur latent) et ii) une fonction linéaire (le produit scalaire) est utilisée pour modéliser l'interaction entre l'utilisateur et l'item. Le modèle de GMF peut automatiquement apprendre différents poids pour chaque facteur latent et envisager des liens non-linéaires entre les utilisateurs et les items. La non-linéarité peut être obtenue par une fonction d'activation non-linaire (e.g., sigmoïde) sur le produit scalaire. Le réseau de neurone de MLP, contrairement à celui de GMF qui n'utilise qu'un produit fixé de pu.qi, permet d'apprendre des interactions plus subtiles entre utilisateurs et items. Cela est rendu possible par l'ajout de couches cachées (hidden layers) sur le vecteur concaténé de pu et de qi. Enfin, les dernières couches cachées de GMF et MLP sont concaténées pour générer la prédiction finale du rating d'un item par un utilisateur.

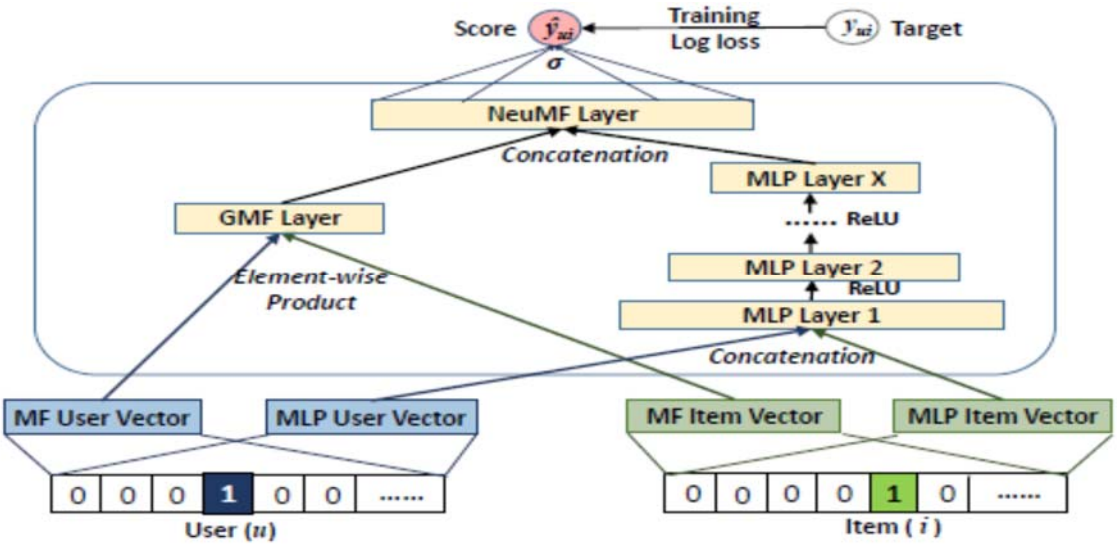


FIGURE III.4 : Architecture complet du modèle NeuMF

III.4.3 Co-occurrence CNN for recommendation (CoCNN)

Co-occurrence CNN for recommandation (CoCNN), est un nouveau modèle de recommandation, qui combine un modèle de cooccurrence et un modèle CNN pour un filtrage collaboratif avec feedback implicite. L'idée clé du modèle de cooccurrence est que certains éléments apparaissent toujours entre paires sur la liste des favoris d'un utilisateur. Dans le réseau CoCNN, les relations de cooccurrence agissent comme un pont entre les paires utilisateur-item et les paires item-item, qui ne sont pas observées directement.

Un exemple du modèle de cooccurrence de film est illustré dans la figure III.5. Lorsqu'un utilisateur (qui est un fan de Tom Hanks) regarde les célèbres films de Tom Hanks, tels que "Forrest Gump", "The Terminal" et ''CastAway'', il est très susceptible de leur attribuer une note élevée. Ainsi, pour capturer la relation entre les items, les informations de cooccurrence item-item sont utilisées pour construire le modèle. Ainsi, le modèle de cooccurrence est utilisé pour explorer les relations entre les items.

