

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Blida -1-



Faculté des Sciences

Département d'Informatique

Mémoire de fin d'étude présenté en vue d'obtention du diplôme

MASTER EN INFORMATIQUE

Option : Systèmes Informatiques & Réseaux

Thème

**Intégration de l'aspect social dans
la recommandation des services cloud**

Présenté par :

M. BRAIRI ISMAIL

M. ESSALHI MOHAMED AMINE

Soutenus le 04 octobre 2021, devant le jury :

Mme Mancer Yasmine (Présidente)

Mme Lahiani Nesrine (Examinatrice)

Mme Bey Fella (Promotrice)

Mme Kichou Saida (Encadreuse)

Promotion: 2020/2021

Remerciement

Avant tout nous formulons notre gratitude à Allah le tout puissant de nous avoir donné la force d'achever ce travail,

Nous tenons aussi à remercier : Mme BEY.F, Mme KICHOU.S notre promotrices, pour leurs conseils, leurs orientations, et leurs disponibilité, qui nous ont permis de réaliser ce travail dans les meilleures conditions,

Ainsi les membres du jury, pour avoir fait l'insigne honneur d'accepter de lire et juger ce travail,

Un très grand merci à tous les enseignants de notre département, qui ont assuré notre enseignement pendant tout notre parcours académique.

Sans oublier de remercier nos parents pour leur contribution, leur soutien et leur patience, nos proches, nos amis, nos collègues et toutes personnes qui nous ont aidées par leur soutien permanent de près ou de loin de nos études.

ملخص

تعتبر الخدمات السحابية ذات أهمية كبيرة اليوم ، وظهرت أهمية هذه الخدمات خلال الأزمة التي أثرت على العالم في هذه الحالة جائحة كوفيد 19 ، حيث زاد استخدامها بشكل كبير بسبب إجراءات الحجر الصحي المتخذة في معظم دول العالم. بسبب انتشار الخدمات السحابية ، يواجه العميل صعوبات في اختيار الخدمة السحابية المناسبة التي تلبى احتياجاته. بالإضافة إلى ذلك ، مع توفر عدد كبير من الخدمات السحابية التي توفر نفس الوظائف وتفي بمعايير جودة الخدمة ، من الضروري بالتالي اقتراح خدمات معينة والتوصية بها للعملاء لتلبية احتياجاتهم وتسهيل اختيارهم.

في هذه المذكرة، حاولنا إيجاد حل لمشكلة دمج الجانب الاجتماعي في التوصية بالخدمات السحابية ، وأيضاً مراعاة آراء المستخدمين كمعيار من أجل التوصية بالخدمات التي تلبى الاحتياجات و إرضاء هذا العميل.

لتنفيذ الحل المقترح ، تمت نمذجته في خطوتين ، خطوة استخراج الخصائص من التعليقات ، وخطوة التوصية بالخدمات. وباستعمال لغة البرمجة بايثون، تم تطوير نظام لتوصية الخدمات السحابية ، وقد اثبتت التجارب مدى فعالية الحل المقترح.

الكلمات المفتاحية: الخدمات السحابية ، الجانب الاجتماعي ، التوصية ، تعليقات المستخدم.

Résumé

Les services cloud sont d'une grande importance à l'heure actuelle, et l'importance de ces services est apparue lors de la crise qui a touché le monde en l'occurrence la pandémie de Covid 19, où leur utilisation a considérablement augmenté en raison des mesures de quarantaine prises dans la plupart des pays du monde. En raison de la prolifération des services cloud, le client est confronté à des difficultés pour choisir le service cloud approprié qui répond à ses besoins. En outre, avec la disponibilité d'un grand nombre de services cloud fournissant les mêmes fonctionnalités et répondant aux critères de la QoS, il est donc primordial de suggérer et recommander au client certains services pour satisfaire ses besoins et lui faciliter le choix.

Dans ce mémoire, nous avons essayé de trouver une solution au problème d'intégration de l'aspect social dans la recommandation des services cloud, et aussi de prendre en compte les avis des utilisateurs comme critère afin de recommander des services qui répondent aux besoins et à la satisfaction de ce client.

Pour mettre en œuvre la solution proposée, celle-ci a été modélisée en deux étapes, l'étape d'extraction de caractéristiques des commentaires, et l'étape de recommandation de services. Et ce en utilisant le langage de programmation Python, un système a été développé pour recommander des services cloud, et des tests ont prouvé l'efficacité de la solution proposée.

Mots clés : Services Cloud, Aspect Social, Recommandation, Avis des Utilisateurs.

Abstract

Cloud services are of great importance today, and the importance of these services emerged during the crisis that affected the world in this case the Covid 19 pandemic, where their use has increased considerably due to quarantine measures taken in most countries of the world. Due to the proliferation of cloud services, the customer faces difficulties in choosing the appropriate cloud service that meets their needs. In addition, with the availability of a large number of cloud services providing the same functionality and meeting QoS criteria, it is therefore essential to suggest and recommend certain services to the customer to meet their needs and facilitate their choice.

In this thesis, we have tried to find a solution to the problem of integrating the social aspect in the recommendation of cloud services, and also to take into account the opinions of users as a criterion in order to recommend services that meet the needs and to the satisfaction of that customer.

To implement the proposed solution, it was modeled in two steps, the step of extracting characteristics from comments, and the step of recommending services. By using the Python programming language, a system has been developed to recommend cloud services, and tests have proven the effectiveness of the proposed solution.

Key words: Cloud Services, Social Aspect, Recommendation, User Feedback.

Table des matières

INTRODUCTION GÉNÉRALE.....	1
1 Introduction.....	1
2 Problématique	1
3 Objectifs.....	2
4 Organisation du mémoire	2
CHAPITRE I : GÉNÉRALITÉS SUR LE CLOUD COMPUTING	4
1 Introduction.....	5
2 Définition du cloud computing	5
3 Modèles de déploiement du cloud computing	6
1.1 Cloud privé.....	6
1.2 Cloud public.....	7
1.3 Cloud communautaire	7
1.4 Cloud hybride.....	7
4 Types de services dans les environnements de cloud computing	8
4.1 Infrastructure as a Service (IaaS).....	8
4.2 Platform as a Service (PaaS)	9
4.3 Software as a Service (SaaS)	9
5 Acteurs du cloud computing.....	10
5.1 Consommateur du Cloud (Cloud consumer).....	11
5.2 Fournisseur du Cloud (Cloud Provider)	11
5.3 Courtier du Cloud (Cloud Broker).....	12
5.4 Auditeurs du cloud	13
5.5 Transporteur du Cloud (Cloud Carrier)	13
6 Caractéristiques du cloud computing	13
7 Avantages et inconvénients du cloud computing	15
7.1 Avantages du cloud computing	15
7.2 Inconvénients du cloud computing	15
8 Qualité de service dans un environnement cloud	16
8.1 Caractéristiques.....	16
9 Conclusion	17

CHAPITRE II : LA RECOMMANDATION DES SERVICES CLOUD	18
1 Introduction.....	19
2 Système de recommandation basé sur le cloud	19
2.1 Définition du Système de Recommandation	19
3 Techniques utilisées dans le Système de Recommandation	20
3.1 Approche de filtrage collaboratif	20
3.2 Approche de filtrage basée sur le contenu (content-based).....	21
3.3 Approche hybride.....	22
4 Défis et limites du Système de Recommandation	23
4.1 Rareté des données (Data Sparseness)	23
4.2 Évolutivité (Scalability).....	23
4.3 Démarrage à froid (Cold Start).....	23
4.4 Confidentialité	24
4.5 Confiance.....	24
5 Travaux connexes	24
6 Étude comparative	31
6.1 Les critères de comparaison des solutions	33
6.1.1 Classification selon les techniques utilisées dans le Système de Recommandation	34
6.1.2 Comparaison selon les critères utilisées dans le Système de Recommandation 36	
7 Apprentissage profond (Deep Learning) pour les systèmes de recommandation	38
7.1 Techniques d'apprentissage profond pour les systèmes de recommandation	39
7.1.1 Méthodes d'intégration (Embedding Methods)	39
7.1.2 Réseaux Feedforward et Auto-encoder pour le filtrage collaboratif	39
7.1.3 Méthodes d'extraction de Deep Feature	39
7.1.4 Recommandation basée sur la session avec des réseaux de neurones récurrents 40	
8 Conclusion	40
CHAPITRE III : CONCEPTION D'UN MODÈLE DE SYSTÈME DE RECOMMANDATION	42
1 Introduction.....	43
2 Principe général	43
3 Architecture globale du modèle.....	44

4	Extraction de caractéristiques et classification basée sur Auto-encoder	45
4.1	Extraction des caractéristiques (Features Extraction)	45
4.1.1	Prétraitement	46
4.1.2	Vectorisation	48
4.2	Classification de texte	50
5	Recommandation des services cloud	53
5.1	Recommandation basée sur la popularité	53
5.2	Filtrage collaboratif basé sur l'utilisateur	54
6	Conclusion	55
CHAPITRE IV : EXPÉRIMENTATION ET RÉSULTATS		56
1	Introduction	57
2	Matériel informatique (Hardware)	57
3	Logiciel (Software)	57
4	Les Bibliothèques	58
5	Base de données (Dataset)	58
6	Implémentation	59
7	Entraînement et validation	60
8	Test	61
9	Recommandation des services cloud	61
9.1	Recommandation basée sur la popularité (Classification des services)	61
9.2	Filtrage collaboratif basé sur la similarité des utilisateurs	62
10	Code source	63
11	Métriques d'évaluation	68
12	Résultat	70
13	Conclusion	73
CONCLUSION GÉNÉRALE		74
BIBLIOGRAPHIE		76

Liste des Figures

Figure 1: Principe du Cloud Computing.....	6
Figure 2: Types de services dans le Cloud computing [4].....	8
Figure 3: Différences de portée et de contrôle entre les modèles de service cloud [7].	10
Figure 4: Architecture de référence du Cloud Computing [8].....	11
Figure 5: modèle visuel de la définition du cloud computing d'après NIST [11].	14
Figure 6: Filtrage collaboratif.	21
Figure 7: Filtrage basé sur le contenu.	22
Figure 8: L'organigramme du modèle en deux étapes proposées [15].	25
Figure 9: L'organigramme de sélection de service cloud proposé par [16].	27
Figure 10: Classification des solutions étudiées sur les systèmes de recommandation.....	35
Figure 11: Traitement de la recommandation des services selon les techniques utilisées.	36
Figure 12: Architecture de notre modèle de système de recommandation.	44
Figure 13: Un exemple de suppression des chiffres et de la ponctuation.	46
Figure 14: Un exemple de tokenisation d'un texte.....	47
Figure 15: Un exemple de suppression des mots vides (StopWords).....	48
Figure 16: Modèle Word2vec (Continuous Bag of Words (CBOW) et Skip-gram [36].	50
Figure 17: Un schéma descriptif de la technique Auto-encoder.....	51
Figure 18: FC basé sur la similarité des utilisateurs.	55
Figure 19: La structure du fichier Reviews.csv.	59
Figure 20: L'affichage de résultats des prédictions.....	61
Figure 21: L'interface graphique de notre system de recommandation.	63
Figure 22: Les données téléchargées et les packages importés.	64
Figure 23: La fonction de prétraitement que nous avons créée.	64
Figure 24: L'extraction des caractéristiques de notre dataset à l'aide de Word2vec pour l'étape de vectorisation.	65
Figure 25: L'initialisation des deux classes de commentaires.	65
Figure 26: Extraction de caractéristiques pour la classification et la compilation de l'Auto-encoder.....	66
Figure 27: Calcul de la moyenne des commentaires positifs.....	66
Figure 28: Calcul de WR.	66
Figure 29: La recommandation des meilleurs services selon WR.	67
Figure 30: La recommandation des services basée sur les utilisateurs similaires.	67
Figure 31: La description de l'interface graphique.	68
Figure 32: L'exactitude de l'entraînement et de la validation.	71
Figure 33: Le volume de perte de notre modèle.	72
Figure 34: Les résultats de recommandation pour chaque utilisateur.....	72

Liste des tableaux

Tableau 1: Comparaison des modèles de services de Cloud Computing [3].....	7
Tableau 2: Comparaison entre les approches proposées.	31
Tableau 3:Tableau comparatif des approches de recommandation de service Cloud.	37
Tableau 4:Toutes les couches avec les tailles de jetons et les fonctions d'activation.	53
Tableau 5: Les résultats de notre modèle.....	70
Tableau 6: Les résultats du rappel et la précision.	71

INTRODUCTION GENERALE

1 Introduction

Avec l'avènement d'Internet dans les années 1990 aux installations informatiques actuelles, Internet a radicalement changé le monde de l'informatique. Il est passé du concept d'informatique parallèle à l'informatique distribuée à l'informatique en grille et plus récemment au cloud computing.

« Le Cloud computing marque une nouvelle avancée vers l'infrastructure informatique dématérialisée. Le Cloud fournit des ressources informatiques, logicielles ou matérielles, accessible à distance en tant que service » [13].

Avec la croissance rapide de la plateforme de services cloud, il est de plus en plus difficile pour les utilisateurs de découvrir des services cloud répondant à leurs demandes et satisfierait leurs besoins. Dans ce cas, il s'agit d'un système de recommandation des services cloud pour donner à l'utilisateur le résultat attendu.

La thématique de recherche nous a été proposée au niveau du Centre de recherche sur l'information scientifique et technique (CERIST). C'est l'un des axes de recherche parmi plein d'autres qui sont traités au sein de cet établissement. Le CERIST, est un établissement public algérien à caractère scientifique et technologique sous la tutelle du ministère de l'enseignement supérieur et de la recherche scientifique. Il a été créé en 1985, et avait pour mission principale de mener toute recherche relative à la création, à la mise en place et au développement d'un système national d'information scientifique et technique. Le CERIST est organisé en départements administratifs et techniques et en divisions de recherche. Outre le siège central fixé à Alger, le centre dispose de sites régionaux.

2 Problématique

Il existe dans la littérature une panoplie de travaux visant à améliorer le processus de recommandation de manière générale et la recommandation de services cloud de manière particulière. Pour cette dernière, les travaux se basent majoritairement sur la QoS (la Qualité de Services) des services cloud. L'avènement des médias sociaux a permis aux utilisateurs d'être

producteurs d'information via leurs participation directe par des commentaires, tags, votes, etc. Ceci a procuré une masse de données importantes pouvant apporter d'avantages d'informations sur les utilisateurs.

Beaucoup de travaux s'intéressent aux préférences de l'utilisateur comme critère, mais jusqu'ici, la préférence de l'utilisateur est limitée dans un contexte et environnement liés au service cloud en question (les avis et préférences des utilisateurs par exemple dans la page de description du service par son auteur) et non dans les médias sociaux grand public. Il existe de multiples sources d'informations qui peuvent être utilisées pour comprendre les besoins réels des utilisateurs, et donc rendre la recommandation et la sélection des services Cloud plus précises et efficaces.

La problématique de ce travail consiste à l'intégration des avis utilisateurs dans les réseaux sociaux sur les services cloud, pour améliorer la recommandation de ceux-ci.

3 Objectifs

Les objectifs visés à travers ce travail sont les suivants :

1. Étude bibliographique sur la recommandation des services cloud de manière générale.
2. Étude bibliographique sur la recommandation des services cloud intégrant l'aspect social.
3. Étude comparative et proposition d'une nouvelle approche intégrant l'aspect social dans la recommandation de services cloud.
4. Proposition d'une approche de recommandation de service cloud basée sur les avis des utilisateurs.
5. Réalisation et validation de l'approche proposée.

4 Organisation du mémoire

La structure de ce document est organisée comme suit :

Le premier chapitre est un chapitre de généralités, il représente une vue globale sur le domaine du Cloud Computing, il comporte des définitions pour les différents concepts, et il cite les modèles de déploiement du Cloud, les types, les acteurs, les caractéristiques, ainsi que les avantages et inconvénients du Cloud Computing.

Le deuxième chapitre est consacré à l'étude de la recommandation des services cloud, il se divise

en trois parties. La première partie décrit les différentes techniques utilisées dans le système de recommandation, elle montre aussi les défis et limites de ceux-ci. La deuxième partie est consacrée à l'analyse des travaux connexes, pour avoir une idée sur les différentes solutions qui existent et pour définir les objectifs relatifs au processus de recommandation, et se termine par une étude comparative de ces travaux. La troisième partie présente un ensemble de techniques de l'apprentissage profond intégré pour améliorer le processus de recommandation.

Le troisième chapitre, ici nous allons définir l'architecture de notre nouvelle solution, avec la spécification des différents besoins du système. Nous allons spécifier et détailler les processus nécessaires de l'intégration de l'aspect social dans la recommandation des services cloud.

Le quatrième chapitre dans le dernier chapitre, nous présenterons les différents outils utilisés pour construire notre modèle. Nous avons testé notre proposition de la solution du problème de recommandation et nous avons montré par les expérimentations la performance de notre proposition.

CHAPITRE I : GENERALITES SUR LE CLOUD COMPUTING

1 Introduction

Avec l'invention et le développement d'Internet, tous les objets du monde sont interconnectés, et avec l'ambition humaine de s'étendre et d'être présent partout, un besoin est apparu, à savoir l'accessibilité et la disponibilité des informations personnelles, de n'importe où et à tout moment, pour les entreprises comme pour les particuliers.

D'un autre côté, le développement technologique a mené à l'amélioration des équipements informatiques, en termes de la capacité du stockage, de la vitesse de calcul et le taux de transfert. En essayant de bénéficier des précédents facteurs, afin de répondre aux exigences d'accessibilité et de disponibilité de l'information, cela a donné naissance au Cloud Computing qui a permis d'offrir plusieurs types de services avec une meilleure utilisation des ressources, des infrastructures et une grande réduction du coût d'exploitation.

Aujourd'hui, le Cloud est présent partout avec un intérêt croissant grâce aux offres des ressources à la demande. Dans ce chapitre, nous allons présenter une description générale de la technologie du Cloud Computing avec ces définitions, ces types, ces caractéristiques et ces éléments constitutifs. Nous terminons par une conclusion.

2 Définition du cloud computing

Il existe plusieurs définitions du cloud computing, parmi ces définitions celle de NIST (National Institute of Standards and Technology) qui définit le cloud computing comme un modèle qui permet un accès omniprésent, pratique et à la demande à un réseau partagé et à un ensemble de ressources informatiques configurables (comme exemple : des réseaux, des serveurs, du stockage, des applications et des services) qui peuvent être provisionnées et libérées avec un minimum d'administration. Ce modèle est composé de cinq caractéristiques essentielles, de trois modèles de services et de quatre modèles de déploiement [1].

Une autre définition proposée par Microsoft qui définit le cloud computing par la fourniture de services informatiques (notamment des serveurs, du stockage, des bases de données, la gestion réseau, des logiciels, des outils d'analyse, l'intelligence artificielle) via Internet, dans le but d'offrir une innovation plus rapide, des ressources flexibles et des économies d'échelle. En règle générale, vous payez uniquement les services cloud que vous utilisez (réduisant ainsi vos coûts

d'exploitation), gérez votre infrastructure plus efficacement et adaptez l'échelle des services en fonction des besoins de votre entreprise [2].

La figure ci-dessus donne une vue générale sur le Cloud Computing où les utilisateurs peuvent accéder et utiliser des services cloud sur des réseaux étendus (internet comme exemple) via un terminal.

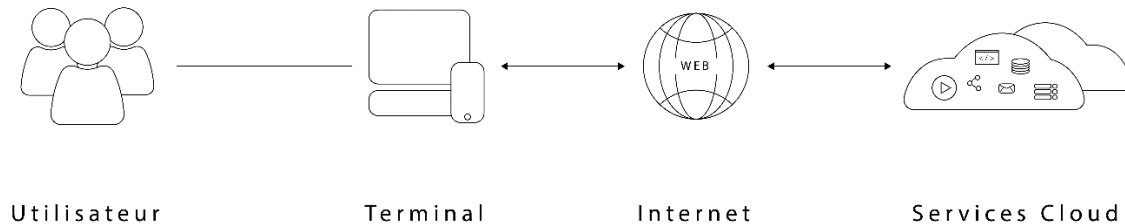


Figure 1: Principe du Cloud Computing

D'après les définitions précédentes, le cloud est donc un modèle où on consomme des ressources informatiques comme des applications, des serveurs, une puissance de calcul ou l'espace de stockage. Sans posséder d'infrastructure technique qui est chère. Le cloud, et ses services informatiques sont accessibles via un réseau étendu comme Internet.