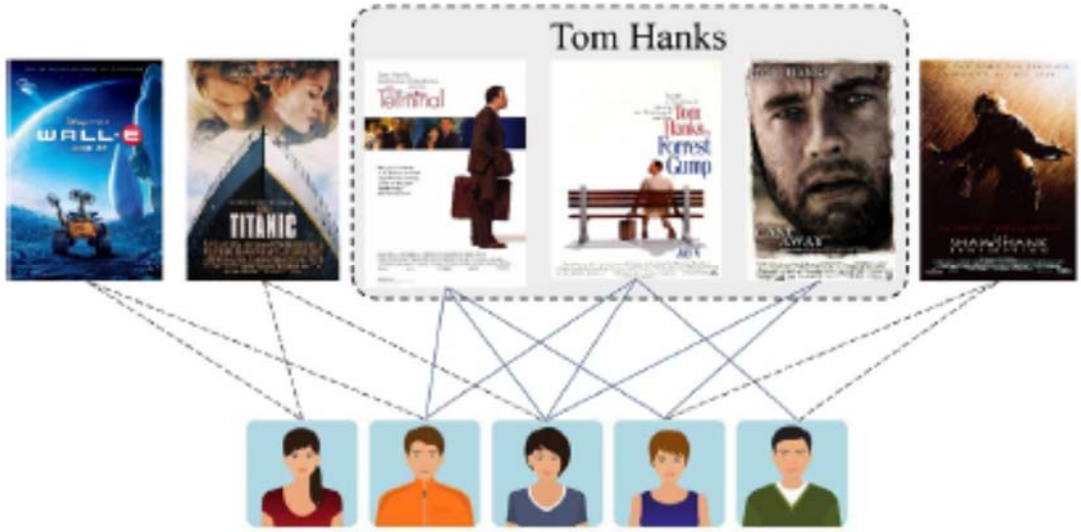
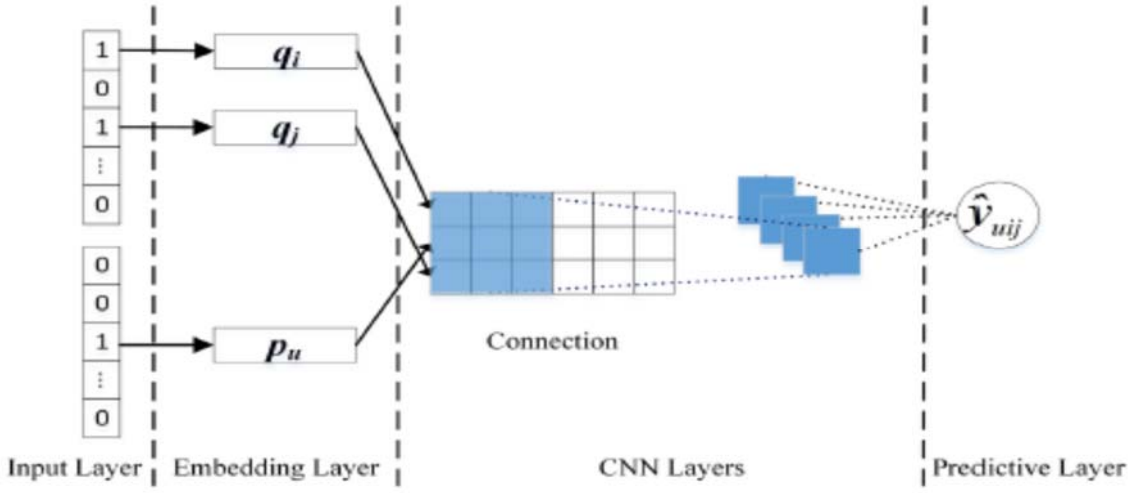
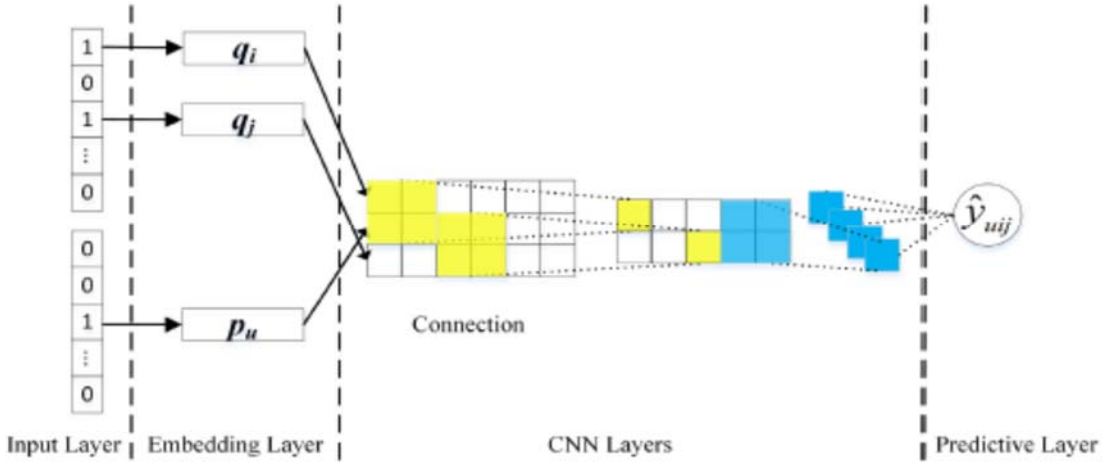


FIGURE III.5 : Un exemple de modèles de cooccurrence dans un film.

La méthode proposée vise à apprendre le classement des recommandations personnalisées à partir des informations d'interaction utilisateur-item. Il existe deux conceptions clés dans le modèle : un filtre de taille 2 × 2 capture le lien entre un utilisateur et un item, et un filtre de taille 3 × 3 capture les liens entre un utilisateur et ses items. Dans CoCNN, le facteur latent de l'utilisateur (User embeddings) u et sa préférence par paire pour les items i et j sont combinées dans une matrice. Ensuite, la matrice est transmise à un réseau CNN pour apprendre le comportement par paire. Pour modéliser les relations utilisateur-item et item-item, nous considérons deux méthodes, et leurs architectures sont présentées à la Figure III.6.



a) CoCNN avec un filtre de taille 3 × 3



b) CoCNN avec un filtre de taille 2 × 2

FIGURE III.6 : Architecture de CoCNN

III.4.4 CNN for recommandation

Le schéma du modèle est présenté dans la figure III.7. CNN qui est une version simplifiée de CoCNN sans motif de cooccurrence. Dans ce modèle, les deux premières couches (entrée et embedding) sont les mêmes que dans le modèle CoCNN. La différence clé réside dans la partie connexion. Dans la connexion du CNN, les deux facteurs latents (embeddings) d'un utilisateur et de son item sont combinés. Les couches restantes (CNN et prédictive) sont les mêmes que dans le modèle CoCNN.

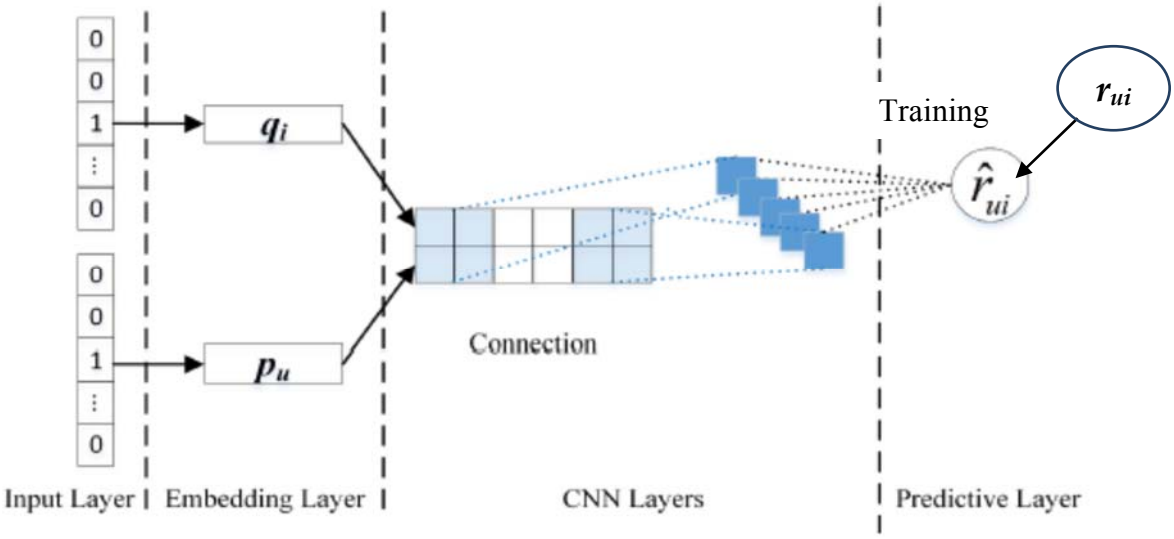


FIGURE III.7 : Architecture de CNN for recommandation

III.5 Environnement de travail

Dans ce qui suit nous allons présenter l'environnement du développement et les outils que nous avons utilisé pour réaliser notre travail.

III.5.1 Environnement matériel

Pour l'environnement matériel, nous avons utilisé deux machines qui ont les caractéristiques suivantes :

Un ordinateur lenovo

Processeur	Intel(R) Core(TM) i5-7440HQ CPU @ 2.80GHz 2.81 GHz
Mémoire RAM	8,00 Go
Type du système	Systèmed'exploitation 64 bits, processeur×64
Systèmed'exploitation	Windows 11Pro

Tableau III. 2 Caractéristiques du matériel utilisé.

III.5.2 Environnement de développement

Pour l'environnement de développement nous avons utilisé :

Anaconda : Anaconda est une distribution libre et open source des langages de programmation Python et R appliqué au développement d'applications dédiées

à la science des données et à l'apprentissage automatique (traitement de données à grande échelle, analyse prédictive, calcul scientifique), qui vise à simplifier la gestion des paquets et de déploiement. Les versions se paquetages sont gérées par le système de gestion de paquets conda. La distribution Anaconda est utilisée par plus de 6 millions d'utilisateur et comprendre plus de 250 paquets populaires en science des données adaptés pour Windows, Linux et MacOs.

Jupyter : Jupyter est une application web utilisée pour programmer, initialement développés pour les langages de programmation Julia, Python et R (d'où le nom Jupyter, et supporte près de 40 langages. Jupyter est une évolution du projet IPython. Jupyter permet de réaliser des calepins ou notebooks qui sont utilisés en science des données pour explorer et analyser des données. La cellule est l'élément de base d'un notebook jupyter. Elle peut contenir du texte formaté au format markdown ou du code informatique qui pourra être exécuté.

Python : Python est un langage de programmation assez généraliste, c'est-à-dire qu'il est à peu près possible de tout faire avec : des sites et applications web, des applications mobiles, des scripts personnels, des applications de bureau, de l'analyse de données et même des jeux vidéo. Grâce à ses bibliothèques et packages de data science, il est devenu le langage le plus populaire pour les algorithmes de machine learning, data science et le big data [58]. Nous avons utilisé python 3.11 sur Windows 11.

III.6 Configuration des modèles comparés

Un bon classifieur en machine Learning et en deep learning, est un classifieur qui généralise, autrement dit, il a la capacité de faire des prédictions non seulement sur les données utilisées pour le construire, mais surtout sur de nouvelles données. Pour obtenir ces deux types de données, la procédure consiste à effectuer un fractionnement en deux sous-ensembles sur le jeu de données :

- Ensemble de données d'apprentissage : communément appelé train set, utilisé pour entraîner le classifieur, l'ensemble des observations (enregistrements) est appelé x_train et le vecteur de classes (target) correspondantes aux observations (enregistrements) est décrit par y_train.