3 Modèles de déploiement du cloud computing

Le Cloud Computing joue un rôle important pour répondre à la demande croissante de l'utilisation des différents services soit pour les entreprises ou autre type d'utilisateur. Globalement, le cloud offre quatre modèles de déploiement comme suit :

1.1 Cloud privé

Ce type de cloud fonctionne pour une organisation ou une entreprise définie, par exemple cloud pour une organisation spécifique [3].

Selon la définition d'Intel, l'infrastructure cloud est utilisée par les organisations. Le cloud privé est hébergé dans le data center et géré par le service informatique de l'entreprise. Les avantages de cloud privé sont plus de sécurité et de protection des données, un meilleur contrôle et une efficacité énergétique et économique.

1.2 Cloud public

Les clouds publics sont facilement disponibles auprès de Google, Amazon, Microsoft, etc. Le cloud public fournit une infrastructure et des services au public ou à toute organisation. Les ressources sont partagées par des centaines ou des milliers de personnes [3].

Lors de l'utilisation d'un cloud public, la responsabilité de la gestion et de la maintenance du système est directement assumée par le fournisseur.

1.3 Cloud communautaire

Dans un cloud communautaire, les services et l'infrastructure sont fournis à des organisations ayant des intérêts similaires [3].

L'infrastructure cloud communautaire est partagée par plusieurs organisations pour répondre aux besoins des communautés qui souhaitent partager des ressources (par exemple, mission, exigences de sécurité, politique, etc.), elle est généralement exploitée par un organisme communautaire ou un tiers et peut être placée dans les locaux ou à l'extérieur.

1.4 Cloud hybride

Ce type de cloud est un mélange de cloud privé et public. Bien que les nuages soient mélangés, chacun a toujours son identité individuelle et facilite donc plusieurs déploiements [3].

Intel a défini que les clouds hybrides combinent des clouds publics et privés afin qu'ils puissent partager des données et des applications entre eux et aider les entreprises à faire évoluer facilement les services entre leur propre infrastructure et les clouds publics.

Le tableau ci-dessus donne une comparaison des modèles de services de cloud computing.

Tableau 1: Comparaison des modèles de services de Cloud Computing [3]

Modèle	Population Cible	Dirigé par	Niveau de sécurité
Modèle public	Grand public et industries	Fournisseurs de services cloud	Faible
Modèle privé	Organisation unique	Organisation unique	Haut

Modèle communautaire	Organisations ayant des politiques similaires et les mêmes problèmes de sécurité	De nombreuses organisations ou fournisseurs de services cloud	Haut
Modèle hybride	Public et organisation	Public et organisation	Moyen

4 Types de services dans les environnements de cloud computing

Le Cloud Computing est principalement composé de trois services qui s’organisent en trois niveaux successifs : le niveau infrastructure (IaaS), le niveau plateforme (PaaS) et le niveau application (SaaS). La figure suivante (figure 2) montre ces types et niveaux [4].

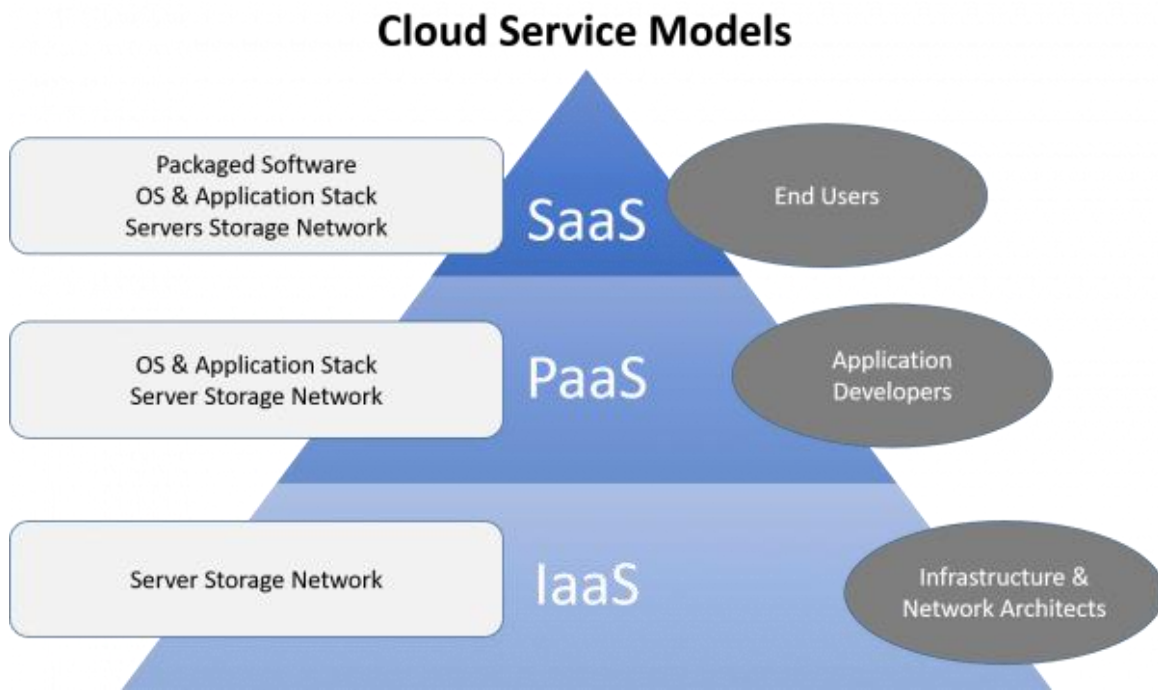


Figure 2: Types de services dans le Cloud computing [4]

Les différents types de services dans l’environnement cloud sont définis comme suit :

4.1 Infrastructure as a Service (IaaS)

Cela permet aux clients d’accéder à distance à certains appareils préconfigurés qu’ils peuvent surveiller et utiliser comme s’ils accédaient au même appareil physique sur le terrain. Il s’agit d’un "serveur virtuel" qui permet aux clients d’utiliser les fonctions d’un serveur traditionnel, tout comme d’accéder à un serveur physique avec les mêmes spécifications techniques. Le terme «

serveur virtuel » est utilisé, car le client n'accède pas à une instance individuelle de matériel située dans les locaux du fournisseur informatique, mais utilise plutôt une proportion définie des ressources partagées d'un puissant centre de données. Avec cette forme d'IaaS, le fournisseur n'est responsable que de la maintenance et du fonctionnement du « serveur virtuel » et du matériel sous-jacent. Le client est responsable de l'exécution et de la maintenance du système d'exploitation et de tous les logiciels et applications exécutés sur le « serveur virtuel » [5].

4.2 Platform as a Service (PaaS)

Il s'agit de la livraison d'IaaS avec l'ajout d'un environnement d'exécution « système d'exploitation » fourni par le fournisseur. Cela permet à un client d'exécuter un logiciel sur les serveurs d'un fournisseur informatique au sein du système d'exploitation virtuel préconfiguré. Sous PaaS, un client a peu ou pas de contrôle sur le système d'exploitation et les ressources matérielles sous-jacents. Contrairement à IaaS, c'est le fournisseur qui est responsable de la fourniture et de la maintenance du système d'exploitation et du matériel sous-jacent. Le client est uniquement responsable de la sélection et de la gestion du logiciel exécuté sur le système d'exploitation virtuel. Des exemples de solutions PaaS incluent Rackspace Cloud Sites, Force.com de Salesforce.com et Google App Engine, Microsoft Azure [5].

4.3 Software as a Service (SaaS)

Le service cloud SaaS désigne un modèle déployé et exécuté sur une plateforme cloud afin que les utilisateurs puissent utiliser le logiciel en tant que service en ligne. Les applications hébergées comme Gmail et Google Docs peuvent être accessibles via différents appareils tels que les smartphones et les ordinateurs portables, etc. Les clients qui utilisent les services cloud SaaS sont moins chargés de l'investissement initial ou de la gestion des systèmes. Cependant, ils doivent payer un montant fixe en fonction de la période de service ou de l'utilisation [6].

Dans ce modèle, les fournisseurs de services cloud sont responsables de l'exécution et de la maintenance des applications software, du système d'exploitation et d'autres ressources.

La figure 3 illustre les différences de portée et de contrôle entre le consommateur de cloud et le fournisseur de cloud pour chacun des modèles de service décrits ci-dessus. Cinq couches conceptuelles d'un environnement cloud généralisé sont identifiées dans le diagramme central et s'appliquent aux clouds publics, ainsi qu'à chacun des autres modèles de déploiement. Les flèches

à gauche et à droite du diagramme indiquent la portée approximative de la portée du contrôle du fournisseur de cloud et du consommateur de cloud sur l'environnement cloud pour chaque modèle de service. En général, plus le niveau de support disponible auprès d'un fournisseur de cloud est élevé, plus la portée et le contrôle du consommateur de cloud sur le système sont étroits.

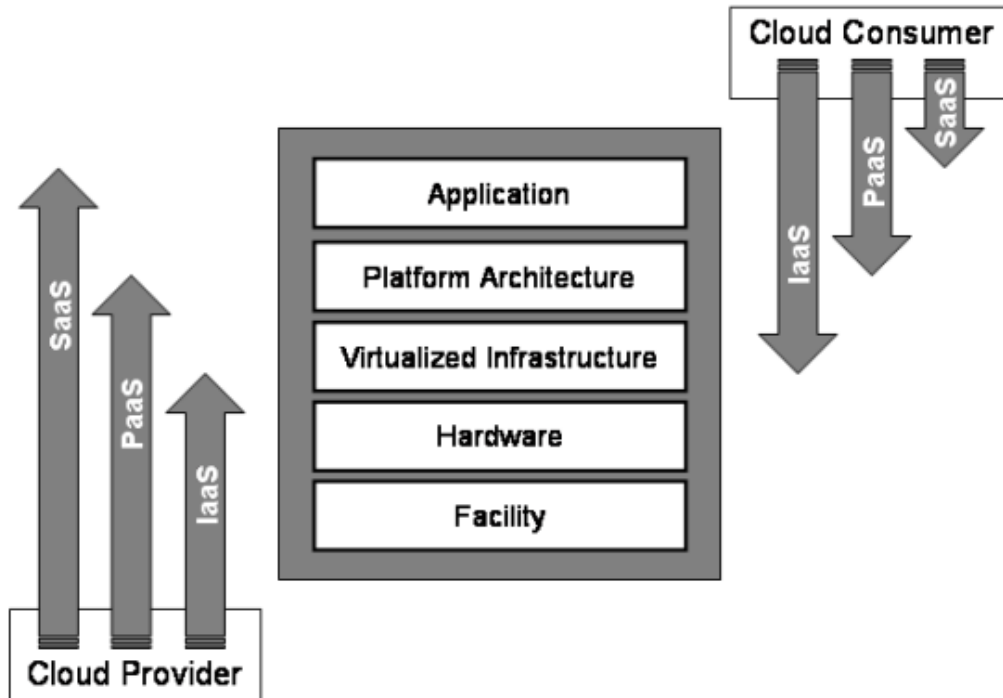


Figure 3: Différences de portée et de contrôle entre les modèles de service cloud [7].

5 Acteurs du cloud computing

Dans l'architecture de référence du Cloud Computing de NIST (voir la figure suivante) qui définit les acteurs majeurs (Cloud Consumer, Cloud Service Provider, Cloud Auditor, Cloud Broker, and Cloud Carrier), et pour bien comprendre le fonctionnement du Cloud computing, il est nécessaire de connaître les différents acteurs du domaine, et le rôle de chacun d'eux.

Les consommateurs de cloud peuvent demander des services cloud aux fournisseurs de services cloud directement ou via cloud broker. Les auditeurs cloud effectuent des audits indépendants et peuvent contacter d'autres personnes pour recueillir les informations nécessaires.

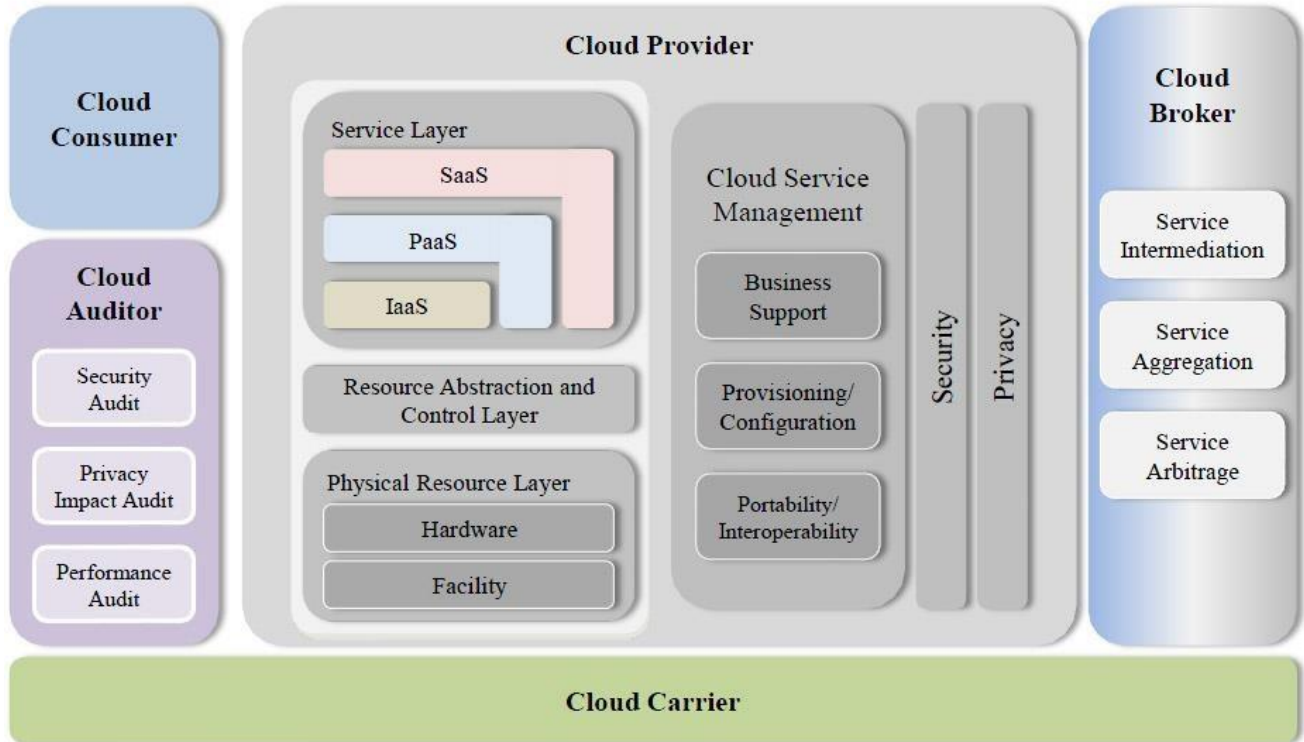


Figure 4: Architecture de référence du Cloud Computing [8]

5.1 Consommateur du Cloud (Cloud consumer)

Il s'agit de l'utilisateur qui consomme les services cloud. Le consommateur de cloud pourrait être l'un des suivants [32]:

- Les développeurs de l'entreprise qui créent les applications à l'aide de l'infrastructure cloud.
- Employés de bureau et consommateurs finaux qui accèdent aux applications de stockage et de productivité.
- Équipe de support informatique qui utilise les services cloud tels que les sauvegardes cloud pour compléter leurs ressources.

5.2 Fournisseur du Cloud (Cloud Provider)

Il s'agit d'une organisation qui fournit le service cloud. Le fournisseur de services cloud possède et contrôle la plateforme de cloud computing. Les services incluent SaaS (Software as a Service), PaaS (Platform as a Service), IaaS (Infrastructure as a Service), parmi les fournisseurs on peut citer Amazon, Google, Microsoft et Sales force [32].

Les fournisseurs de services cloud prêtent attention à la fourniture de services, à la gestion et au contrôle d'allocation des ressources et à la couche de ressources physiques qui constitue le cloud. Les couches de service sont représentées de manière à montrer qu'il est possible d'optimiser chaque couche de service jusqu'à la couche d'abstraction et de contrôle des ressources en plus de construire couche après couche. En outre, ils sont également responsables de toute la gestion du cloud, comme décrit dans la section Gestion des services cloud CSM (Cloud Service Management).

Gestion des services cloud : Le CSM est composé de trois parties [33] :

- **Le support commercial (Business Support) :** y compris tous les services liés à la gestion des affaires des clients et aux processus de support.
- **Approvisionnement/Configuration :** Gérez le déploiement et la configuration de tous les aspects du déploiement, des changements de ressources, de la surveillance et de la mesure.
- **Portabilité/ Interopérabilité :** Prend en charge la portabilité et l'interopérabilité des services et la migration des données entre les clouds.

5.3 Courtier du Cloud (Cloud Broker)

Cloud Service Brokerage (CSB) fournit des services d'intermédiation entre le consommateur et le fournisseur. Le Cloud Broker (responsable du service de courtage) est émergé comme une couche intermédiaire entre les CSP (Cloud Service Provider) et les CSU (Cloud Service User) pour aider les CSU à choisir les meilleurs services dans un environnement de type Cloud. D'autre part, le Cloud Broker peut agir en tant que négociateur pour les CSU avec de nombreux environnements de Cloud. Ainsi, il peut rechercher et réserver les ressources disponibles dans les autres CSP, en fonction de différents niveaux de service [9].

Les consommateurs de cloud exigent des fournisseurs de cloud qu'ils fournissent des services, et les fournisseurs fournissent des services à leurs clients et génèrent des bénéfices financiers, établissant ainsi une relation commerciale entre les clients de cloud et les fournisseurs de cloud. Cloud Broker est un service intermédiaire entre plusieurs fournisseurs de services cloud et consommateurs de services cloud. Cloud Broker fournit des services et d'autres opérations interactives en fonction des besoins des clients.

5.4 Auditeurs du cloud

Les auditeurs du cloud peuvent évaluer les services fournis par les fournisseurs de cloud en termes de contrôle de sécurité, d'impact sur la confidentialité, de performances, etc., afin de mettre en œuvre correctement les contrôles, de fonctionner comme prévu et de fournir les résultats attendus qui répondent aux exigences de sécurité du système [33].

5.5 Transporteur du Cloud (Cloud Carrier)

Cloud Carrier est un agent qui connecte et transfère les services cloud des fournisseurs de cloud aux consommateurs de cloud. Les opérateurs de cloud fournissent un accès aux consommateurs via le réseau, les télécommunications et d'autres dispositifs d'accès. Un fournisseur de cloud mettra en place des SLA (Service-Level Agreement) avec un opérateur de cloud pour fournir des services cohérents avec le niveau de SLA proposé aux consommateurs de cloud. Les consommateurs peuvent demander au fournisseur de services cloud de fournir une connexion cryptée dédiée [33].

6 Caractéristiques du cloud computing

Les systèmes du cloud computing satisfont à de nombreuses caractéristiques intéressantes. Le NIST a défini cinq caractéristiques essentielles des systèmes de cloud computing et nous les décrivons ci-dessous:

Libre-service à la demande (On-demand self-service) : les services cloud tels que le temps de CPU, le stockage, l'accès au réseau, temps du serveur, les applications Web, etc. peuvent être attribués automatiquement selon les besoins des consommateurs sans aucune interaction humaine [10].

Accès au réseau étendu (Broad Network Access) : Les ressources cloud doivent être disponibles sur le réseau et via plusieurs appareils. De même, les ressources doivent être disponibles en utilisant des mécanismes « standard ». Cette caractéristique se concentre sur l'accessibilité des ressources cloud, sans avoir besoin d'un appareil spécifique ou d'un logiciel propriétaire. En règle générale, les ressources sont disponibles via un navigateur Web utilisant des protocoles tels que HTTP, XML ou d'autres [11].

Regroupement des ressources (Resource Pooling) : les ressources informatiques physiques et virtuelles sont mises en commun dans le cloud. Ces ressources ne dépendent pas de l'emplacement

dans le sens où le client n'a aucun contrôle ni n'a connaissance de leur emplacement [10].

Élasticité rapide (Rapid Elasticity) : les ressources informatiques peuvent être rapidement et élastiquement provisionnées et libérées en fonction de la demande du consommateur. Les consommateurs voient ces ressources comme si elles étaient infinies et peuvent être achetées dans n'importe quelle quantité à tout moment [10].

Services mesurés (Measured Services) : Le système cloud surveille et optimise automatiquement l'utilisation des ressources en utilisant des fonctions de mesure à un niveau d'abstraction spécifique correspondant au type de service (tel que le stockage, le traitement, la bande passante et les comptes d'utilisateurs actifs). Les ressources peuvent être suivies, surveillées et signalées, ce qui offre une transparence aux fournisseurs et aux consommateurs des services utilisés. Cela signifie effectivement que le fournisseur suit l'utilisation pour s'assurer que les ressources sont optimisées. Si nous examinons les avantages nets du cloud du point de vue des économies potentielles, cette fonctionnalité deviendra une fonctionnalité incontournable [11].

Les différentes catégories de modèles et d'architectures de service sont représentées graphiquement sur la figure 5.

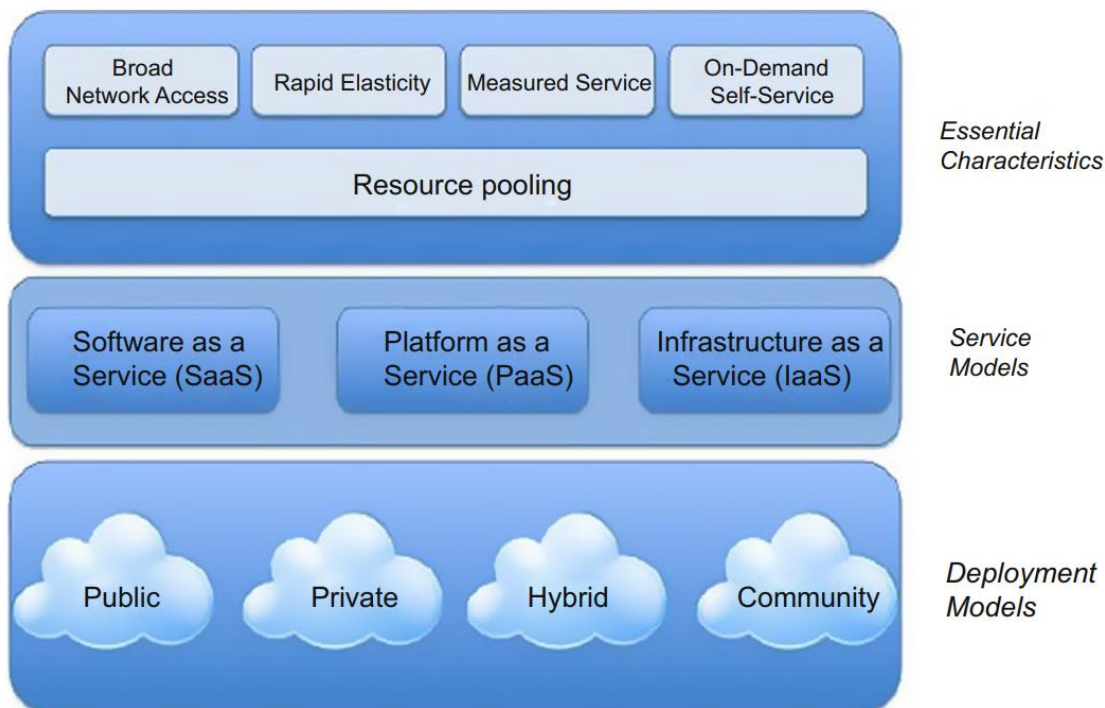


Figure 5: modèle visuel de la définition du cloud computing d'après NIST [11].

7 Avantages et inconvénients du cloud computing

7.1 Avantages du cloud computing

Le cloud computing offre plusieurs avantages qui attirent les utilisateurs. Parmi ces avantages :

Gestion facile : la maintenance de l'infrastructure est simplifiée, des applications faciles à être utilisé dans l'environnement cloud [12].

Réduction des coûts : utilisation du cloud réduit le coût surtout pour les entreprises, il offre une puissance de calcul et de stockage qui permet à l'entreprise d'éviter d'acheter du matériel coûteux et préfère utiliser les services cloud qui ont un coût réduit et des services riches [12].

Services ininterrompus : Le service de cloud computing réduit les temps d'arrêt et offre aux utilisateurs un service ininterrompu. Cependant, il y a eu quelques défaillances dans le passé, par exemple Gmail a été interrompu en 2009 [12].

Gestion des catastrophes : les fournisseurs des services effectuent des sauvegardes de manière circulaire pour sauvegarder les données dans des sites différents, dans le cas d'une catastrophe dans un site, les mêmes données qui existent dans ce site, existent dans un autre site sauvegarde [12].

Informatique verte (Green Computing) : Les émissions dangereuses dues à l'utilisation intensive des systèmes dans les organisations, les déchets électroniques générés au fil du temps et la consommation d'énergie sont le principal inconvénient des systèmes informatiques actuels. Cela peut être réduit dans une certaine mesure en utilisant des services de cloud computing. Cela conduit à la préservation de l'environnement. Les déchets électroniques sont également générés dans une mesure minimale [12].

7.2 Inconvénients du cloud computing

D'un autre côté, l'augmentation de l'utilisation des services cloud fait apparaître des inconvénients dans l'environnement du cloud. Parmi les inconvénients on trouve :

- Le problème de sécurité : le problème fondamental reste la sécurisation de l'accès à l'application entre le client et le serveur distant.
- Le problème de confidentialité : les données sont hébergées en dehors de l'entreprise. Le fournisseur proposant le service héberge les données de l'entreprise utilisatrice. Cela peut donc poser un risque potentiel pour l'entreprise de voir ses données mal utilisées ou volées.

Il s'agit donc de s'assurer que le fournisseur dispose d'une sécurité suffisante et qu'il propose une politique de confidentialité concernant les données de l'utilisateur.

- Le problème de connectivité et accès libre : Les services nécessaires à notre activité dépendent de la connexion internet. Une panne peut perturber notre organisation.
- Fiabilité et interopérabilité du cloud : Signifie que même si de nombreux composants du cloud échouent, le cloud est toujours disponible. Lorsque le cloud sera plus résilient et adaptable aux conditions changeantes, il sera plus fiable.

8 Qualité de service dans un environnement cloud

La qualité de service QoS (Quality of Service) est un concept de gestion qui a pour but d'optimiser les ressources et de garantir des performances convenables aux applications. De plus, la qualité de service désigne la capacité d'un service à répondre par ses caractéristiques aux différentes exigences de ses utilisateurs [13]. La QoS représente un ensemble d'attributs non fonctionnels des services, elle comprend la qualité de service côté serveur (par exemple, prix, popularité) et la qualité de service côté client (par exemple, disponibilité, débit, temps de réponse, taux d'échec et sécurité). Elle est utilisée comme critère de sélection et recommandation du service cloud dans plusieurs approches de recherche.

8.1 Caractéristiques

La qualité de service permet de véhiculer dans des bonnes conditions un type de trafic donné, en termes de plusieurs paramètres. Ces paramètres QoS peuvent être associés à un des services cloud par exemple :

Latence : caractérise le retard entre l'émission et la réception d'un paquet dans le réseau.

La gigue est définie comme la variation de ce délai [9].

Bande passante : définit le volume d'information (bits) par unité de temps (seconde) pour une application.

Taux de perte de paquets : correspond à la non-délivrance d'un paquet de données. Il est calculé comme étant le pourcentage des paquets échoués par rapport aux paquets envoyés.

Temps de réponse : il s'agit de l'intervalle de temps entre l'arrivée d'une requête CSU à une machine virtuelle dans le centre de données DC (Data Center) du CSP et l'envoi d'une réponse à cette requête depuis cette machine virtuelle.

Disponibilité : Décrit combien de temps le service est disponible dans une période de temps

spécifique par rapport à la durée totale d'ouverture du service.

Commentaires (feedback) : Indicateurs qualitatifs utilisés pour calculer la QoS fournie. Les avis clients incluent une série de rapports sur la qualité des interactions entre les clients et les fournisseurs de services cloud.

9 Conclusion

Dans ce chapitre, nous avons présenté les généralités du cloud computing pour bien comprendre ce domaine et pour avoir une idée globale sur l'environnement du cloud. Le but est de voir l'importance de cet environnement qui devient très large et très utile pour les utilisateurs et les entreprises.

Les services offerts par le cloud donnent plusieurs avantages qui attirent les clients et les entreprises. L'utilisation des services cloud a beaucoup augmenté, ce qui pose plusieurs problèmes et exigences. Les fournisseurs de services cloud cherchent à améliorer leurs services pour satisfaire les besoins de leurs clients, l'évolution des services du cloud est obligatoire pour répondre à ces besoins.

Avec la prolifération de ces services cloud, une recommandation (ou sélection) à partir de différentes plates-formes Cloud devient nécessaire.

Le chapitre suivant sera dédié à l'étude de l'état de l'art sur les différentes approches de recommandation des services cloud existantes. Nous allons porter un intérêt particulier à l'intégration de l'aspect social qui est le noyau de notre problématique.

CHAPITRE II : LA RECOMMANDATION DES SERVICES CLOUD

1 Introduction

Dans le cloud computing, la connexion entre les services cloud et les utilisateurs est établie par internet, où les clients utilisent un terminal pour exécuter les fonctionnalités offertes par les fournisseurs cloud en tant que services.

Avec la croissance rapide de la plateforme de services cloud, il est de plus en plus difficile pour les utilisateurs de découvrir des services cloud répondant à leurs demandes et satisfaisant leurs besoins. Les services cloud sont atomiques et chaque service assure une fonctionnalité. Un utilisateur peut rechercher un service cloud selon ses besoins mais les résultats de recherche peuvent donner un service qui n'est pas capable de satisfaire les besoins de cet utilisateur. Ou le nombre de services répondant aux critères de recherche sera très grand, ce qui peut mettre l'utilisateur dans l'embarras du choix. Ceci est dû au fait que le nombre de services répondant aux critères utilisés (généralement répondant aux éléments de la qualité de service) est énorme. Dans ce cas, un système de recommandation de services cloud est préconisé pour proposer à l'utilisateur le résultat le plus performant et le plus attendu possible en se basant sur d'autres critères.

2 Système de recommandation basé sur le cloud

L'expansion rapide et continue d'Internet et la disponibilité facile de nombreux services de commerce électronique et de réseaux sociaux, ont entraîné un volume considérable de données collectées quotidiennement par les fournisseurs de services. L'accumulation continue de ce volume massif de données a déplacé l'attention de la communauté de recherche sur le problème de recherche d'informations de base vers le filtrage des informations pertinentes, les rendant ainsi plus pertinentes et personnalisées en fonction de la requête de l'utilisateur. Par conséquent, la plupart des recherches visent à développer un système de recherche d'informations plus intelligent et plus autonome, à savoir un système de recommandation.

2.1 Définition du Système de Recommandation

Les systèmes de recommandation (SR) sont une sous-classe de systèmes de filtrage d'informations qui cherchent à prédire la « note (rating) » ou la « préférence » qu'un utilisateur donnerait à un article. Le système de recommandation est intégré aux applications pour fournir des

recommandations aux utilisateurs en leur fournissant des options suffisantes en fonction de leurs intérêts et de leurs besoins et en les aidant à prendre des décisions [34].

3 Techniques utilisées dans le Système de Recommandation

Fondamentalement, trois approches principales sont utilisées dans les systèmes de recommandation :

3.1 Approche de filtrage collaboratif

Le filtrage collaboratif (FC) est une technique de recommandation populaire et largement utilisé. Il est basé sur ses prédictions et recommandation en fonction des évaluations ou du comportement des autres utilisateurs du système. L'hypothèse principale derrière cette méthode est que les opinions des autres utilisateurs peuvent être collectées et minimisées de manière à fournir une prédiction raisonnablement précise de la préférence de l'utilisateur actif. Intuitivement, il est basé sur l'hypothèse que, si les utilisateurs s'accordent sur la qualité ou la pertinence de certains éléments, ils seront probablement d'accord sur d'autres éléments [14].

Avantages :

- La qualité des produits peut être jugée par les évaluations des utilisateurs.

Inconvénients :

- Problèmes de démarrage à froid de différents utilisateurs et de nouveaux produits.

La figure suivante présente le principe de fonctionnement de la méthode de filtrage collaboratif.

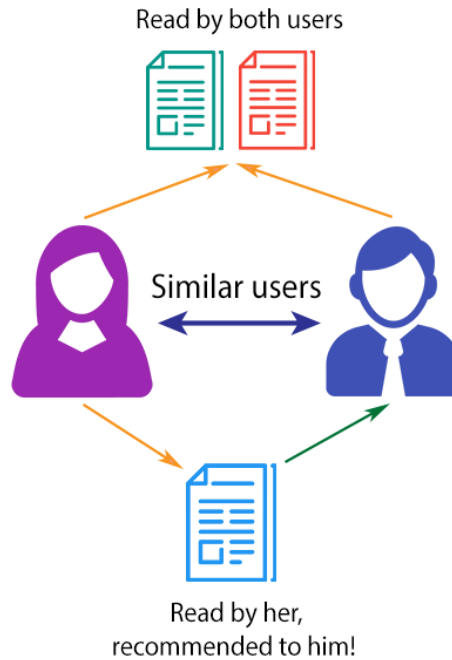


Figure 6: Filtrage collaboratif.

3.2 Approche de filtrage basée sur le contenu (content-based)

Le système de recommandation basé sur le contenu produit des recommandations à l'utilisateur en fonction des mots-clés saisis par l'utilisateur. Il trouve la correspondance des mots-clés et produit les recommandations. L'approche basée sur le contenu crée un modèle ou un profil de l'utilisateur sur la base des recherches et des évaluations effectuées précédemment par l'utilisateur actuel. Le profil ou modèle généré est une donnée structurée comprenant l'intérêt de l'utilisateur qui est adopté pour générer des nouvelles recommandations. Cette approche aide à générer des recommandations appropriées en filtrant les correspondances avec le profil et à décider de ce qu'il faut recommander à l'utilisateur et également très utile pour décider de ce qu'il ne faut pas recommander à l'utilisateur dans le scénario négatif [14].

Avantages :

- Aucune autre donnée utilisateur n'est requise.
- Pas de rareté des données ni de démarrage à froid.

Inconvénients :

- Une analyse de contenu est nécessaire pour déterminer les attributs de l'article.

- Il est impossible d'évaluer la qualité du produit.

La figure suivante présente le principe de fonctionnement de la méthode de filtrage basée sur le contenu.

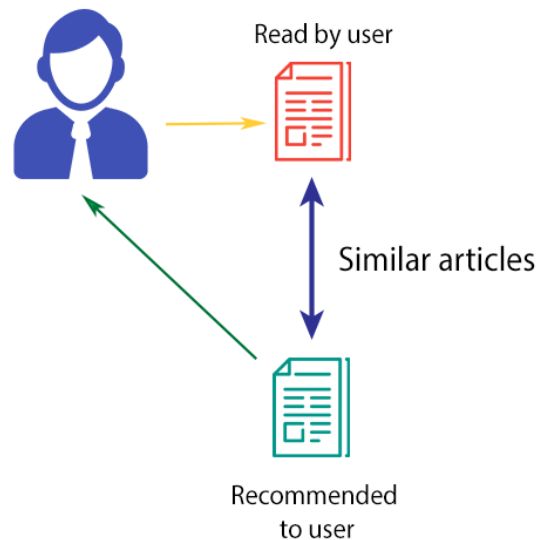


Figure 7: Filtrage basé sur le contenu.

3.3 Approche hybride

Les systèmes de recommandation actuels utilisent l'approche hybride qui n'est rien d'autre que la combinaison de deux approches, à savoir le filtrage basé sur le contenu et le filtrage collaboratif, pour surmonter les inconvénients des systèmes de recommandation individuels [14].

Les deux approches peuvent être combinées de l'une des quatre manières suivantes :

- 1) En mettant en œuvre à la fois l'approche de filtrage basée sur le contenu et celle basée sur la collaboration séparément et en combinant leurs prédictions pour produire les recommandations.
- 2) Incorporer certaines des caractéristiques du filtrage basé sur le contenu dans l'approche de filtrage collaboratif.
- 3) Incorporer certaines des caractéristiques du filtrage collaboratif dans une approche basée sur le contenu.

- 4) Incorporer les caractéristiques de l'approche de filtrage basé sur le contenu et de filtrage collaboratif en construisant le modèle unificateur général.

4 Défis et limites du Système de Recommandation

4.1 Rareté des données (Data Sparseness)

Le système de recommandation recommande généralement en créant des voisins pour l'utilisateur qui partage des intérêts similaires pour un élément particulier. Ici, la recommandation est basée sur l'hypothèse que si l'utilisateur A et l'utilisateur B partagent un intérêt similaire pour l'élément « x », alors il y a une probabilité que les deux partagent l'intérêt similaire pour l'élément « Y ». Mais, généralement, les utilisateurs évaluent très peu d'articles, il est donc très difficile de déterminer ses références et ses besoins, ce qui peut conduire à la création d'un mauvais voisin et des recommandations appropriées ne seront pas générées. Ainsi, la rareté est le problème qui se pose en raison du manque d'informations sur les éléments que l'utilisateur peut avoir notés.

4.2 Évolutivité (Scalability)

Le nombre d'utilisateurs utilisant le système et les articles entrant dans le système augmente de jour en jour. La plupart des systèmes de recommandation traditionnels existants souffrent de ce problème d'évolutivité. C'est un défi majeur pour le système de recommandation de fournir des recommandations en temps réel là où des calculs complexes doivent être effectués en analysant cet énorme volume de données en très moins de temps pour fournir des recommandations appropriées. Certains des systèmes de recommandation utilisent les techniques d'apprentissage automatique (Machine Learning) et d'exploration de données (Data Mining) pour générer un sous-ensemble de données. Parce qu'il existe un lien direct entre le jeu de données réduit généré et la qualité des recommandations générées.

4.3 Démarrage à froid (Cold Start)

Le problème typique qui existe dans le système de recommandation est le problème de démarrage à froid. Nous savons que le travail du système de recommandation est de recommander à l'utilisateur en fonction de son historique antérieur ou des recherches effectuées par l'utilisateur. Mais, ce que le système devrait recommander si l'utilisateur est nouveau dans l'application puisqu'il y a peu d'informations disponibles sur l'utilisateur pour lui recommander quelque chose. Cette situation est appelée problème de « démarrage à froid ». Ce problème de démarrage à froid n'est

pas seulement limité à l'utilisateur, il concerne également les éléments (produits, article, service ...). Lorsque l'élément est nouveau dans le système et n'a pas été évalué auparavant, il serait difficile de recommander l'article.

4.4 Confidentialité

Avec l'augmentation de la culture d'achat en ligne, le problème de la confidentialité augmente également, car il inclut nos informations personnelles telles que les détails de la carte de crédit, les données démographiques et les données de localisation d'un utilisateur particulier. Par conséquent, la confidentialité joue un rôle majeur dans la protection de ses données sensibles. Et l'aspect de la confidentialité est également lié à d'autres caractéristiques majeures telles que la fiabilité, la sécurité et la confidentialité des informations fournies.

4.5 Confiance

Dans ce problème de confiance, les évaluations jouent un rôle majeur. Différents peuples ont une histoire différente. Certains sont très actifs et certains ne sont pas aussi actifs dans l'évaluation, les commentaires et les avis sur un élément particulier. On ne peut pas conclure que la personne très active donne toujours une évaluation correcte. Par conséquent, le facteur de confiance joue un rôle essentiel dans l'évaluation de la personne appropriée pour extraire les informations afin de fournir les recommandations.

5 Travaux connexes

Pendant les dernières années et avec la croissance rapide des services cloud, il y a de plus en plus de services répondant aux critères liés à la qualité de service (QoS), le principal élément de sélection utilisé. Les utilisateurs trouvent des difficultés pour découvrir des services cloud répondant à leurs demandes et satisfaisant leurs besoins. L'intégration de nouveaux critères de sélection s'impose. Ceci a reçu beaucoup d'attention de la part de nombreux chercheurs. Le but est de chercher une solution optimale pour satisfaire les besoins des utilisateurs des services cloud et répondre à leurs demandes.

Il existe plusieurs approches proposées dans la littérature. Parmi celles-ci nous citons et discutons ici quelques approches :

Yuanchun Jiang et al. [15], ont proposé un modèle en deux étapes pour la recommandation des services cloud en intégrant les informations de description de service, textes descriptifs et étiquettes de service (Tags) [15].

Dans la première étape, ils ont proposé un modèle HDP (Hierarchical Dirichlet Processes) pour regrouper les services cloud en un nombre optimal de clusters. Ce modèle est basé sur les textes descriptifs [15].

Dans la deuxième étape, ils ont proposé d'appliquer l'algorithme PageRank personnalisé basé sur des tags de services pour recommander et classer les services cloud dans chaque cluster [15].