- Ensemble de données test : Communément appelé test set, l'ensemble des observations (enregistrements) est appelé x_test et le vecteur de classes correspondantes aux

observations est appelé y_test. Ce sont les données qui n'ont pas servi à l'entraînement, utilisées comme une entrée du modèle pour tester sa performance en comparant les prédictions données par le classifieur (y_pred) et les valeurs attendues (y_test).

- Le module "division de jeu de données" s'occupe de ce fractionnement en ensemble de test et ensemble d'apprentissage, dans notre implémentation 80% du jeu de données est consacré pour l'apprentissage quant aux 20% restantes elles forment l'ensemble de test.

Nous avons implémenté quatre modèles de l'état de l'art, à savoir KNN, NCF, CoCNN et CNN. Nous spécifions ci-dessous les hyperparamètres de chacun d'eux.

III.6.1 L'algorithme KNN

Nous avons implémenté l'algorithme KNN à l'aide du modèle KNeighborsClassifier présent dans la bibliothèque sklearn, accompagné d'un ensemble de paramètres à spécifier afin d'obtenir le modèle le plus optimal possible. Ces paramètres sont les suivants :

Hyperparametres	Définition	Valeur	Valeur optimale
n_neighbors	Le nombre de voisins (k) pris par le modèle	[5-100] par pas de 5	10
Métriques	mésures de similarité utilisée	Cosinus, Euclidien, Manhattan, Jaccard	

Tableau III. 3 : Les hyperparamètres de l'algorithme KNN.

III.6.2 L'algorithme NCF

Cet algorithme a été implémenté à l'aide du Framework TensorFlow. NCF a beaucoup de paramètres, les plus importants sont les suivants :

Hyperparametres	Définition	Valeurs	Valeuroptimale
n_factors	Dimension de l'espace latent	[1-400] avec un pas de 50	100
layer_sizes	Taille de la couche d'entrée et des couches cachées de MLP	[[16 , 8, 4]-[64, 32, 16]]	[16, 8, 4]
n_epochs	Nombre d'entrainement du réseau de neurones	[1-300] avec un pas de 50	
model_type	Nous pouvons former un modèle « MLP », « GMF » ou combiné « NCF »	Model_type = « neuMF » Sigmoide et ReLU	

Tableau III. 4 : Les hyperparamètres de l'algorithme NCF.

III.6.3 L'algorithmeCoCNN

Le modèle a plusieurs paramètresclés, les plus importants sont: embedding size (d),learning rate (τ),one regularization parameter for learning parameters(λ), the ratio of each task (α), kernel_size, epochs et batch size.

Hyperparametres	Définition	Valeurs
embedding size (d)	Taille des facteurslatents	32/64
learning rate (τ)	le tauxd'apprentissage	0.01
one regularization parameter for learning parameters (λ)	un paramètre de régularisation pour les paramètres d'apprentissage	10^{-6}
the ratio of each task (α),	le ratio de chaque tâche	1
kernel_size	Taille du noyau	2/3
epochs	Nombred'époques	100
Batch size	taille du lot	1024
model_type	Types de modèles	CoCNN2 (kernel_size=2), CoCNN3 (kernel_size=3) et CNN

Tableau III. 5 : Les hyperparamètres de l'algorithme CoCNN.

Le tableau ci-dessous présente l'architecture du modèle.

```
Net(
  (embedding_user): Embedding(943, 32)
  (embedding_item): Embedding(1682, 32)
  (net_ui): UInet(
    (embedding_user): Embedding(943, 32)
    (embedding_item): Embedding(1682, 32)
    (cnn): Conv2d(1, 32, kernel_size=(2, 2), stride=(1, 1))
    (relu): ReLU()
    (linear): Linear(in_features=992, out_features=1, bias=True)
  )
  (net_uii): UIInet(
    (embedding_user): Embedding(943, 32)
    (embedding_item): Embedding(1682, 32)
    (cnn1): Conv2d(1, 32, kernel_size=(2, 2), stride=(1, 1))
    (cnn2): Conv2d(32, 32, kernel_size=(2, 2), stride=(1, 1))
    (relu): ReLU()
    (linear): Linear(in_features=960, out_features=1, bias=True)
  )
)
```

```
==========================================
Layer (type:depth-idx)                   Param #
==========================================
Net                                      --
├─Embedding: 1-1                         30,176
├─Embedding: 1-2                         53,824
├─UInet: 1-3                             84,000
│    └─Embedding: 2-1                    (recursive)
│    └─Embedding: 2-2                    (recursive)
│    └─Conv2d: 2-3                       160
│    └─ReLU: 2-4                         --
│    └─Linear: 2-5                       993
├─UIInet: 1-4                            84,000
│    └─Embedding: 2-6                    (recursive)
│    └─Embedding: 2-7                    (recursive)
│    └─Conv2d: 2-8                       160
│    └─Conv2d: 2-9                       4,128
│    └─ReLU: 2-10                        --
│    └─Linear: 2-11                      961
==========================================
Total params: 258,402
Trainable params: 258,402
Non-trainable params: 0
==========================================
```