Des expériences sur un ensemble réel de données ont montré que la segmentation des services cloud en clusters dans les deux étapes proposées du modèle était bonne et obtenait des résultats de recommandation précis par rapport aux méthodes traditionnelles [15].

La figure 8 illustre l'organigramme du modèle en deux étapes proposées.

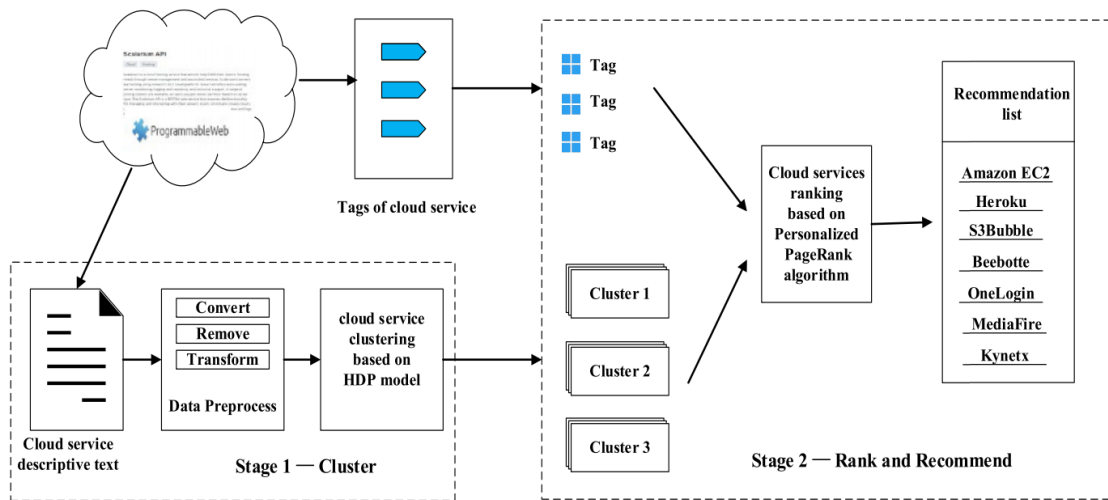


Figure 8: L'organigramme du modèle en deux étapes proposées [15].

Une méthode de sélection de services cloud basée sur la confiance et le clustering des préférences des utilisateurs a été proposée par Yubiao Wang et al. [16], une évaluation de confiance complète a été effectuée, qui est utilisée pour évaluer et sélectionner des services cloud.

Pendant ce temps, ils proposent une méthode de clustering hiérarchique condensée améliorée basée sur la similarité et les préférences de l'utilisateur pour améliorer encore

l'exactitude de la confiance des recommandations. Premièrement, une approche basée sur le modèle cloud est utilisée pour calculer les similitudes entre les utilisateurs, les paramètres de fonctionnalité du modèle cloud peuvent représenter les caractéristiques globales de chaque utilisateur, il est donc plus précis pour calculer la similarité de préférence d'un utilisateur [16].

Deuxièmement, l'algorithme de clustering hiérarchique condensé amélioré est utilisé pour regrouper les utilisateurs en fonction de la similarité des utilisateurs et pour former le domaine de préférence des utilisateurs. Grâce au domaine de préférence optimal de l'utilisateur, ils peuvent obtenir le degré de confiance de recommandation efficace. La confiance de recommandation comprend l'approbation de recommandation intra-domaine et l'approbation de recommandation extra-domaine. La confiance globale des services cloud qui consiste en une confiance directe et une confiance recommandée. La précision de la confiance des services cloud est calculée en temps réel. C'est une stratégie efficace qui permet aux utilisateurs de sélectionner les services cloud.

Les résultats expérimentaux montrent que la méthode de sélection de service cloud améliore le taux de réussite des transactions et permet aux utilisateurs de sélectionner les services cloud satisfaisants [16].

La figure 9 illustre l'organigramme de sélection de service cloud proposé.

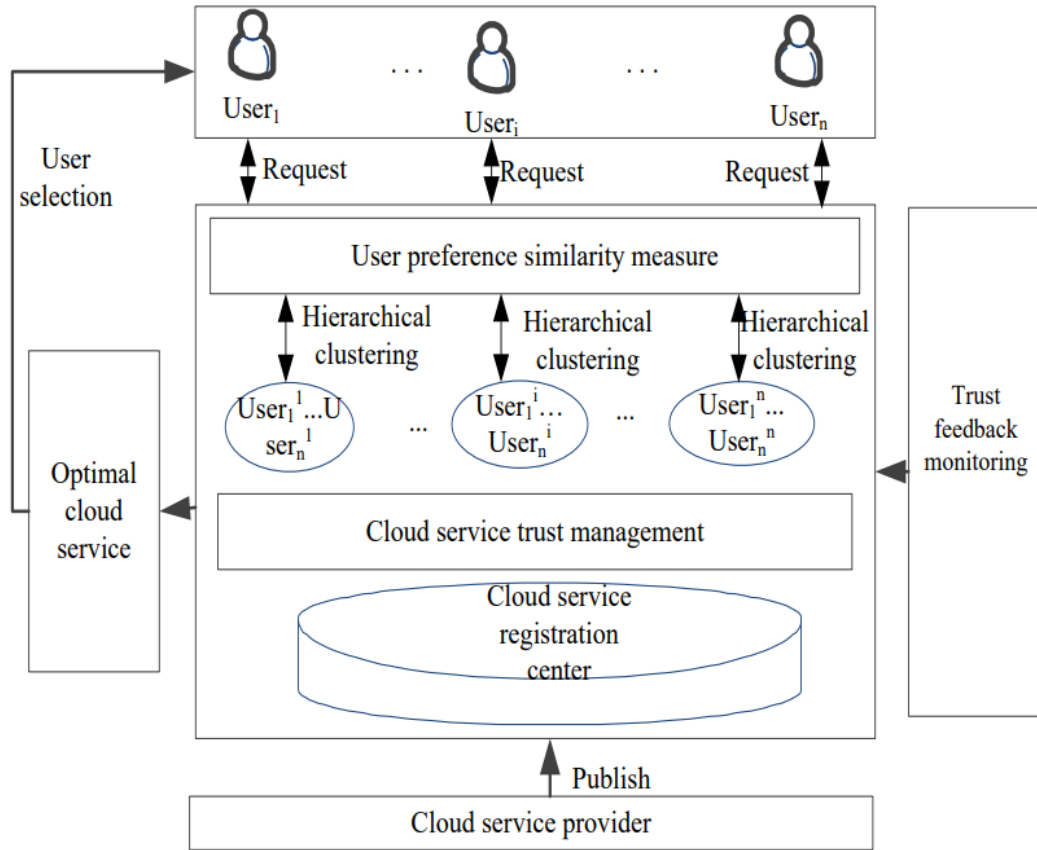


Figure 9: L'organigramme de sélection de service cloud proposé par [16].

Le travail proposé dans [17] présente une solution pour le problème de recommandation des services cloud, c'est un système de recommandation basé sur le filtrage collaboratif pour les services cloud à l'aide de Fuzzy-FCA (Fuzzy Formal Concept Analysis). Fuzzy-FCA est basé sur la théorie de réseaux, son principe est de représenter et visualiser des données en utilisant la représentation en treillis pour générer des recommandations fiables. Cette méthode permet une transformation référentielle de services cloud en un ensemble de petits clusters, où les relations entre les services de haute qualité et les utilisateurs ayant les meilleures expériences sont organisées à l'aide de FCA. La représentation en treillis permet d'extraire les informations pertinentes (utilisateurs similaires à un utilisateur actif, les top services ...etc.). C'est en cachant les services cloud mal notés et éliminer les utilisateurs moins similaires qui partagent peu de services mutuels avec l'utilisateur actif. Cela limite le processus de recommandation aux services les mieux notés et les utilisateurs

les plus similaires. Cette approche est efficace pour exploiter les préférences des utilisateurs et garantir des recommandations précises [17].

Shuai Ding et al [18], proposent une approche de recommandation de services cloud en fonction de temps TASR (Time-Aware Service Recommendation). Cette approche est sensible au temps et intègre la prédiction de la QoS basé sur CF et l'analyse des séries chronologiques. Les auteurs ont d'abord développé une approche de filtrage collaboratif (CF) améliorée par la similarité des utilisateurs pour capturer les caractéristiques temporelles de la similarité des utilisateurs et remédier à la rareté de données dans les PIT (Point In Time) existants. TASR a amélioré d'abord l'analyse de similarité en intégrant la similarité globale de l'utilisateur et la similarité d'invocation. TASR adopte un temps de similarité d'utilisateur pour décrire la nature dynamique de cette similarité. Ensuite TASR remplit les valeurs de QoS manquantes dans les PIT précédents et adopte le modèle ARIMA pour produire une meilleure recommandation pour les PIT suivants. Dans cet article les auteurs ont fait l'effort pour exploiter les données QoS structurés, les résultats expérimentaux montrent que TASR atteint des améliorations significatives des performances par rapport aux autres approches existantes [18].

Dans [19], les auteurs proposent un système de recommandation de services cloud basé sur la QoS. Pour cela ils proposent une approche de filtrage collaboratif en utilisant le coefficient de Spearman pour recommander les services cloud. Cette approche est utilisée pour prédire à la fois les évaluations QoS et les classements des services cloud (QoS ratings & rankings). Les systèmes de recommandation utilisés dans ce travail aident les fournisseurs cloud à promouvoir leurs services et les consommateurs du cloud à identifier les services qui répondent à leurs exigences de la qualité de services. D'après ces systèmes de recommandation, le filtrage collaboratif est la technique la plus efficace pour construire des systèmes de recommandation. Pour démontrer l'efficacité de l'approche, les auteurs effectuent des simulations approfondies et comparent l'approche avec une approche CF basée sur la notation utilisant le coefficient de Pearson. Les résultats expérimentaux montrent que l'approche CF utilisant le coefficient de Spearman permet d'obtenir des classements plus fiables, mais des notations moins précises que l'approche CF utilisant le coefficient de Pearson [19].

Une approche de filtrage hybride sensible au temps est proposée dans [20] par Meng et al, qui est une approche pour la recommandation des services cloud, qui intègre un modèle de voisinage (neighborhood) avec le modèle LFM (Latent Factor Model). Parce que l'environnement cloud est dynamique et incertain, donc il rend la qualité du cloud service sensible au temps. Cependant, la plupart des systèmes de recommandation existant n'ont pas tenu en compte l'influence temporelle. Face à ce défi, les auteurs proposent une méthode de recommandation collaborative hybride sensible au temps pour les services cloud. Au début, les dynamiques temporelles sont prises en compte dans le modèle de filtrage collaboratif basé sur le voisinage en distinguant les métriques QoS temporelles à partir de métriques QoS stables. Puis un modèle LFM temporel basé sur la décomposition CP (CANDECOMP/PARAFAC) est présenté et combiné pour améliorer les performances de la recommandation. Enfin des expériences basées sur un ensemble de données de service réels sont menées pour démontrer l'efficacité de cette proposition [20].

Dans [21], Afify et al proposent un système de recommandation personnalisé basé sur la réputation QoS pour les services SaaS (software-as-a-service), son principe est la demande basée sur les mots clés spécifiés par l'utilisateur dans un ensemble d'offres de services cloud. Les recommandations passent par quatre étapes :

En premier, correspondance fonctionnelle basée sur les opérations de service, suivis par la correspondance non fonctionnelle basée sur la qualité de service et les caractéristiques du service, filtrage hybride basé sur la réputation et les métadonnées et enfin, la personnalisation en fonction des intérêts des utilisateurs. La liste des services cloud recommandés est classée selon des critères uniques prenant en considération les préférences de l'utilisateur. Les défis abordés dans SaaSREC (SaaS-Recommend) incluent : le problème de démarrage à froid, une performance des recommandations de réputation basées sur les commentaires est faible, la variance de la qualité de service en fonction des localisations des personnalisations des utilisateurs est négligeable. L'approche de recommandation proposée surpasse les autres approches de recommandation basées sur le filtrage collaboratif en ce qui concerne la précision, le démarrage à froid et le taux d'erreur [21].

Dans [22] présente un filtrage collaboratif combiné, Kumar et Swati utilisent une librairie Mahout sur environnement Hadoop pour la recommandation des films. En raison de la

grande taille de données, le système de recommandation souffre du problème d'évolutivité, pour cela les auteurs ont utilisé Hadoop pour résoudre ce problème, aussi ils ont utilisé Mahout qui est une bibliothèque java source qui favorise le filtrage collaboratif sur l'environnement Hadoop pour gérer Big Data d'une façon statique [22].

L'algorithme proposé dans ce travail est divisé en deux parties. Dans la première phase, les auteurs appliquent le CF basé sur l'utilisateur et le CF basé sur l'article pour obtenir les résultats de recommandation. Dans la seconde phase, les auteurs combinent les résultats obtenus à partir de CF basé sur l'utilisateur et CF basé sur les éléments. Les résultats expérimentaux montrent que cette approche est efficace [22].

Shixing Yan et al [23] ont proposé une conception de haut niveau d'un Framework systématique pour les entreprises sur la recommandation et la sélection de services cloud dans une plate-forme de gestion de cloud hybride. Cette approche permet aux entreprises de recommander et sélectionner des services cloud basée sur l'évaluation selon leurs besoins et les spécifications du cloud offert sur la plateforme. L'élément clé du Framework est la mise en œuvre de la politique Moteur qui a été prototypé avec succès sur la base des techniques de vérification et de résolution de contraintes, et il fournit une détection automatisée des conflits et explications des conflits pour identifier les besoins problématiques des utilisateurs. L'intelligence de cloud dans le Framework exploite les techniques MCDM (Multiple Criteria Decision Making) pour générer des recommandations et des sélections de services cloud basées sur plusieurs critères, par exemple, des métriques quantitatives pour les services cloud, les préférences des utilisateurs, l'intelligence collective de l'intranet et les médias sociaux, et le système de notation [23].

Dans [24], les auteurs ont proposé un système de recommandation de services cloud sur le marché du cloud qui aide un utilisateur de sélectionner la meilleure combinaison de services de différents fournisseurs cloud qui correspondent à leurs exigences. Ce système de recommandation recommande les services cloud basés sur la QoS du réseau et les facteurs de plate-forme de machine virtuelle (VM) de différents fournisseurs de cloud. Les paramètres QoS comprend le temps d'exécution, le temps moyen d'exécution, le temps de réponse et le temps moyen de réponse. Service-rank (S Rank) considère la qualité des hyperviseurs de virtualisation utilisés par différentes plates-formes de services Cloud, les

commentaires des utilisateurs et le coût de services pour offrir de meilleurs services. Les résultats expérimentaux montrent que le système de recommandation proposé dans cet article peut recommander efficacement une bonne combinaison de services cloud pour les utilisateurs [24].

6 Étude comparative

Les différentes approches cherchent à améliorer les systèmes de recommandation des services cloud pour découvrir des services répondant aux besoins des utilisateurs. Le tableau suivant montre les avantages et les limites des approches étudiées :

Tableau 2: Comparaison entre les approches proposées.

Nom	Approche	Avantages	Inconvénients
Yuanchun Jiang et al. [15]	Recommandation de service cloud basée sur des informations textuelles non structurées	<ul style="list-style-type: none"> • Une bonne précision. 	X
Yubiao Wang et al. [16]	Une méthode de sélection de service cloud basée sur la confiance et la mise en cluster des préférences de l'utilisateur	<ul style="list-style-type: none"> • Une bonne précision. • La fiabilité de classement des services (confiance) 	<ul style="list-style-type: none"> • Une faible sécurité
Haithem Mezni et Taher Abdeljaoued [17]	Un système de recommandation de services cloud basé sur l'analyse de concept formelle floue	<ul style="list-style-type: none"> • Une bonne précision. • La fiabilité de classement des services (confiance) 	<ul style="list-style-type: none"> • Le temps de réponse élevé.

Shuai Ding et al. [18]	Recommandation de service cloud en fonction du temps à l'aide d'un filtrage collaboratif amélioré par similarité et du modèle ARIMA	<ul style="list-style-type: none"> • Une bonne précision. 	<ul style="list-style-type: none"> • Absence d'évaluation intégrée pour l'évaluation des services
X.Zheng et al. [19]	Recommandation de QoS dans les services cloud	<ul style="list-style-type: none"> • La fiabilité de classement des services (confiance) 	<ul style="list-style-type: none"> • Une faible précision.
Meng et al. [20]	Une méthode de recommandation collaborative hybride sensible au temps pour le service cloud	<ul style="list-style-type: none"> • Prise en compte de l'influence temporelle 	<ul style="list-style-type: none"> • Problème de démarrage à froid. • Problème de rareté des données (Data Sparseness)
Afify et al. [21]	Un système de recommandation personnalisé pour les services SaaS	<ul style="list-style-type: none"> • Une bonne précision. • Prise en compte le taux d'erreur. 	<ul style="list-style-type: none"> • Une faible confiance.
Kumar et Swati [22]	Personnalisation du système de recommandation à l'aide d'un algorithme de filtrage collaboratif sur le cloud à l'aide de Mahout	<ul style="list-style-type: none"> • Temps de réponse faible. • Une haute évolutivité. • Une bonne précision. 	<ul style="list-style-type: none"> • Problème de démarrage à froid. • N'a pas géré les données en temps réel.

Shixing Yan et al. [23]	Recommandation et sélection de services cloud pour les entreprises	<ul style="list-style-type: none"> • Un coût d'exploitation réduit. • Une haute sécurité. • QoS élevée. 	<ul style="list-style-type: none"> • L'évaluation de l'efficacité du système n'est pas prise en compte.
Seung-min Han et al. [24]	Système de recommandation de services efficace pour le marché du cloud computing	<ul style="list-style-type: none"> • Un temps de réponse réduit. • QoS élevée. • Haute évolutivité. 	<ul style="list-style-type: none"> • Une faible disponibilité ainsi que la confiance.

Dans ce qui suit, nous allons présenter une synthèse, sous forme d'un schéma de taxonomie, sur quelques études qui touchent la recommandation des services dans un environnement cloud en montrant les différents critères sur lesquels les auteurs se sont basés.

6.1 Les critères de comparaison des solutions

Le problème de la recommandation des services dans le cloud a été largement étudié, où les chercheurs se sont basés sur les techniques utilisées dans un système de recommandation, les chercheurs croient en l'obtention des meilleurs services répondant aux besoins des utilisateurs.

Les approches proposées sont classifiées en trois catégories, approches proposant des solutions à la recommandation des services cloud utilisant la technique du filtrage collaboratif, des approches utilisant la technique de filtrage basée sur le contenu et approches hybrides.

Les différentes approches proposées dans [16], [17], [18], [19] et [22] sont basées sur la recommandation des services dans le cloud utilisant la technique de filtrage collaboratif.

Dans [15], [20], [21] et [23] l'approche hybride a été utilisé dans le système de recommandation.

6.1.1 Classification selon les techniques utilisées dans le Système de Recommandation

Approche basée sur le filtrage collaboratif :

- Yubiao Wang et al. [16].
- Haithem Mezni et Taher Abdeljaoued [17].
- Shuai Ding et al. [18].
- X.Zheng et al. [19].
- Kumar et Swati [22].
- Ma et Hu [26].

Approche basée sur le contenu :

- Rodrigues, R. B. [25].

Approche hybride :

- Yuanchun Jiang et al. [15].
- Meng et al. [20].
- Afify et al. [21].
- Shixing Yan et al. [23].

La figure 10 représente les techniques utilisées dans les différentes approches proposées pour un système de recommandation. La classification illustre que le système de recommandation est classé en trois approches différentes et qu'il s'agit d'une approche basée sur le contenu, d'une approche basée sur le filtrage collaboratif (CF) et d'une approche hybride.

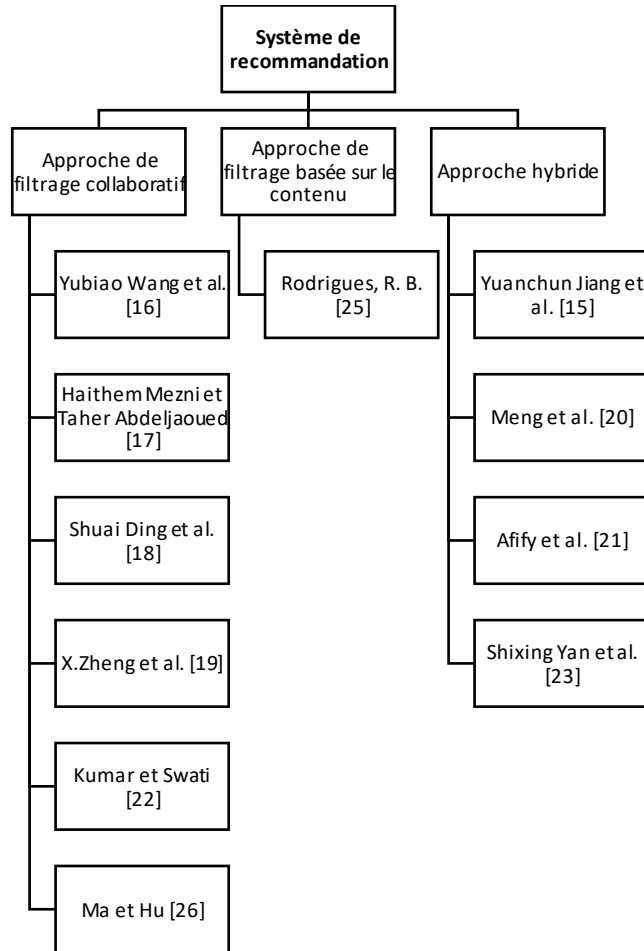


Figure 10: Classification des solutions étudiées sur les systèmes de recommandation.

Discussion

Comme nous l’avons mentionné dans la figure 10, plusieurs travaux dans ce contexte utilisent la technique du filtrage collaboratif pour la recommandation des services dans un environnement cloud, CF est considéré comme l’une des approches les plus répandues et les plus prometteuses dans les systèmes de recommandation.

L’objectif de l’utilisation du CF est d’augmenter la précision et la fiabilité de classement des services pour satisfaire la demande des utilisateurs.

La figure suivante montre le pourcentage des travaux qui traitent le problème de recommandation des services cloud selon les trois différents approches mentionnées.

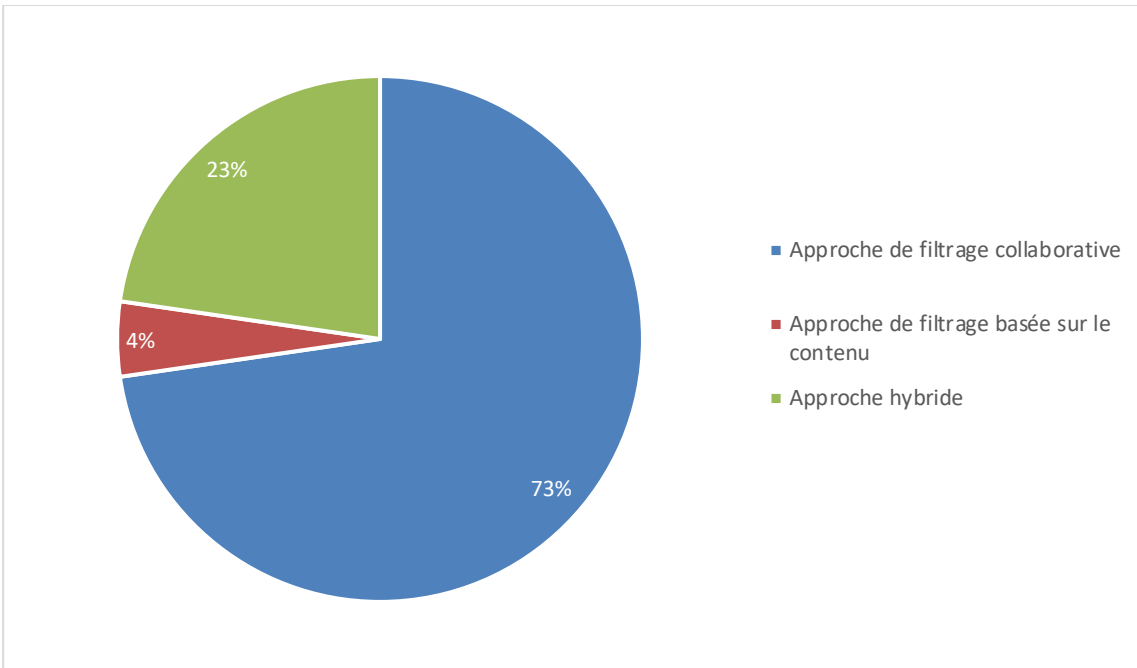


Figure 11: Traitement de la recommandation des services selon les techniques utilisées.

Ainsi que dans le tableau 3, la plupart des approches ont adopté un filtrage collaboratif. Il a été constaté que le filtrage collaboratif est la méthode la plus efficace pour résoudre le problème de recommandation car il prend en compte le comportement passé des voisins de l'utilisateur ciblé. Les utilisateurs actifs sauront si le service vaut la peine d'être testé.

6.1.2 Comparaison selon les critères utilisés dans le Système de Recommandation

Après avoir étudié plusieurs stratégies et méthodes différentes de la recommandation des services cloud, les chercheurs ont utilisé plusieurs critères (confiance, précision de la recommandation, QoS, la précision de la prédiction, sécurité, similarité, etc.) pour obtenir les meilleurs résultats adaptés aux besoins des utilisateurs du services cloud.

Les différentes approches proposées dans [18], [19], [21] et [26] sont basées sur les paramètres QoS le principal élément de sélection utilisé.

Dans [16] et [17], la confiance est un critère d'obtention de recommandations répondant aux besoins des utilisateurs.

D'autres normes sont utilisées dans diverses méthodes, comme illustré dans [15], où un texte descriptif et des étiquettes de service sont utilisés.

Le tableau 3 présente les différents critères utilisés pour recommander des services cloud, ainsi que le contexte et les techniques utilisées dans les approches examinées. Le contexte des systèmes de recommandation est général (recommande les services IaaS, PaaS et SaaS ensemble).

Tableau 3:Tableau comparatif des approches de recommandation de service Cloud.

Ref	Approche	Technique utilisée	Contexte	Critère de recommandation
[15]	Yuanchun Jiang et al. (2019)	Hybride	Général	Textes descriptifs, tags.
[16]	Yubiao Wang et al. (2019)	FC	Général	Confiance.
[17]	Mezni et Taher Abdeljaoued (2018)	FC	IAAS	Confiance.
[18]	Shuai Ding et al. (2018)	FC	Général	Le temps, Qos (débit et temps de réponse).
[19]	X.Zheng et al. (2017)	FC	Général	Qos (débit).
[20]	Meng et al. (2016)	Hybride	Général	Le temps.
[21]	Afify et al. (2016)	Hybride	SAAS	Réputation, Qos(précision et rappel).
[26]	Ma et Hu (2015)	FC	Général	Qos (confiance et coût).
[22]	Kumar et Swati (2015)	FC	Général	Accélération, efficacité.
[27]	Yu (2014)	FC	Général	RPPNMTF
[25]	Rodrigues, R. B. (2013)	Filtrage basée sur le contenu	Des dossiers	Facteurs de cloud.
[23]	Shixing Yan et al. (2012)	Hybride	Entreprises	Politiques de l'entreprise.

Discussion

Comme le montre le tableau 3, certains systèmes de recommandation fonctionnent à l'aide de normes de confiance et qualité de service (interaction, disponibilité, débit, délai, bande passante, perte de paquets, frais généraux, fiabilité, etc.), qui est les éléments de sélection les plus importants utilisés pour obtenir une très bonne précision de recommandation et une très bonne fiabilité de classement. Certains paramètres pourraient être pris en compte par certains chercheurs alors que simultanément pourraient être négligés par d'autres.

7 Apprentissage profond (Deep Learning) pour les systèmes de recommandation

L'apprentissage profond ou en profondeur (Deep Learning) est l'un des prochains grands développements de la technologie des systèmes de recommandation. Ces dernières années, les réseaux de neurones profonds ont connu un énorme succès dans un certain nombre de tâches complexes d'apprentissage automatique telles que la reconnaissance vocale et le traitement du langage naturel. Après son adoption relativement lente par la communauté des systèmes de recommandation, l'apprentissage en profondeur pour les systèmes de recommandation est devenu très populaire ces dernières années.

L'apprentissage en profondeur peut généralement être considéré comme un sous-domaine de l'apprentissage automatique (Machine Learning). C'est-à-dire que l'essence typique de l'apprentissage en profondeur est qu'il apprend des représentations profondes, c'est-à-dire qu'il apprend plusieurs niveaux de représentations et d'abstractions à partir de données. Pour des raisons pratiques, nous considérons toute architecture neuronale différentiable comme un « apprentissage en profondeur » tant qu'elle optimise une fonction objectif différentiable en utilisant une variante de stochastic gradient descent (SGD). Les architectures neuronales ont démontré un énorme succès dans les tâches d'apprentissage supervisées et non supervisées [51].

7.1 Techniques d'apprentissage profond pour les systèmes de recommandation

Plusieurs études ont intégré les techniques de l'apprentissage automatique en général et l'apprentissage profond en particulier, pour améliorer le processus de recommandation. Nous citons un certain nombre d'exemples.

7.1.1 Méthodes d'intégration (Embedding Methods)

Cette classe de méthodes utilise des techniques d'intégration inspirées des méthodes d'apprentissage en profondeur telles que word2vec pour intégrer des profils d'utilisateur, d'élément ou de contexte dans des espaces latents. Ces intégrations peuvent ensuite être utilisées soit directement pour fournir des recommandations, soit en tant qu'entrée dans d'autres méthodes (généralement supervisées) (apprentissage en profondeur) qui fournissent les recommandations. Alors que la factorisation matricielle peut également être considérée comme une technique d'intégration, les techniques d'intégration de type 2vec sont souvent un peu plus flexibles que les modèles de factorisation matricielle souvent utilisés dans le filtrage collaboratif [52].

7.1.2 Réseaux Feedforward et Auto-encoder pour le filtrage collaboratif

Ces méthodes utilisent des réseaux Feedforward ou Auto-encoder directement sur les interactions des utilisateurs afin de construire des modèles de filtrage collaboratif qui peuvent ensuite être utilisés pour des recommandations. Ces méthodes peuvent également être considérées comme une forme de méthodes de factorisation en profondeur [52].

- **L'Auto-Encodeur (AE)** est un modèle non supervisé qui tente de reconstruire ses données d'entrée dans la couche de sortie. La couche intermédiaire (couche d'étranglement) est généralement utilisée comme représentation des faits saillants des données d'entrée [51].

7.1.3 Méthodes d'extraction de Deep Feature

Cette classe de méthodes d'apprentissage en profondeur pour les recommandations se concentre sur l'utilisation de réseaux profonds pour effectuer l'extraction de caractéristiques sur les éléments. Ces fonctionnalités sont ensuite soit utilisées dans des méthodes de filtrage collaboratif plus standard, soit souvent l'extracteur de fonctionnalités qui fait partie d'une architecture plus profonde qui modélise également d'autres aspects des données [52].

7.1.4 Recommandation basée sur la session avec des réseaux de neurones récurrents

Ces méthodes sont basées sur le fait que souvent les interactions des utilisateurs avec le contenu se font dans des sessions, par exemple écouter de la musique, faire du shopping. De plus, on ne dispose souvent pas d'un identifiant utilisateur fiable pour modéliser les préférences de l'utilisateur. Dans ce cas, on peut recourir à des sessions de modèles en tant que données séquentielles et l'un des modèles les plus performants pour les données séquentielles sont les réseaux de neurones récurrents [52].

Discussion

L'apprentissage en profondeur fournit aux recommandeurs un ensemble de nouveaux outils pour l'extraction de caractéristiques et la modélisation des données d'utilisateurs, ce qui peut améliorer considérablement la qualité des recommandations fournies aux utilisateurs. Une partie de la puissance des techniques d'apprentissage en profondeur dans les systèmes de recommandation provient du fait que les méthodes d'apprentissage en profondeur permettent une meilleure extraction de caractéristiques à partir des éléments telles que l'image, la vidéo et l'audio par rapport aux techniques traditionnelles. Cela permet une modélisation plus précise des éléments en fonction de leur contenu.

8 Conclusion

Dans ce chapitre nous avons étudié le problème de recommandation des services cloud dans le cloud computing. Nous avons défini les techniques utilisées dans le système de recommandation et les différents défis et limites de ceux-ci. Nous avons aussi présenté des travaux étudiés dans ce contexte pour avoir une idée sur les différentes solutions qui existent et pour définir les objectifs relatifs au processus de recommandation.

Nous avons effectué une étude comparative des différentes approches proposées, tenant en compte plusieurs métriques, pour voir les avantages et les limites de chacune de ces approches. Nous avons également présenté un ensemble de techniques de l'apprentissage profond intégré pour améliorer le processus de recommandation.

Après cette fructueuse étude, nous sommes convaincus de la nécessité de l'intégration de nouveaux éléments liés notamment aux activités sociales des utilisateurs. Même si ces activités ont été considérées dans certaines études, ceci reste insuffisant du moment que ces travaux se base

surtout sur les éléments de la QoS. Nous considérons que les feedbacks des utilisateurs sont d'une grande importance puisqu'ils sont issus d'une manière totalement libre. Cet aspect doit être exploité dans un but d'améliorer le processus de recommandation. Chose que nous allons voir en détail dans le prochain chapitre.

CHAPITRE III : CONCEPTION D'UN MODELE DE SYSTEME DE RECOMMANDATION

1 Introduction

Dans le chapitre précédent nous avons étudié la recommandation des services cloud dans l'environnement du cloud computing, et nous avons comparé les différentes solutions proposées dans la littérature. Cette étude nous facilite la réalisation de notre solution qui va être présentée dans ce chapitre.

Nous allons présenter notre solution permettant d'intégrer l'aspect social dans la recommandation des services cloud, celui-ci est exprimé par les avis des utilisateurs (feedback, rating) pour chaque service cloud. De même, nous allons définir l'architecture de notre système de recommandation avec la spécification des différents besoins du système, et nous allons effectuer les processus nécessaires de la recommandation des services à partir des avis des utilisateurs jusqu'à l'exécution de la recommandation de service.

Nous allons donc dans ce chapitre présenter la phase conceptuelle de notre solution, nous commençons par présenter le principe général puis une architecture globale de notre système que nous allons détailler au fur et à mesure.

2 Principe général

Notre objectif principal est d'intégrer l'aspect social dans le processus de recommandation dans un but d'améliorer celui-ci. Nous considérons que le retour de l'utilisateur est une information très importante qui pourrait sans doute apporter un plus.

Notre modèle se compose de deux parties principales. Dans la première partie, nous avons utilisé Auto-Encoder pour l'extraction des caractéristiques et la classification de texte des commentaires.

Nous extrayons les attributs du commentaire en deux étapes : d'abord prétraiter les données, puis utiliser word2vec pour vectoriser le texte. Dans la deuxième étape, nous prenons l'ensemble de données divisé en trois parties, "Ensemble de données d'entraînement", "Ensemble de données de validation" et "Ensemble de données de test". Ensuite, nous utilisons Auto-Encoder pour la partie classification de texte, avec "Ensemble de données d'entraînement" pour construire notre modèle. Dans la deuxième partie nous recommandons les services cloud. Nous utilisons deux méthodes de recommandation, recommandation basée sur la popularité et le filtrage collaboratif.

3 Architecture globale du modèle

Après l'étude des travaux existants, nous pouvons maintenant définir les différentes tâches nécessaires de notre système de recommandation. Notre travail est composé de deux parties, nous allons dans ce qui suit détailler chacune d'elles.

- Partie 1 : Extraction de caractéristiques et classification basée sur Auto-encoder.
 - Étape 1 : Extraction des caractéristiques (Features Extraction).
 - Prétraitement (Preprocessing).
 - Vectorisation.
 - Étape 2 : Classification de texte.
- Partie 2 : Recommandation des services cloud.

Dans des travaux antérieurs, les chercheurs ont utilisé de nombreuses méthodes pour résoudre ce problème de recommandation de cloud. Cependant, selon nos recherches, aucun travail n'a utilisé Auto-encoder sur les commentaires des utilisateurs pour recommander des services cloud.

Dans la figure ci-dessous, nous montrons toutes les parties et étapes de notre modèle de recommandation de service cloud. Dans la première partie, nous utilisons un auto-encodeur pour l'extraction de caractéristiques et la classification de texte. La deuxième partie est dédiée à la partie recommandation où nous utilisons deux types de recommandation : la recommandation basée sur la popularité et le filtrage collaboratif basé sur l'utilisateur. Dans la section suivante, nous expliquerons chaque partie de ce modèle.

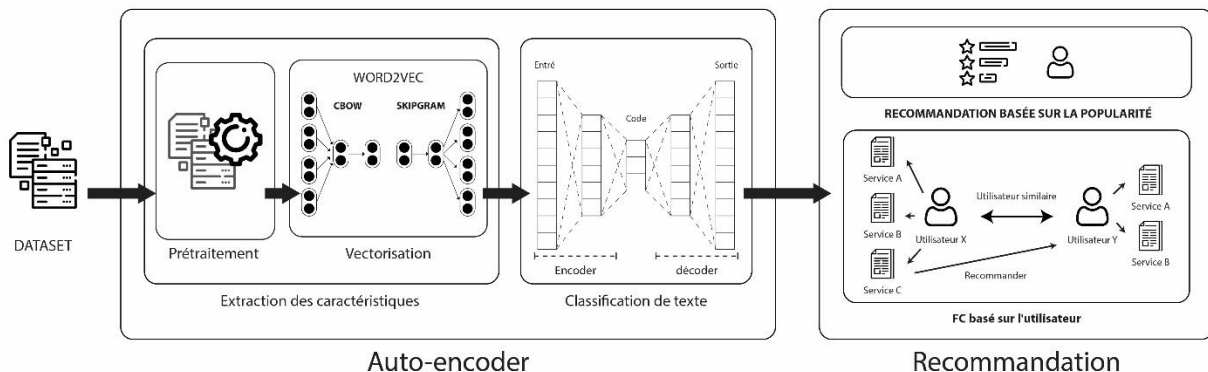


Figure 12: Architecture de notre modèle de système de recommandation.

4 Extraction de caractéristiques et classification basée sur Auto-encoder

Auto-encoder est un type de réseau de neurones artificiels, qui peut être utilisé pour apprendre une représentation compressée des données. La forme la plus simple d'un Auto-encoder comprend une entrée, une sortie et une ou plusieurs couches cachées qui leur sont connectées. Depuis, il n'utilise pas d'étiquettes, ce processus est donc un apprentissage non supervisé [50].

Nous avons choisi la technique Auto-encoder afin d'obtenir le meilleur résultat de prédiction de caractéristiques (commentaires positifs, commentaires négatifs) des commentaires des utilisateurs.

En tant que méthode d'apprentissage profond non supervisée, l'auto-encodeur a été largement utilisé dans divers domaines avec plusieurs façons et avec différents types de données pour ses excellentes performances en matière de réduction de la dimensionnalité des données, d'extraction de caractéristiques et de reconstruction de données. Pendant ce temps, des recherches récentes ont montré la grande efficacité de l'auto-encodeur dans les tâches de recherche d'informations et de recommandation.

Dans le domaine de la recommandation, les utilisations de l'auto-encodeur consistaient à utiliser des numéros comme données d'entrée (évaluation du service), mais nous avons vu que nous pouvons utiliser l'auto-encodeur avec des données textuelles en utilisant la technique Word2vec qui nous permet de convertir chaque commentaire en un vecteur de nombres entiers tandis que chaque mot sera représenté par un numéro dans ce vecteur. Cela nous permet de dire que l'application de cette méthode est une nouvelle solution qui n'a pas été utilisée auparavant par les chercheurs dans ce contexte.

4.1 Extraction des caractéristiques (Features Extraction)

Dans cette partie, nous souhaitons extraire les caractéristiques du commentaire liées au fait que celui-ci soit positif, c'est-à-dire en faveur du service commenté, ou négatif qui veut dire en défaveur du service commenté. Cette étape se présente en deux phases :

4.1.1 Prétraitement

L'objectif principal de cette étape est de réduire le texte, de le nettoyer en mots simples et de le normaliser pour en faciliter l'utilisation. Il peut représenter des mots comme des mots-clés que nous pouvons utiliser pour reconnaître du texte.

Le prétraitement des données convertit les données dans un format que les utilisateurs peuvent traiter plus facilement et plus efficacement, comme dans les réseaux de neurones [35]. Divers outils et techniques sont utilisés pour le prétraitement, notamment :

- **Echantillonnage (Sampling)** : sélection d'un sous-ensemble représentatif d'un grand ensemble de données [35].
- **Conversion (Transformation)** : Traitement des données brutes pour créer une seule entrée [35].
- **Débruitage (Denoising)** : Supprime le bruit des données [35].
- **Standardisation (Normalization)** : organiser les données pour améliorer l'efficacité de l'accès [35].