Tableau III. 6 Architecture du modèle CoCNN

III.7 Evaluation des modèles

Dans ce qui suit nous allons tester les modèles avec leurs hyperparamètres à fin de déterminer les valeurs optimales qui obtiennent les meilleurs résultats en terme de NDCG et HR

III.7.1 Evaluation du modèle CoCNN

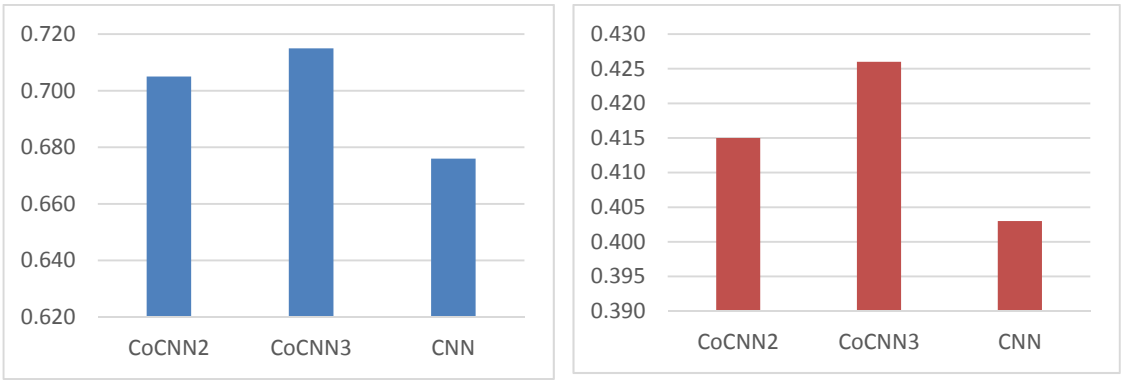
III.7.1.1 Impact du modèle de cooccurrence

Pour évaluer pleinement les performances du motif de cooccurrence dans le modèle CoCNN, nous avons réalisé quelques expériences pour le comparer avec la version simplifiée sans motif de cooccurrence (CNN). Les résultats présentés dans la figure III.8. Dans ces expériences, nous avons fixé la taille d'embedding, d, et le taux d'apprentissage, τ, à 32 et 0,01, respectivement. La figure III.8 nous permet d'observer les points suivants :

Premièrement, par rapport aux résultats de CoCNN2 (filtre de taille 2×2) et CoCNN3 (filtre de taille 3×3), les résultats du modèle simplifiée (CNN) sont moins bons, car des informations et des caractéristiques plus détaillées peuvent être extraites par un motif de cooccurrence.

Deuxièmement, sur le même ensemble de données, CoCNN3 a obtenu des résultats comparables à ceux de CoCNN2. La raison la plus probable est que les deux modèles de cooccurrence appliqués dans le modèle peuvent apprendre automatiquement les relations entre les items, contribuant ainsi au développement de la qualité de la recommandation.

Troisièmement, le modèle, CNN, qui combine directement les embedding, obtient également des résultats comparables. Par conséquent, l'alimentation directe de CNN en embeddings est également efficace.



(a) HR (b) NDCG

FIGURE III.8 : Résultats des différents modèles comparés

III.7.1.2 Impact de la taille des embeddings

Pour examiner plus en détail l'impact de la taille desembedding (qui représente la taille de l'information transportée par l'embedding) sur la performance du modèle CoCNN, nous avons choisi d parmi {8, 16, 32, 64, 128, 256} un par un. Pour des raisons de simplicité, seul CoCNN3 a été utilisé pour évaluer les performances. La figure III.9 montre les valeurs moyennes de HR et de NDCG en fonction de la taille des embeddings. Comme le montre la figure, nous observons que les deux scores du modèle présentent une tendance commune en ce sens que leurs courbes augmentent d'abord et diminuent ensuite. Au début, les

embeddings de plus grande dimensionnalité, contenant plus de caractères d'items et d'informations sur les préférences des utilisateurs, améliorent les performances du modèle. Ensuite, les embeddings de trop grande dimensionnalité, qui contiennent trop d'informations redondantes, augmentent le risque de surajustement. Par conséquent, nous concluons qu'une taille d'embeddings appropriée est effectivement utile pour améliorer la recommandation.

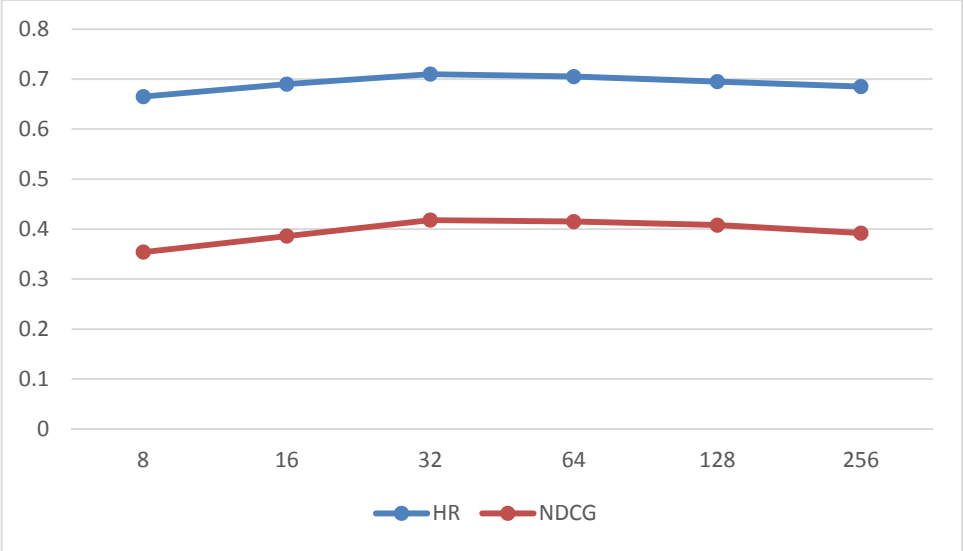


FIGURE III.9 : Impact de la taille des embeddings.