Les techniques que nous avons utilisées dans cette phase sont les suivantes :

- **Supprimer la ponctuation**

La ponctuation n'a aucun effet sur l'analyse du texte du commentaire, elle est donc supprimée, car elle ne fournit aucune information utile pour la classification. Elle comprend des symboles tels que "@" et "+".

- **Suppression des numéros**

La suppression de numéros a pour but de faciliter le traitement des mots du texte.

```
In [59]: text = "2Data preprocessing!!! Transforms000 the%% Data @Into */a format #that will be more4 Effectively"
```

After remove the digits And Remove Punctuation :

```
Data preprocessing Transforms the Data Into a format that will be more Effectively
```

Figure 13: Un exemple de suppression des chiffres et de la ponctuation.

- **Suppression des majuscules**

Nous convertissons toutes les lettres majuscules en lettres minuscules.

- **Tokenisation**

La tokenisation est une technique qui convertit le texte en données très simples stockées dans des listes nommées. La tokenisation peut être utilisée pour protéger les données sensibles en remplaçant les données d'origine par des valeurs distinctes de même longueur et de même format.

```
In [59]: text = "2Data preprocessing!!! Transforms000 the% Data @Into */a format #that will be more4 Effectively"

Tokenization :

['data', 'preprocessing', 'transforms', 'the', 'data', 'into', 'a', 'format', 'that', 'will', 'be', 'more', 'effectively']
```

Figure 14: Un exemple de tokenisation d'un texte.

- **Suppression des mots vides**

Les mots vides sont des mots qui apparaissent souvent dans la langue, ils ne fournissent pas d'informations utiles pour la classification. Au lieu de cela, ils affectent la classification du texte, ils sont généralement très courants et apparaissent dans la plupart des commentaires. Généralement, les mots vides sont les déterminants, les prépositions et les pronoms. Dans la langue anglaise on peut trouver des mots vides tels que « are », « the » et « and ».

```
In [59]: text = "2Data preprocessing!!! Transforms000 the%% Data @Into */a format #that will be more4 Effectively"
```

```
After remove the digits And Remove Punctuation :
```

```
After Normalize all characters to lowercase :
```

```
After Tokenization :
```

```
remove stop words :
```

```
['data', 'preprocessing', 'transforms', 'data', 'format', 'effectively']
```

Figure 15: Un exemple de suppression des mots vides (StopWords).

- **Lemmatisation**

Le but de la racinisation est de supprimer les inflexions et d'attribuer un mot à sa forme racine. Il transforme les mots en leur racine réelle. Par exemple, la réduction de "builds", "building", ou "built" au lemme "build". Il peut utiliser un dictionnaire tel que WordNet pour les mappages ou certaines approches spéciales. Dans notre cas, nous avons utilisé WordNet.

4.1.2 Vectorisation

C'est la deuxième étape importante de la partie extraction des caractéristiques. Son objectif est de transformer les mots en vecteurs pour terminer la partie d'extraction des caractéristiques et pour préparer l'entrée du modèle de similarité.

Dans cette étape, nous avons utilisé l'incorporation de mots (Word Embedding) comme méthode de vectorisation.

Incorporation de mots (Word Embedding)

C'est une méthode d'apprentissage d'une représentation de mots dans un document par des nombres réels, elle est notamment utilisée en traitement automatique du langage. De plus, il s'agit d'un type de représentation de mots qui permet aux algorithmes d'apprentissage automatique de comprendre des mots ayant des significations similaires. Cette technique permet à chaque mot de notre commentaire dans l'ensemble de données d'être représenté par un vecteur de nombres réels. Pour faire du Word Embedding, nous avons utilisé l'algorithme Word2Vec [37].

➤ **Word2Vec**

Word2vec est un algorithme d'inclusion de mots. C'est une technique de tri qui peut convertir le langage naturel en une représentation vectorielle distribuée. Il peut littéralement capturer le contexte dans un espace multidimensionnel et est généralement utilisé comme étape préliminaire d'un modèle de prédiction sémantique, et des tâches de collecte d'informations. Nous l'utilisons dans notre cas pour convertir tout le commentaire en vecteurs, et nous pouvons extraire la similitude entre les mots. Word2vec est basé sur un réseau de neurones à deux couches, ce dernier ayant deux architectures neuronales appelées CBOW et SkipGram [36].

- **Modèle CBOW (Continuous Bag of Words)**

Ce modèle est basé sur le contexte et prédit les mots cibles. Le résultat de la couche cachée est une nouvelle représentation de mot [37].

- **Modèle de Skip-Gram**

Le modèle est basé sur le mot cible, et prédit les mots du contexte. Le résultat de la couche cachée est la nouvelle représentation du mot [37].

La couche cachée

Dans les réseaux de neurones, une couche cachée est située entre l'entrée et la sortie de l'algorithme, dans laquelle la fonction applique des poids aux entrées et les dirige via une fonction d'activation en tant que sortie. En bref, les couches cachées effectuent des transformations non linéaires des entrées qui sont entrées dans le réseau [53].

La figure 16 décrit le processus Word2vec, qui implique deux composants distincts : Continuous Bag of Words (CBOW) et skip-gram. Le composant CBOW déduit le mot cible lorsqu'on lui donne les mots de contexte, tandis que le composant skip-gram déduit les mots de contexte lorsqu'on lui donne un mot d'entrée.

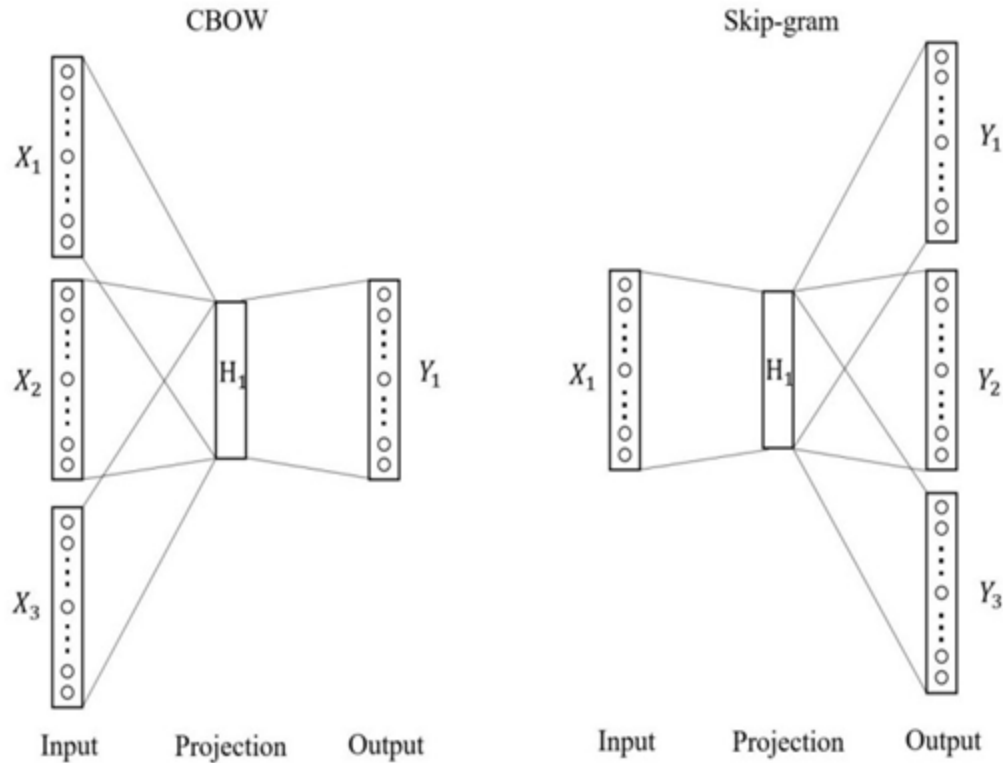


Figure 16: Modèle Word2vec (Continuous Bag of Words (CBOW) et Skip-gram [36].

4.2 Classification de texte

Dans cette section, nous allons présenter la méthode que nous avons utilisée pour faire le classement de texte des commentaires des utilisateurs. Nous avons utilisé un Auto-encodeur pour apprendre une représentation compressée des caractéristiques d'entrée pour un problème de modélisation prédictive de classification de texte.

Nous utiliserons une tâche de classification binaire (2 classes) à deux dimensions et nous prenons 260 mots (la taille maximale du commentaire) comme nombre de caractéristiques d'entrée pour les deux paires (2, 260), et nous prenons 409287 échantillons pour l'apprentissage. La première classe représente les commentaires négatifs et la deuxième représente les commentaires positifs.

Le modèle prendra l'entrée, puis produira les mêmes valeurs d'entrée. Il apprendra à recréer correctement le modèle d'entrée.

L'encodeur apprend à interpréter l'entrée et à la compresser pour créer une représentation compressée que nous appelons la couche cachée (latent layer). Le décodeur prend la sortie de l'encodeur (latent layer) et tente de recréer la même entrée. Avant d'adapter le modèle, nous diviserons notre ensemble de données en « entraînement », « validation » et « tests » et mettrons à l'échelle les données d'entrée en normalisant les valeurs dans la plage [0 : 1], ce qui est une bonne pratique MLP (Multilayer Perceptron).

La figure ci-dessous représente un schéma descriptif de la technique Auto-encoder que nous avons utilisé pour obtenir les caractéristiques des commentaires (positif ou négatif). La première étape est l'extraction de caractéristiques où nous avons prétraité notre dataset et utilisé la technique Word2vec. Dans la deuxième étape, nous avons fait la classification texte afin d'obtenir le meilleur résultat de prédiction de caractéristiques des commentaires des utilisateurs.

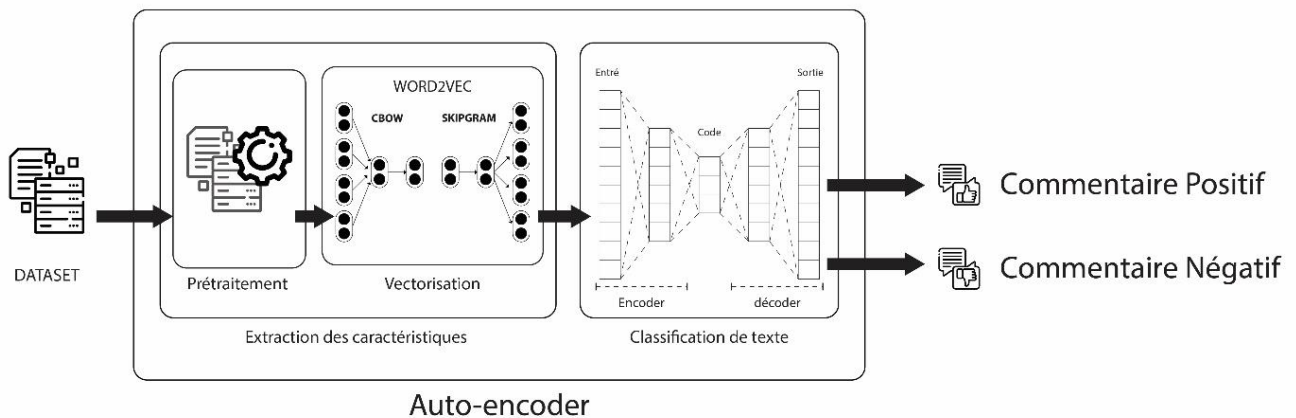


Figure 17: Un schéma descriptif de la technique Auto-encoder.

On va définir un encodeur avec 3 couches cachées, la première a le nombre d'entrées (2, 260), la seconde a moins d'entrées, et ainsi de suite. Enfin, la dernière couche cachée encodée nous fournit une couche cachée avec moins d'entrées que la couche précédente (10). Le décodeur sera défini avec une structure similaire, il récupérera la latente en entrée et extraira les caractéristiques pour obtenir les données de sortie, ce sont les mêmes données que nous l'avons entrée dans les couches encodées.

Fonctions d'activation

Les fonctions d'activation sont les parties principales des couches de réseaux de neurones artificiels, nous les utilisons en deux étapes pour les couches cachées et pour la couche de sortie. Dans les couches cachées pour contrôler la façon dont l'architecture du réseau apprend l'ensemble de données d'entraînement, et dans la couche de sortie pour définir le type de résultats de prédiction. Dans notre solution, nous avons utilisé deux fonctions d'activation, nous les définirons comme suit :

- **Sigmoid**

La fonction sigmoid est une activation utile pour diverses raisons. Cette fonction agit comme une fonction d'écrasement continue qui limite sa sortie dans la plage (0, 1). Il est également centré sur zéro, créant une limite de décision simple pour les tâches de classification binaire [54].

Il est défini mathématiquement comme suit :

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- **ReLU (Rectified Linear Unit)**

Cette fonction simple est devenue populaire car elle a montré une convergence plus rapide par rapport au sigmoid et au tanh (hyperbolic tangent), probablement en raison de son gradient non saturant dans le sens positif. En plus d'une convergence plus rapide, la fonction ReLU est beaucoup plus rapide en termes de calcul. Les fonctions sigmoid et tanh nécessitent des exponentielles qui prennent beaucoup plus de temps qu'une simple opération max [54].

L'avantage de l'utilisation de la fonction ReLU est que tous les neurones ne sont pas activés en même temps. Cela implique qu'un neurone ne sera désactivé que lorsque la sortie de la transformation linéaire est nulle. Il est défini mathématiquement comme suit :

$$f(x) = \max(0, x)$$

Le tableau suivant montre la taille des jetons que nous utilisons à chaque niveau ainsi que la fonction d'activation que nous utilisons.

Tableau 4: Toutes les couches avec les tailles de jetons et les fonctions d'activation.

Couche	Taille des jetons	Fonction d'activation
L'entrée	(2, 260)	X
Encodé	120	Relu
Encodé	10	Relu
Décodé	120	Relu
Décodé	260	Sigmoid

La couche de sortie aura le même nombre de nœuds que les données d'entrée et utilisera une fonction d'activation linéaire pour sortir des valeurs numériques comme la fonction d'activation RELU.

Après cela, le modèle sera ajusté en utilisant la version effective de la descente de gradient stochastique *mean_squared_error* pour minimiser l'erreur quadratique moyenne car le problème de régression en sortie est un type de régression multiple.

À la fin, nous pouvons entraîner le modèle pour reproduire plus facilement l'entrée. Après l'entraînement, nous pouvons tracer les courbes d'apprentissage pour l'entraînement et les tests afin d'évaluer si le modèle a bien appris le problème de reconstruction. Enfin, nous pouvons enregistrer le modèle Auto-encoder.

5 Recommandation des services cloud

Dans notre modèle, nous avons utilisé deux techniques de recommandation : recommandation basée sur la popularité et recommandation basée sur l'utilisateur.

5.1 Recommandation basée sur la popularité

Nous offrirons des recommandations généralisées à chaque utilisateur, basées sur la popularité du service en fonction des commentaires des utilisateurs. Notre système recommande les mêmes services aux utilisateurs ayant des caractéristiques similaires. L'idée de base de notre travail est que les services les plus populaires et les plus appréciés des utilisateurs seront plus susceptibles d'être appréciés du public.

Dans cette partie, après avoir calculé la valeur du score (Weighted_Rating) de chaque service et l'avoir trié, nous avons recommandé le top des services cloud à tous les utilisateurs.

$$Score (WR) = \left(\frac{V}{V + M} \cdot R \right) + \left(\frac{M}{V + M} \cdot C \right)$$

Où :

V : nombre de commentaires par service.

M : est le nombre minimum de commentaires requis pour être répertorié.

C : est la somme des commentaires positifs / nombre de service.

R : moyenne des commentaires positifs par service.

Nous avons utilisé WR pour que la recommandation de service soit assez juste, car un service avec une moyenne de commentaire positif élevé et petit nombre de commentaires ne peut pas être considéré comme meilleur qu'un service avec une moyenne de commentaire positif qui n'est pas élevé mais avec un très grand nombre de commentaires. Nous avons fait ce processus pour assurer l'intégrité de la classification pour chaque service, ce qui peut améliorer considérablement la qualité des recommandations fournies aux utilisateurs.

5.2 Filtrage collaboratif basé sur l'utilisateur

Dans cette section, nous utiliserons la technique de filtrage collaboratif pour faire des recommandations des services cloud.

Tout d'abord, nous choisissons les services qui ont la meilleure popularité. Après cela, nous choisissons les utilisateurs qui se ressemblent dans plusieurs commentaires positifs des mêmes services qu'ils ont utilisé.

De chaque utilisateur $u \in U = \{1, 2, \dots, m\}$ qui commente positivement à un service $s \in S = \{1, 2, \dots, n\}$ on obtient un tableau de deux dimension $T = (U_i, S_{ij})$. La similarité entre les utilisateurs sera calculée comme suit :

$$SIM = ([u_i][s_{i,j}] \cap [u_{i+1}][s_{i+1,j}])$$

Tel que $i \in m$ et $j \in n$

Après avoir extrait la similarité entre les utilisateurs, nous recommandons les services qui ne sont pas utilisés par l'un d'entre eux, cette recommandation des services s'applique au niveau de tous les utilisateurs similaires.

La figure ci-dessous représente un schéma descriptif de fonctionnement de la technique de filtrage collaboratif basé sur la similarité entre les utilisateurs en utilisant les commentaires positifs. Si un utilisateur X a utilisé et commenté positivement les services {A, B et C} et l'utilisateur Y a utilisé et commenté positivement les services {A, B} alors on peut dire que l'utilisateur X et Y sont similaires. Nous pouvons donc recommander le service C à l'utilisateur Y.

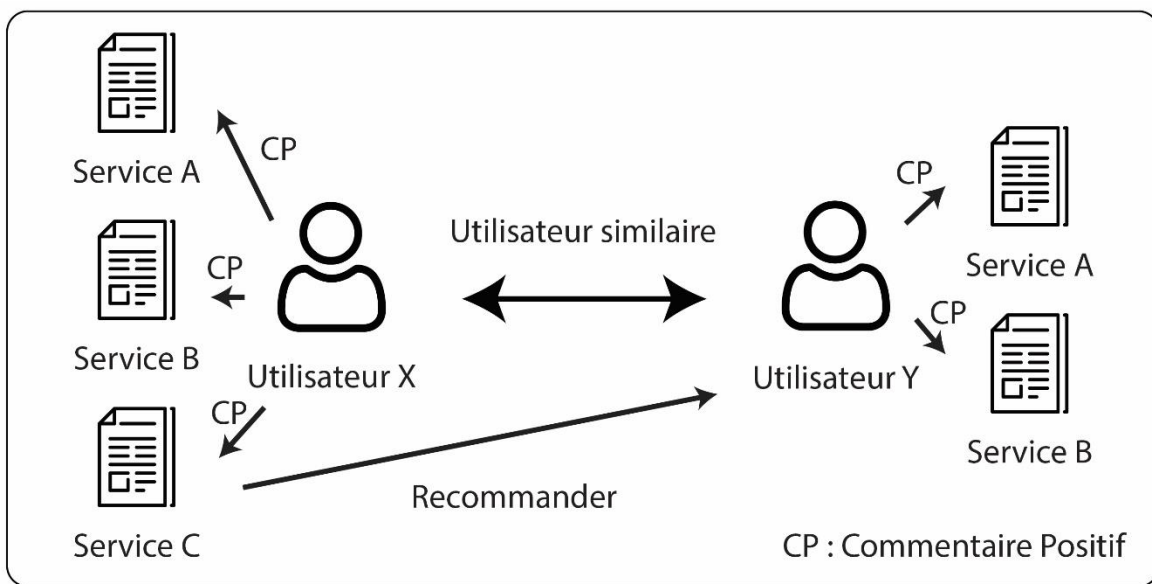


Figure 18: FC basé sur la similarité des utilisateurs.

6 Conclusion

Dans ce chapitre, nous avons présenté les différentes parties et étapes de notre nouveau modèle. Surtout, nous approfondissons dans chaque partie pour mieux comprendre les bases de toutes les méthodes utilisées afin que tout soit clair.

Dans le chapitre suivant, nous présenterons nos expérimentations réalisées dans l'objectif d'évaluer et de valider notre proposition. Nous décrirons tous les outils matériels (Hardware) et logiciels (Software) que nous avons utilisés pour construire notre modèle. De plus, nous présenterons les résultats que nous avons obtenus dans notre nouveau modèle.

CHAPITRE IV : EXPERIMENTATION ET RESULTATS

1 Introduction

Après l'étape de modélisation de notre approche, nous arrivons dans ce dernier chapitre à l'implémentation et la mise en œuvre de la solution proposée, le test, la récupération des résultats pour la comparaison, l'analyse et la discussion.