III.7.2 Evaluation du modèle NCF

Nous évaluons les performances de l'algorithme NCF en variant l'hyperparamètres: n_factors (taille des facteurs). Le graphe III.10 présente la variation des deux métriques d'évaluation HR et NDCG en fonction de l'hyperparamètre n_factors. Nous avons choisi n_factors parmi {8, 16, 32, 64}

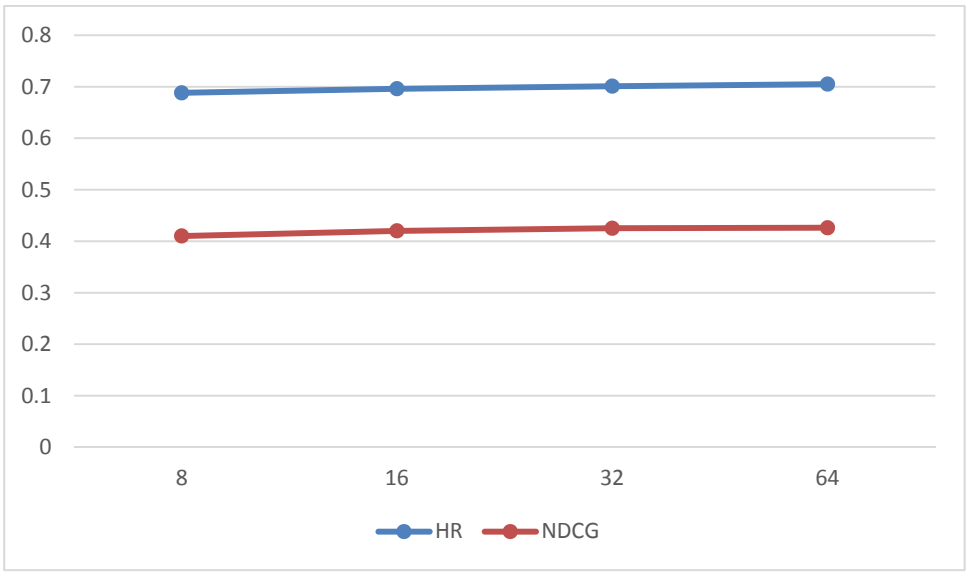


FIGURE III.10 : Impact de l'hyperparamètren_factors (NCF)

II.7.3 Evaluation du modèle KNN

Nous avons testé le modèle avec ses hyperparamètres à fin de déterminer les valeurs optimales qui obtiennent les meilleurs résultats en terme de NDCG et HR. Nous avons testés le modèle pour différentes valeur de k_neighbors [5 -100]et avec différentes mesures de similarité tels que : Cosinus, Euclidien, Manhattan, Jaccard. Nous remarquons que le modèle KNN donne les meilleurs résultats HR est proche de 0.66 et NDCG est proche de 0.39 lorsque la valeur de k_neighbors est égale à 10et que la mesure de similarité est « Cosinus ».

III.7.4. Comparaison des performances globales

Nous présentons dans cette section les résultats des expériences qui ont été réalisées pour vérifier les performances des différents modèles étudiés à savoir CoCNN (avec sa version simplifiée CNN), NCF et KNN.Les métriques(HR et NDCG) obtenues par les algorithmes implémentés pour notre jeu de donnéesMovielenssontprésentés dans le tableau III.7.

Selon les résultats obtenus sur l'ensemble de données movielens, le modèle CoCNN est plus performants que les autres modèles (CNN, NCF et KNN) ce qui montre que dans les tâches de recommandation, le modèle a une forte capacité à faire des prédictions. CNN, NCF et KNN sont moins performants que CoCNN, la raison potentielle est que les deux méthodes sont modélisées sur des paires utilisateur-item et ne prennent pas en compte les effets des relations entre les items.

	CoCNN	**CNN**	**NCF**	**KNN**
HR	0.719	0.676	0.705	0.655
NDCG	0.426	0.403	0.424	0.386

Tableau III. 7 : Comparaison des performances des différents modèles

III.8 Conclusion

Dans ce chapitre, nous avons présenté les résultats de l'implémentation des différents modèles de recommandation basés sur les algorithmes de machine et deep learning.

Précisément, l'objectif de notre travail est dans un premier temps d'implémenter et d'évaluer les trois modèles : La cooccurrence CNN pour la recommandation (CoCNN), CNN pour la recommandation (CNN) qui est un modèle simplifié de CoCNN, le filtrage collaboratif neuronal (NCF), le filtrage collaboratif en utilisant k-voisins les plus proches (KNN), en utilisant un ensemble de données movielens et un ensemble de métriques, et ensuite de comparer les résultats d'évaluation obtenus par les différents modèles avec les valeurs optimales de leurs hyperparamètres.