Pour réaliser nos recherches et construire notre modèle de recommandation des services cloud en intégrant l'aspect social, nous avons utilisé un ensemble d'outils de logiciels et de matériels.

2 Matériel informatique (Hardware)

Pour réaliser notre model, nous avons utilisé un ordinateur qui tourne avec un système d'exploitation Windows 10, alimenté par un processeur AMD Ryzen 5 3400G 3.70GHz x64 avec une carte graphique Radeon Vega 11, et une RAM de 16 GO.

3 Logiciel (Software)

Pour construire notre modèle, nous utilisons Jupyter, un environnement de travail simple et facile à utiliser.

De plus, nous avons utilisé Python 3.8 comme langage de programmation. Nous avons déduit des travaux antérieurs qu'il s'agit du langage le plus approprié pour de nombreuses raisons :

- C'est un langage interprété, ce qui signifie que Python exécute directement le code ligne par ligne.
- Un langage de programmation de haut niveau a une syntaxe de type anglais. Il est plus facile de lire et de comprendre le code.
- C'est un langage très large, il est utilisé pour tous les domaines.
- Il dispose de bibliothèques diverses et riches, celles dédiées au deep learning et celles utilisées pour la gestion d'autres structures de données et autres.
- Sa simplicité permet d'exprimer des équations et des formules complexes, c'est pourquoi il est facile, ce qui consomme moins de temps pour l'ensemble du processus de développement.

4 Les Bibliothèques

Dans cette partie, nous présenterons toutes les bibliothèques utilisées pour construire notre modèle.

TensorFlow : Il s'agit d'une plate-forme open source et d'une bibliothèque Python, créée par des chercheurs de l'équipe Google Brain pour le domaine des techniques et méthodes d'apprentissage automatique (Machine Learning) et d'apprentissage profond (Deep Learning). Il dispose d'une architecture flexible d'outils qui facilitent le déploiement sur diverses plates-formes telles que GPU (Graphics Processing Unit), CPU (Central Processing Unit) et TPU (Tensor Processing Unit) [38].

Keras : il s'agit d'une bibliothèque Python gratuite et open source, utilisée pour les modèles d'apprentissage en profondeur (Deep Learning) et d'apprentissage automatique (Machine Learning), et elle agit comme une API de la bibliothèque TensorFlow. Keras aide à exécuter des expériences facilement et plus rapidement et le même code peut être exécuté sur CPU ou sur GPU, de manière transparente [39].

Genism : Il s'agit d'une bibliothèque open source fournie par le langage Python, elle est utilisée pour les techniques d'apprentissage automatique et d'apprentissage profond, et pour les sujets non supervisés, elle a fourni plusieurs algorithmes comme l'algorithme Word2vec pour l'étape de vectorisation dans la partie extraction de caractéristiques [40].

Scikit-learn : est une bibliothèque Python gratuite et open source, créée par plusieurs chercheurs pour l'apprentissage automatique et l'apprentissage en profondeur, et elle est commercialement utilisable. Il permet le passage à des versions efficaces de nombreux algorithmes actuels [41].

NumPy : c'est l'une des nombreuses bibliothèques open source de python, c'est pour les tableaux numériques. Il a été créé spécialement pour le calcul scientifique, en particulier le calcul matriciel, tout en offrant de multiples fonctions permettant la création et la manipulation de matrices et de vecteurs [42].

5 Base de données (Dataset)

Dans cette section, nous discuterons de la base de données utilisée dans notre recherche.

Amazon Fine Food Reviews cet ensemble de données se compose de critiques alimentaires d'Amazon. Les données couvrent plus de 10 ans et incluent presque tous les 500 000 commentaires

jusqu'à octobre 2012. Les avis incluent des informations sur les produits et les utilisateurs, des avis et des commentaires textuels simples. Il comprend également des évaluations pour toutes les autres catégories Amazon [43].

L'ensemble de données contient 2 fichiers nommés "Reviews.csv" extraits de la table SQLite correspondante nommée Reviews dans "database.sqlite".

Les données contenues dans le fichier "Reviews.csv" représentent les avis d'octobre 1999 à octobre 2012, avec 568 454 commentaires, 256 059 utilisateurs, 74 258 produits et 260 utilisateurs ayant des avis > 50. La figure 19 montre la structure du fichier « Reviews.csv ».

# Id	ProductId	UserId	ProfileName	Score	Time	Summary	Text
1	B801E4KFG8	A3SGXH7AUHU8GW	delmartian	5	1303852400	Good Quality Dog Food	I have bought several of the Vitality canned dog food products and have found them all to be of good...
2	B80813GRC4	A1D87F6ZCVE5NK	dll pa	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanuts...the peanuts were actually small sized unsalted. No...
3	B808LQ0CH8	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	4	1219017600	"Delight" says it all	This is a confection that has been around a few centuries. It is a light, pillowy citrus gelatin wi...
4	B808UA0QIQ	A395E0RC6FGVXV	Karl	2	1307923200	Cough Medicine	If you are looking for the secret ingredient in Robitussin I believe I have found it. I got this in...

Figure 19: La structure du fichier Reviews.csv.

6 Implémentation

Dans cette section, nous allons commencer le travail principal pour construire notre modèle à l'aide de l'Auto-encoder. Après l'extraction des caractéristiques (prétraitement et vectorisation) du Dataset, les données sont divisées en deux catégories : les avis positifs, dans lesquels les utilisateurs notent le produit (4 et 5), et les commentaires négatifs où les utilisateurs ont évalué le produit (1, 2 et 3).

La taille de chaque commentaire est de 260 jetons toute leur valeur entre [0 ; 1]. Si la taille est inférieure à 260 jetons, il ajoute automatiquement 0 au texte pour le remplir.

Nous allons diviser les données en trois parties "entraînement", "validation" et "test". Pour l'entraînement, nous avons pris 80% des données, et pour les tests, nous avons pris 20% des données. De plus, pour la validation, nous avons pris 10% des données d'entraînement. Cela signifie que nous avons pris 72% pour l'entraînement et 8% pour la validation à partir des données principales.

De plus, cela nous aidera à réduire les risques de surapprentissage, car nous validerons notre modèle sur des données qu'il n'aurait pas vues lors de la phase d'entraînement.

Enfin, nous avons obtenu 454763 échantillons pour l'entraînement et 113691 pour les tests. De plus, nous avons pris 10% des tests pour la validation donc les dernières statistiques sont :

Entraînement : 409287 échantillons.

Validation : 45476 échantillons.

Test : 113691 échantillons.

7 Entraînement et validation

Une fois le modèle créé, nous devons le compiler en utilisant le nom de la fonction d'optimisation *RMSprop* (). De plus, nous devons spécifier le type de perte via l'argument *loss*, son nom est *mean_squared_error* puisque la perte après chaque lot sera calculée entre le lot de sortie prédit et le résultat idéal attendu en utilisant *mean_squared_error*.

Après cela, nous avons commencé à entraîner le modèle à l'aide de la fonction *fit* () Keras. Nous avons formé notre modèle pour 200 époques. La fonction *fit* () renvoie un objet historique, en stockant le résultat de cette fonction dans une variable, et nous pouvons l'utiliser plus tard pour tracer un diagramme de la fonction de perte entre l'entraînement et la validation, ce qui nous aidera à analyser visuellement les performances de notre modèle.

Enfin, nous sauvegardons notre modèle en utilisant cette fonction *save_weights()* pour l'utiliser dans une partie des tests.

8 Test

Dans ce cas, nous avons fait le test de prédiction et de classification des données de notre modèle. Après cela, nous montrons les prédictions que nous obtenons. La figure ci-dessous montre les résultats des prédictions.

```
Entrée [197]: correct = np.where(predicted_classes==test_labels)[0]
print("Found %d correct labels" % len(correct))
for i, correct in enumerate(correct[:9]):
    print(i,"Predicted: ", label_ID[predicted_classes[correct]] )

Found 89141 correct labels
0 Predicted: positif
1 Predicted: positif
2 Predicted: positif
3 Predicted: positif
4 Predicted: positif
5 Predicted: positif
6 Predicted: positif
7 Predicted: positif
8 Predicted: positif

Entrée [198]: incorrect = np.where(predicted_classes!=test_labels)[0]
print("Found %d incorrect labels" % len(incorrect))
for i, incorrect in enumerate(incorrect[:9]):
    print(i, "Predicted: ", label_ID[predicted_classes[incorrect]] )

Found 24550 incorrect labels
0 Predicted: positif
1 Predicted: positif
2 Predicted: positif
3 Predicted: positif
4 Predicted: positif
5 Predicted: positif
6 Predicted: positif
7 Predicted: positif
8 Predicted: positif
```

Figure 20: L'affichage de résultats des prédictions.

9 Recommandation des services cloud

Dans cette partie de notre solution, nous avons utilisé le modèle que nous avons obtenu et enregistré. Nous avons utilisé ce modèle sur une partie des données exportées de notre Dataset.

9.1 Recommandation basée sur la popularité (Classification des services)

Dans cette partie de recommandation, nous nous sommes concentrés sur les services qui ont plus de 100 commentaires, et nous avons calculé la moyenne des commentaires positifs de chaque service.

$$Moy = \frac{Nbr_CPpS}{Nbr_CpS}$$

Où :

Moy : moyenne des commentaires positifs par service.

Nbr_CPpS : nombre de commentaires positifs par service.

Nbr_CpS : nombre de commentaires par service.

Après avoir obtenu la moyenne de chaque service, nous l'avons utilisée dans la formule **Weighted_Rating** (WR) afin que nous puissions obtenir la valeur du score de chaque service.

$$WR = \left(\frac{Nbr_CpS}{Nbr_CpS + M} \cdot Moy \right) + \left(\frac{M}{Nbr_CpS + M} \cdot C \right)$$

Où :

Nbr_CpS : nombre de commentaires par service.

M : est le nombre minimum de commentaires requis pour être répertorié.

C : est la somme des commentaires positifs / nombre de service.

Moy : moyenne des commentaires positifs par service.

Nous avons utilisé WR, donc la recommandation de services sera assez juste puisque le service avec 0,9 comme moyenne de commentaires positifs et seulement 3 commentaires ne peuvent pas être considérés comme meilleurs que le service avec 0,75 comme moyenne de commentaires positifs mais 150 commentaires.

Nous avons classé les services selon la valeur du WR à partir du meilleur, et nous avons recommandé les 5 meilleurs services pour chaque utilisateur.

9.2 Filtrage collaboratif basé sur la similarité des utilisateurs

Dans cette partie de recommandation, nous nous sommes concentrés sur les utilisateurs rassemblés par un commentaires positifs dans deux ou plusieurs services.

Ensuite, nous avons recommandé les 3 meilleurs services qui ne sont pas partagés entre des utilisateurs qui sont similaires dans un certain nombre de services, et nous l'avons fait avec tous les utilisateurs.

Pour chaque utilisateur $u \in U = \{1, 2, \dots, m\}$ qui commente positivement à un service $s \in S = \{1, 2, \dots, n\}$ on obtient un tableau de deux dimension $T = (U_i, S_{ij})$.

La similarité entre les utilisateurs sera calculée comme suit :

$$SIM = ([u_i][s_{ij}] \cap [u_{i+1}][s_{i+1,j}])$$

Tel que $i \in m$ et $j \in n$

Dans ce cas, nous utilisons une interface graphique pour faire la recommandation des services. La figure ci-dessous montre l'interface graphique que nous avons créée pour le système de recommandation.

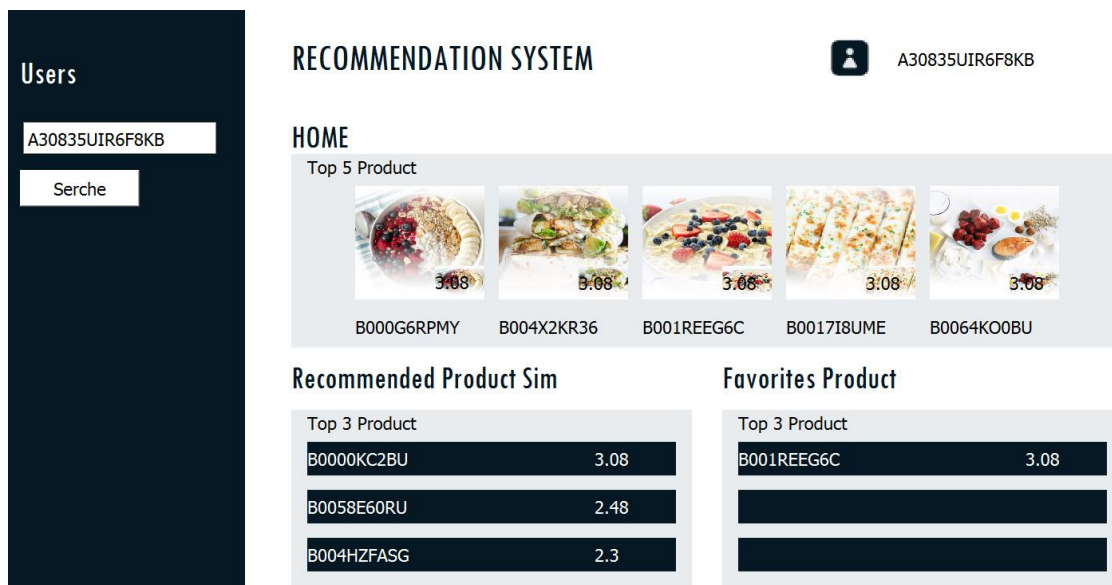


Figure 21: L'interface graphique de notre system de recommandation.

10 Code source

Maintenant, nous expliquons une partie de notre code source que nous avons utilisée pour construire notre modèle.

La figure ci-dessous montre le code source des données téléchargées et les packages importés qui sont utilisés dans notre travail.

```

Entrée [137]: import nltk
import string
from nltk.stem import WordNetLemmatizer
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('words')
nltk.download('wordnet')

[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\WELTINFO\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data] C:\Users\WELTINFO\AppData\Roaming\nltk_data...
[nltk_data] Package punkt is already up-to-date!
[nltk_data] Downloading package words to
[nltk_data] C:\Users\WELTINFO\AppData\Roaming\nltk_data...
[nltk_data] Package words is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data] C:\Users\WELTINFO\AppData\Roaming\nltk_data...
[nltk_data] Package wordnet is already up-to-date!

Out[137]: True

Entrée [138]: import tensorflow as tf
from tensorflow import keras
from keras.datasets import mnist
from keras.layers import Input, Dense, Flatten
from keras.models import Model
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import gzip
from keras.optimizers import RMSprop

```

Figure 22: Les données téléchargées et les packages importés.

La figure 23 présente le code source de la fonction de prétraitement que nous avons créée.

```

Entrée [139]: stopwords = nltk.corpus.stopwords.words('english')
words = set(nltk.corpus.words.words())
lemmatizer = WordNetLemmatizer()

Entrée [140]: def Preprocess_listofSentence(listofSentence):
preprocess_list = []
for sentence in listofSentence :
sentence_w_punct = "".join([i.lower() for i in sentence if i not in string.punctuation])

sentence_w_num = ''.join(i for i in sentence_w_punct if not i.isdigit())

tokenize_sentence = nltk.tokenize.word_tokenize(sentence_w_num)

words_w_stopwords = [i for i in tokenize_sentence if i not in stopwords]

words_lemmatize = (lemmatizer.lemmatize(w) for w in words_w_stopwords)

sentence_clean = ' '.join(w for w in words_lemmatize if w.lower() in words or not w.isalpha())

preprocess_list.append(sentence_clean)

return preprocess_list

```

Figure 23: La fonction de prétraitement que nous avons créée.

Dans la figure ci-dessous, nous présentons le code source de l'extraction des caractéristiques de notre dataset à l'aide de Word2vec pour l'étape de vectorisation dans notre modèle.

```

Entrée [145]: tokenize_sentences = []
              for paragraph in preprocess_list:
                  for i in range(len(paragraph)):
                      tokenize_sentences.append(nltk.tokenize.word_tokenize(paragraph[i]))
              #preprocess_list =None

Entrée [146]: from gensim.test.utils import common_texts
              from gensim.models import Word2Vec

              model_w2v = Word2Vec(sentences=tokenize_sentences , vector_size=200, window=5, min_count=1, workers=4)
              model_w2v.train(tokenize_sentences, total_examples=len(tokenize_sentences), epochs=100)
              model_w2v.save("../models/word2vec.model")

```

Figure 24: L'extraction des caractéristiques de notre dataset à l'aide de Word2vec pour l'étape de vectorisation.

La figure 25 montre le code source de l'initialisation des deux classes que nous avons utilisées dans la partie classification.

```

Entrée [160]: label_ID = {}
              num_classes = 2
              label_ID[1] = 'positif'
              label_ID[0] = 'negatif'

Entrée [161]: train_labels = np.array(train_labels)

```

Figure 25: L'initialisation des deux classes de commentaires.

La figure 26 montre le code source de l'extraction des caractéristiques de l'Auto-encoder pour la classification, et les couches encodées et décodées avec leurs caractéristiques, et la compilation du modèle Auto-encoder.

```

Entrée [170]: input_text= Input(shape=(260,))
#encoded and decoded layer for the autoencoder
encoded = Dense(120, activation='relu')(input_text)
encoded = Dense(10, activation='relu')(encoded)
decoded = Dense(120, activation='relu')(encoded)
decoded = Dense(260, activation='sigmoid')(decoded)

# Building autoencoder
autoencoder = Model(input_text, decoded)
encoder = Model(input_text, encoded)

# compiling the autoencoder
autoencoder.compile(optimizer= RMSprop(), loss='mean_squared_error')

# Fitting the noise trained data to the autoencoder

#filepath="autoencoder-train.hdf5"
#checkpoint = ModelCheckpoint(filepath, monitor='binary_crossentropy', verbose=1, save_best_only=False)

autoencoder.summary()
autoencoder_train = autoencoder.fit(train_X, train_X,
    epochs=nb_epoch,
    batch_size=32,
    shuffle=True,
    validation_data=(valid_X,valid_X),verbose=1
    )

```

Figure 26: Extraction de caractéristiques pour la classification et la compilation de l'Auto-encoder.

Dans les figures ci-dessous, nous présentons le code source que nous avons créé pour calculer la moyenne des commentaires positifs pour chaque service a plus de cent commentaires ainsi que le WR.

```

Entrée [ ]: #Calculate The Average Of Each Product -- average --
#Calculate The Number Of FeedBack Of Each Product -- countF --
average = []
countF = []
for produit in Id_ProduitMin :
    j=0
    i=0
    cpt=0
    for pro in Id_Produit :

        if pro == produit :
            j+=1
            cpt=predicted_classes[i]+cpt
            i+=1
    cpt = cpt / j
    countF.append(j)
    average.append(cpt)

```

Figure 27: Calcul de la moyenne des commentaires positifs.

```

Entrée [ ]: #Calculate The C
c = sum(countF) / len(countF)

Entrée [ ]: #weighted rating To calculate Score
score = []
for i in range(len(countF2)) :
    score.append(( (countF2[i] / (countF2[i] + m) * average2[i] )) + ( ( m / ( m + countF2[i] ) ) * c ))

```

Figure 28: Calcul de WR.

La figure 29 présente le code source de la recommandation des meilleurs services selon WR.

```

Entrée [ ]: #Sorting of the Score
score2.sort(reverse=True)

Entrée [ ]: #Table of Product Sorted By The Best Score
classification = []

for ide in score2 :
    i=0
    for scr in score :
        if ide == scr :
            classification.append(Id_ProduitMin2[i])
            i+=1

Entrée [ ]: classification = list(dict.fromkeys(classification))

Entrée [ ]: #Recommend Top 20 Product
print ("ProductID" , " Score")
classificationDict = {}
for i in range(len(score2)) :
    classificationDict[classification[i]] = score2[i]
print(classification[i] , score2[i])

```

Figure 29: La recommandation des meilleurs services selon WR.

La figure 30 montre le code source de la recommandation des services basée sur les utilisateurs similaires.

```

Entrée [ ]: #List Of ALL Recommendation For ALL User
recommandation = {}
for i in range(len(users)):
    elementes = []
    for j in range(i+1 , len(users)):
        if len(list(set(dicte[users[i]].intersection(dicte[users[j]]))) > 0 :
            if len(list(set(dicte[users[j]].difference(dicte[users[i]]))) != 0 :
                recommandation[users[i]] = list(set(dicte[users[j]].difference(dicte[users[i]])))
            if len(list(set(dicte[users[i]].difference(dicte[users[j]]))) != 0 :
                recommandation[users[j]] = list(set(dicte[users[i]].difference(dicte[users[j]])))

```

Figure 30: La recommandation des services basée sur les utilisateurs similaires.