Nous avons constaté que le modèle CoCNN est celui qui a affiché les meilleurs résultats avec toutes les métriques utilisées.

Conclusion générale

Les systèmes de recommandation automatique sont devenus, tout comme les moteurs de recherche, des outils essentiels pour tout site Web proposant un catalogue riche d'articles variés, qu'il s'agisse d'objets, de produits culturels (livres, films, morceaux de musique, etc.), d'éléments d'information (nouvelles) ou même de simples pages Web (liens hypertextes). L'objectif de ces systèmes est de sélectionner, dans leur catalogue, les éléments les plus susceptibles d'intéresser un utilisateur particulier. Un large éventail de systèmes de recommandation a été développé pour différents domaines, tant académiques qu'industriels.

Actuellement, les systèmes de recommandation tendent vers des méthodes innovantes, multicritères, multidimensionnelles ou basées sur des concepts psychologiques tels que les émotions et les opinions. Cependant, un système de recommandation doit avant tout s'adapter aux données qu'il propose à l'utilisateur, ce qui signifie que le choix de la méthode de recommandation doit principalement être guidé par ce critère.

Le travail présenté dans ce mémoire s'inscrit dans le cadre du filtrage collaboratif, la méthode la plus importante et la plus couramment utilisée dans les systèmes de recommandation. Nous avons mené une étude expérimentale comparative entre plusieurs algorithmes de filtrage collaboratif : l'un est basé sur les techniques de machine learning (KNN) et les autres sur l'apprentissage profond (CoCNN, CNN et NCF). Dans notre cas, nous avons constaté que l'algorithme CoCNN a donné de meilleurs résultats par rapport aux autres algorithmes testés.

Pour ce faire, nous avons d'abord fourni une vue d'ensemble sur le machine learning et le deeplearning et introduit les différents types d'apprentissage. Ensuite, nous avons présenté les réseaux de neurones et leurs principales architectures. Dans un second temps, nous avons défini les systèmes de recommandation et les approches de filtrage, ainsi que les différents types et quelques travaux de l'état de l'art basés sur le machine learning et le deeplearning. Enfin, nous avons présenté notre travail en introduisant l'implémentation des algorithmes basés sur le machine learning et le deeplearning, discuté les résultats d'évaluation et de comparaison, et spécifié le jeu de données, les métriques d'évaluation, l'environnement de développement et les outils utilisés pour la réalisation.

Ce travail n'a pas été exempt d'obstacles. En effet, nous avons été confrontés à différentes difficultés, par exemple au niveau de l'implémentation des algorithmes du deeplearning qui nécessite des ressources matérielles considérable.

Comme perspectives par rapport à ce travail, il serait intéressant de faire cette étude sur différents jeux de données pour montrer si le choix de l'algorithme dépend du jeu de données, de sa taille et aussi des hyperparamètres du l'algorithme.

Bibliographie

Adomavicius, G., & Tuzhilin, A. (2005). oward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering , 17, no 6, 734-749.

Agatonovic, K., & Beresford, R. (2000). Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. Journal of Pharmaceutical and Biomedical Analysis , 22, no 5, 717-727.

Ait Ahmed, N., & Driss Khodja, A. (2018). Système de Recommandation de Cours à Base d'Ontologie.

Akram, B. (2020). La proposition d'une nouvelle approche basée Deep Learning pour la prédiction du cancer du sain. Oum El Bouaghi.

Analytics Yogi. (s.d.). Récupéré sur https://vitalflux.com/different-types-of-cnn-architectures-explained-examples/

Azzoune, I., & Khaldoun, N. (2019-2020). Une approche IA pour la reconnaissance des expressions faciales. Bouira, Informatique, Algérie. Bouira, Informatique, Algérie.

Bastin, J. (2020). Etude des systèmes de recommandations et mise en pratique des algorithmes.

Belhaouci, D. (2019, avrile 30). Démystifier le Machine Learning, Partie 2 : les Réseaux de Neurones artificiels. juripredis .

Benosman, I., & Charif, n. (2015-2016). Utilisation de l'approche pré-filtrage contextuel des systèmes de recommandation sensible au contexte.

Boughaba, M., & Boukhris, B. (2016-2017). L`apprentissage profond (Deep Learning) pour la classification et la recherche d'images par le contenu. Mémoire Master Professionnel . d'Informatique et des Technologies de l'information.

brownlee, j. (2019, july 5). A Gentle Introduction to Deep Learning for Face Recognition. Deep Learning for Computer Vision .

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction , 12, 331-370.

Chen, M., MA, T., & Zhou, X. (2022). CoCNN: Co-occurrence CNN for recommendation. 195 , 116595. Expert Systems with Applications.

Dahah, R. L. (2020, septembre). La Détection de la colère chez le conducteur en utilisant le Deep learning. Mémoire Présenté pour obtenir le diplôme de master académique en informatique . biskra, informatique, algerie.

datascientest. (s.d.). Consulté le mars 15, 2024, sur https://datascientest.com/deep-learning-definition

DataScientest. (s.d.). Consulté le mars 21, 2024, sur https://datascientest.com/transfer-learning

Ferrari Dacrema, M., Cremonesi, P., & Jannach, D. (2019). Are we really making much progress? A worrying analysis of recent neural recommendation approaches. 101-109. Proceedings of the 13th ACM conference on recommender systems.

Fethi, A., & Litim, Y. (2021). Détection des attaques DDOS dans le Cloud Computing. (Doctoral dissertation, university center of abdalhafid boussouf-MILA).

Futura. (s.d.). Consulté le mars 12, 2024, sur https://www.futura-sciences.com/tech/definitions/intelligence-artificielle-deep-learning-17262/). (s.d.).

Geekflare. (s.d.). Consulté le mars 20, 2024, sur https://geekflare.com/fr/neural-networks/

geeksforgeeks. (s.d.). Récupéré sur https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/

Goldberg, D., Nichols, D., OKI, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. Communications of the ACM , 35, no 12, 61-70.

Guo, H., Tang, R., & YE, Y. (2017). DeepFM: a factorization-machine based neural network for CTR prediction.

HAL thesis. (s.d.). Consulté le MARS 25, 2024, sur https://tel.archives-ouvertes.fr/tel-00719609/document

He, X., Liao, L., Zhang, H., Nie, L., & Hu, X. (2017). Neural collaborative iltering iltering. 173-182. Proceedings of the 26th International Conference on World Wide Web, WWW '17.

Jeremy Jordan. (s.d.). Consulté le mars 25, 2024, sur https://www.jeremyjordan.me/autoencoders/

Kabouche, Z. (2022, juin). Développement d'un système d'aide au diagnostic des pathologies des poumons : application d'un apprentissage profond.

Karaouzene, m. (2015, juin 11). Système de recommandation des services web sémantiques.

lebigedata.fr. (s.d.). Consulté le mars 1, 2024, sur https://www.lebigdata.fr/deep-learning-definition.%20(s.d

Lecun, Y., Yoshua, B., & Geoffrey, H. (2015). Deep learning. 521, no 7553 , 436-444. nature.

Liu, W., Wang, Z., LIU, X., & Alsaadi, F. (2017). A survey of deep neural network architectures and their applications. 234 , 11-26.

Medium. (s.d.). Consulté le mars 21, 2024, sur https://towardsdatascience.com/softmax-activation-function-explained-a7e1bc3ad60

Medium. (s.d.). Consulté le mars 25, 2024, sur https://towardsdatascience.com/recommender-system-using-bayesian-personalized-ranking-d30e98bba0b9

Movielens. (s.d.). Consulté le mai 1, 2024, sur https://d2l.ai/chapter_recommender-systems/movielens.htmL

Nacer, F. (2019, julliet). Reconnaissance d'expression faciale à partir d'un visage réel. Guelma, informatique, algerie.

Pinnapareddy, N. R. (2018). Deep Learning Based Recommendation Systems.

Quora. (s.d.). Récupéré sur https://fr.quora.com/Quelles-sont-les-diff%C3%A9rentes-techniques-de-r%C3%A9gularisations-en-apprentissage-automatique

Rao, k. (2008). Application domain and functional classification of recommender systems--a survey. DESIDOC Journal of Library & Information Technology , 28, no 3, 17.

Rawal, A., Fink, D., FRANCE, O., Raju, B., Navruzayn, A., Shahrzad, H., et al. (2019). Evolving deep neural networks, in Artificial Intelligence in the Age of Neural Networks and Brain and computing. 269-287. (Elsevier, Éd.)

ResearchGate. (s.d.). Consulté le mars 15, 2024, sur https://www.researchgate.net/figure/Modele-dun-neurone-biologique_fig7_319939107

ResearchGate. (s.d.). Consulté le mars 21, 2024, sur https://www.researchgate.net/figure/Neural-network-before-a-and-after-b-applying-dropout-regularization-technique-185_fig2_346003426

Saint-Cirgue, G. (2019). Apprendre le machine learning en une semaine.

Sarwar, B., Karypis, G., & Konstan, J. (2001). Item-based collaborative filtering recommendation algorithmsp. 285-295. Proceedings of the 10th international conference on World Wide Web.

Schmitt, T. (2018). Appariements collaboratifs des offres et demandes d'emploi. Thèse de doctorat . Université Paris-Saclay (ComUE).

Sedhain, S., Menon, A. K., & Sanner, S. (2015). Autorec: Autoencoders meet collaborative filtering . 111-112. Proceedings of the 24th international conference on World Wide Web.

Senantic Scholar. (s.d.). Consulté le MARS 25, 2024, sur https://www.semanticscholar.org/paper/Cartes-de-communaut%C3%A9s-pourl'adaptation-interactive-Nguyen-Denos/92979cdc814c089a7b13a57a0eeb632801f43a23

Shervine Amidi. (s.d.). Récupéré sur https://stanford.edu/~shervine/l/fr/teaching/cs-230/pense-bete-reseaux-neurones-convolutionnels

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. Advances in Artificial Intelligence , 2009, no1, 421425.

SuperDataScience. (s.d.). Consulté le mars 21, 2024, sur https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-3-flattening

Wikipedia. (s.d.). Consulté le mars 21, 2024, sur https://fr.wikipedia.org/wiki/Sigmo%C3%AFde_(math%C3%A9matiques)

Zhang, Y., & CHEN, X. (2020). Explainable recommendation: A survey and new perspectives. 14, no 1 , 1-101. Foundations and Trends® in Information Retrieval.