Nous utilisons une interface graphique pour fournir une recommandation aux utilisateurs. La figure ci-dessous montre une description de l'interface graphique que nous construisons pour tester notre modèle.

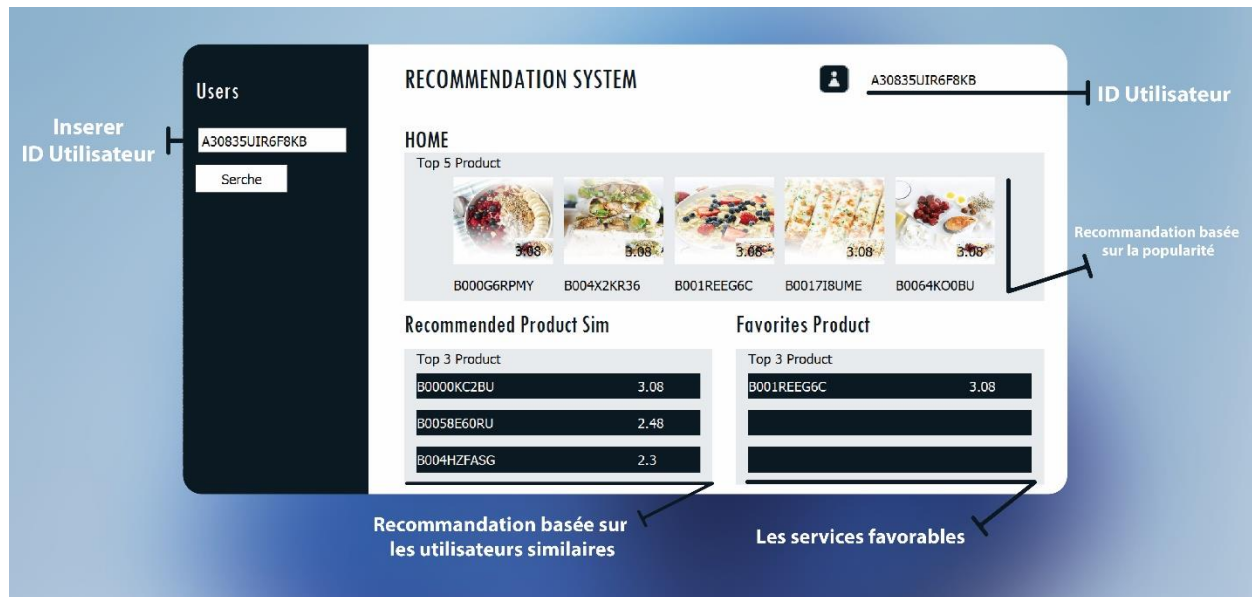


Figure 31: La description de l'interface graphique.

11 Métriques d'évaluation

Dans cette section, nous présenterons les métriques d'évaluation que nous avons utilisées pour évaluer notre nouveau modèle dans le domaine de la recommandation de services cloud.

Nous évaluons les résultats des prédictions à l'aide des métriques d'évaluation telles que le score F1 et la précision et calculons la perte des prédictions à l'aide de la fonction de perte.

- **Rappel (Recall)** : Le Rappel (R) est défini comme le nombre de vrais positifs (Vp) sur le nombre de vrais positifs plus le nombre de faux négatifs (Fn) comme suit [44] :

$$R = \frac{Vp}{Vp + Fn}$$

- **Précision** : La Précision (P) est définie comme le nombre de vrais positifs (Vp) sur le nombre de vrais positifs plus le nombre de faux positifs (Fp) comme suit [44] :

$$P = \frac{Vp}{Vp + Fp}$$

- **Exactitude (Accuracy)** : Exactitude est utilisée pour évaluer les modèles de classification. De manière informelle, l'exactitude est la fraction des prédictions de notre modèle qui sont correctes. Formellement, l'exactitude a la définition suivante [45] :

$$\text{Exactitude} = \frac{\text{Nombre de prédiction correcte}}{\text{Nombre total de prédiction}}$$

De plus, pour la classification binaire, l'exactitude peut être calculée en termes de positifs et de négatifs comme suit [45] :

$$\text{Exactitude} = \frac{Vp + Vn}{Vp + Vn + Fp + Fn}$$

Où Vp = vrais positifs, Vn = vrais négatifs, Fp = faux positifs et Fn = faux négatifs.

- **F1-score** : le F1-score, également appelé F-score, est une mesure de l'exactitude d'un modèle sur un ensemble de données. Il est utilisé pour évaluer les systèmes de classification binaires, qui classent par exemple en « positif » ou « négatif » [46].

Le score F1 peut être interprété comme une moyenne pondérée de la précision et du rappel, où un score F1 atteint sa meilleure valeur à 1 et le pire score à 0. La contribution relative de la précision et du rappel au score F1 est égale. La formule du score F1 est [47] :

$$F1 = 2 \times (\text{Précision} \times \text{Rappel}) / (\text{Précision} + \text{Rappel})$$

- **Perte (Loss)** : La perte est la pénalité pour une mauvaise prédiction. C'est-à-dire que la perte est un nombre indiquant à quel point la prédiction du modèle était mauvaise sur un seul exemple. Si la prédiction du modèle est parfaite, la perte est nulle ; sinon, la perte est plus importante. L'objectif de l'entraînement d'un modèle est de trouver un ensemble de poids et de biais qui ont une faible perte en moyenne, dans tous les exemples [48].

Pour mesurer l'efficacité de notre système de recommandation, trois types d'évaluations existent :

- **Études utilisateurs** : Les études d'utilisateurs sont une technique de jugement à petite échelle. Les recommandations générées par différentes approches de recommandation sont présentées à quelques dizaines ou centaines d'utilisateurs, puis les utilisateurs décident quelles recommandations sont les meilleures [49].

Les études d'utilisateurs sont très coûteuses à mener dans notre système de recommandation.

- **Évaluations hors ligne** : Dans une évaluation hors ligne de la prédiction d'utilisation, nous avons généralement un ensemble de données composé d'éléments que chaque utilisateur a utilisés. Nous sélectionnons ensuite un utilisateur test, masquons certaines de ses sélections et demandons au recommandeur de prédire un ensemble d'éléments que l'utilisateur utilisera [49].
- **Évaluations en ligne** : Dans de nombreuses applications de recommandation réalistes, le concepteur du système souhaite influencer le comportement des utilisateurs. Nous nous intéressons donc à mesurer le changement de comportement des utilisateurs lorsqu'ils interagissent avec différents systèmes de recommandation [49].

Les recommandations sont présentées à des milliers d'utilisateurs d'un produit réel, et le système de recommandation sélectionne au hasard au moins deux approches de recommandation différentes pour générer des recommandations.

Dans notre cas nous avons utilisé l'évaluation hors ligne, on a pris une partie aléatoire de notre dataset composé des services que chaque utilisateur a utilisé avec les commentaires. Nous sélectionnons ensuite un utilisateur et demandons à notre système de lui recommander des services qui peuvent satisfaire ses besoins.

12 Résultat

Dans la présente section, nous présentons les résultats que nous avons obtenus après l'évaluation.

Nous avons obtenu des résultats satisfaisants en utilisant la méthode d'apprentissage en profondeur (deep learning) Auto-encoder. Nous avons un 0,78 pour la l'exactitude et les mesures du score F1. Le tableau 5 montre l'exactitude et les résultats d'évaluation du score F1 que nous avons obtenu.

Tableau 5: Les résultats de notre modèle.

Exactitude	Score F1
0.78	0.78

De plus, la précision que nous obtenons dans la classe 0 est de 0,55 pour les commentaires négatifs, et de 0,79 dans la classe 1 pour les commentaires positifs. Il faut noter aussi que dans le rappel, nous avons obtenu 0,05 dans la classe 0 et nous avons obtenu 0,99 dans la classe 1. Le tableau 6 montre les résultats d'évaluation de la précision et du rappel que nous avons obtenus.

Tableau 6: Les résultats du rappel et la précision.

	Précision	Rappel
Classe 0	0.55	0.05
Classe 1	0.79	0.99

La figure ci-dessous montre les résultats de l'évaluation de notre modèle à l'aide de la métrique de l'exactitude. À partir de cette figure, nous pouvons voir que l'exactitude de notre modèle augmente jusqu'à atteindre la valeur 0,788 à l'époque 200.

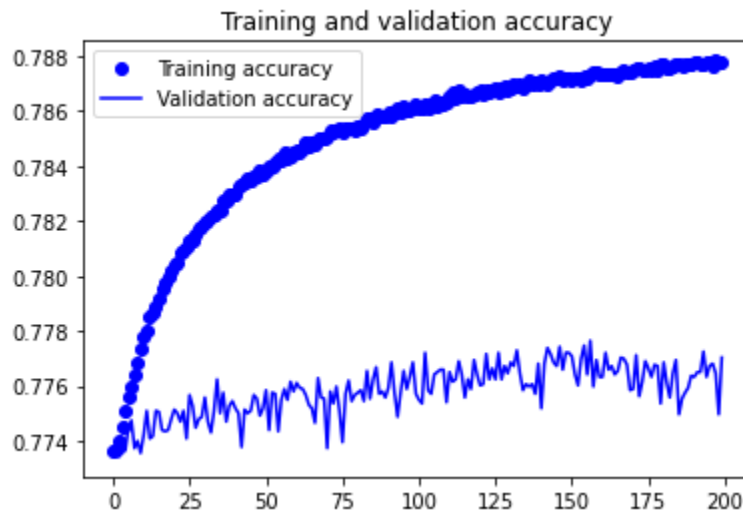


Figure 32: L'exactitude de l'entraînement et de la validation.

De plus, nous présentons combien nous obtenons la perte en utilisant la fonction de perte pour calculer la pénalité de la mauvaise prédiction, que nous avons obtenue lors de l'évaluation de notre modèle. La figure ci-dessous montre les variations de la perte.

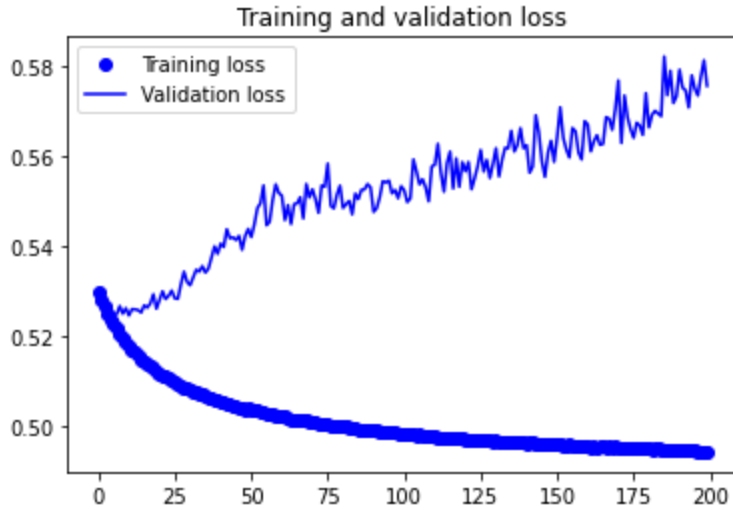


Figure 33: Le volume de perte de notre modèle.

La figure ci-dessous montre les résultats de recommandation pour chaque utilisateur.

```
list of recommendation for the user A1A81KEABE19Y :
ProductID | Score
B002ULC3KW | 2.2988122356003906
B002EDIJCM | 1.1398416314133855
B000EDGBA6 | 1.1071086649971467
B003A199AI | 1.062188727713928
B001FB6B8I | 0
-----
list of recommendation for the user A360IMHJ1EAW33 :
ProductID | Score
B002ULC3KW | 2.2988122356003906
B002EDIJCM | 1.1398416314133855
B000EDGBA6 | 1.1071086649971467
B003A199AI | 1.062188727713928
B001FB6B8I | 0
-----
list of recommendation for the user A26M5053PHZTKN :
ProductID | Score
B002ULC3KW | 2.2988122356003906
B002EDIJCM | 1.1398416314133855
B000EDGBA6 | 1.1071086649971467
B003A199AI | 1.062188727713928
B001FB6B8I | 0
-----
list of recommendation for the user A30835UIR6F8KB :
ProductID | Score
B0000KC2BU | 3.0780995769606254
B0058E60RU | 2.4843568406861607
B004HZFASG | 2.2988122356003906
B002JEYXSK | 1.9445907168002843
```

Figure 34: Les résultats de recommandation pour chaque utilisateur.

Discussion

Notre solution a montré des résultats satisfaisants en prédisant les avis des utilisateurs et en fournissant finalement de bonnes recommandations de service. Notre objectif principal était d'intégrer l'aspect social, chose que nous avons accompli via l'exploitation des commentaires utilisateurs. Nous considérons également que l'utilisation de l'Auto-encoder est un point original et important dans notre solution.

Il faut dire aussi que la disponibilité de datasets dans ce domaine reste un problème majeur, en effet, nous avons eu beaucoup de difficultés à trouver un dataset adéquats à nos besoins. Les datasets utilisés dans le domaine de la recommandation des services cloud étant tous payants ou à accès limité à une communauté particulière, (programmable, epinion...etc).

13 Conclusion

Dans ce dernier chapitre, nous avons présenté et implémenté notre solution avec les différents modules et nous avons montré les différentes étapes de la recommandation des services cloud.

Nous avons aussi testé notre proposition de solution du problème de recommandation et nous avons montré par les expérimentations de performance de notre solution qu'elle est capable de retourner de bons résultats de prédiction et de recommandation qui satisfont les besoins des utilisateurs.

Dans nos travaux futurs, nous planifions de tester notre proposition sur un Datasets dédiés aux services cloud, combiner notre solution avec QoS, ainsi une éventuelle comparaison avec les approches de la littérature.

CONCLUSION GENERALE

Le Cloud Computing propose des milliers de services qui ont des fonctionnalités différentes, beaucoup de ces services offrent les mêmes fonctionnalités. D'un autre côté, il est difficile pour les utilisateurs de trouver leurs besoins et de découvrir des services cloud qui répondent à leurs exigences.

D'une part, les demandes des utilisateurs peuvent ne pas être satisfaites par un service, où il est nécessaire d'avoir une suggestion de certains services pour satisfaire leurs besoins. D'une autre part, l'explosion des médias sociaux a créé une nouvelle source de données très fructueuse pour améliorer notre connaissance sur l'utilisateur qui est désormais producteur de l'information. C'est pourquoi l'intégration de l'aspect social (avis des utilisateurs) dans la recommandation des services cloud est très importante pour répondre aux besoins de ceux-ci.

Dans ce contexte, nous avons proposé une nouvelle solution à ce problème dans le cloud basé sur une approche hybride, qui prend en compte les avis des utilisateurs comme critère de recommandation. Avant la réalisation de notre nouvelle solution, nous avons défini des généralités sur le contexte général, qui est le Cloud Computing, et nous avons étudié la problématique de la recommandation des services cloud, avec la comparaison des différents travaux relatifs dans ce contexte. Ceci nous a aidé à modéliser et définir notre solution, puis la présentée et discutée après une série de tests et d'expérimentations.

Notre nouvelle solution est basée sur une approche hybride qui utilise l'Auto-encodeur qui est une technique d'apprentissage en profondeur, une recommandation basée sur la popularité et un filtrage collaboratif basé sur la similarité des utilisateurs pour recommander des services cloud aux utilisateurs en utilisant leurs commentaires. Nous avons utilisé Python comme langage de programmation et Jupyter comme environnement de travail pour développer et tester notre solution. Les expérimentations montrent que notre solution retourne de bons résultats de prédiction et de recommandation qui satisfont les besoins des utilisateurs.

Cependant, notre solution laisse place à d'autres améliorations. Comme travaux futurs, nous pensons qu'il est possible d'optimiser notre modèle et d'utiliser plus de techniques et combiner notre solution avec QoS pour obtenir de bons résultats de recommandation en optimisant le facteur temps. De plus, nous aimerions faire de notre nouvelle solution un service en ligne et l'utiliser pour

la recommandation des services cloud. Nous planifions également de tester notre solution sur un dataset dédié aux services cloud, chose qui n'a pas été possible dans le cadre de ce travail par cause d'indisponibilité de ces datasets.

BIBLIOGRAPHIE

- [1] Final Version of NIST Cloud Computing Definition Published. (2018b, January 8). NIST. <https://www.nist.gov/news-events/news/2011/10/final-version-nist-cloud-computing-definition-published>.
- [2] *Qu'est-ce que le cloud computing ? Guide du débutant*. (n.d.). Microsoft Azure. Retrieved February 27, 2021, from <https://azure.microsoft.com/fr-fr/overview/what-is-cloud-computing/>.
- [3] Aaqib Rashid, Amit Chaturvedi, (2019). Cloud Computing Characteristics and Services: A Brief Review. *International Journal of Computer Sciences and Engineering*, 7(2), 421-426.
- [4] Fu, A. (2017, March 3). *7 Différents types de Structures de Calcul en Nuage que vous devriez savoir*. Uniprint.Net. <https://www.uniprint.net/fr/7-types-cloud-computing-structures>.
- [5] Joint, A., & Baker, E. (2011). Knowing the past to understand the present—issues in the contracting for cloud based services. *Computer Law & Security Review*, 27(4), 407-415.
- [6] Lee, R. (2018). *Big Data, Cloud Computing, Data Science & Engineering (Studies in Computational Intelligence, 786)* (1st ed. 2019 ed.). Springer.
- [7] Jansen, W. (2011, December 9). *SP 800–144, Guidelines on Security and Privacy in Public Cloud Computing | CSRC*. Csrc.Nist.Gov. <https://csrc.nist.gov/publications/detail/sp/800-144/final>.
- [8] Demchenko, Y. (2013, July 17). *Defining the Big Data Architecture Framework (BDAF)*. Bigdatawg.Nist.Gov. https://bigdatawg.nist.gov/_uploadfiles/M0055_v1_7606723276.pdf.
- [9] *Thèse de Doctorat. Autonomie, sécurité et QoS de bout en bout dans un environnement de Cloud Computing MOHAMAD HAMZE - PDF Téléchargement Gratuit*. (2015, December 7). <Http://Www.Theses.Fr/>. <https://docplayer.fr/33144839-These-de-doctorat-autonomie-securite-et-qos-de-bout-en-bout-dans-un-environnement-de-cloud-computing-mohamad-hamze.html>.

- [10] Dillon, T., Wu, C., & Chang, E. (2010). Cloud Computing: Issues and Challenges. *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, 27-33.
- [11] Samani, R., Reavis, J., & Honan, B. (2014). *CSA Guide to Cloud Computing*. Elsevier Gezondheidszorg.
- [12] Jadeja, Y., & Modi, K. (2012). Cloud computing - concepts, architecture and challenges. *2012 International Conference on Computing, Electronics and Electrical Technologies (ICCEET)*, 877–880. <https://doi.org/10.1109/icceet.2012.6203873>.
- [13] Kouki, Y., Ledoux, T., Serrano, D., Bouchenak, S., Lejeune, J., Arantes, L., ... & LIP6–INRIA, P. (2013, January). SLA et qualité de service pour le Cloud Computing. In *Conférence d'informatique en Parallélisme, Architecture et Système, ComPAS* (Vol. 2013).
- [14] Sindhu, J., & Roopa, G. M. (2018). A Survey-Approaches and Challenges for Cloud Based Recommendation System.
- [15] Jiang, Y., Tao, D., Liu, Y., Sun, J., & Ling, H. (2019). Cloud service recommendation based on unstructured textual information. *Future Generation Computer Systems*, 97, 387-396.
- [16] Wang, Y., Wen, J., Zhou, W., Tao, B., Wu, Q., & Tao, Z. (2019). A cloud service selection method based on trust and user preference clustering. *IEEE Access*, 7, 110279-110292.
- [17] Mezni, H., & Abdeljaoued, T. (2018). A cloud services recommendation system based on Fuzzy Formal Concept Analysis. *Data & Knowledge Engineering*, 116, 100-123.
- [18] Ding, S., Li, Y., Wu, D., Zhang, Y., & Yang, S. (2018). Time-aware cloud service recommendation using similarity-enhanced collaborative filtering and ARIMA model. *Decision Support Systems*, 107, 103-115.
- [19] Zheng, X., Da Xu, L., & Chai, S. (2017). Qos recommendation in cloud services. *IEEE Access*, 5, 5171-5177.

- [20] Meng, S., Zhou, Z., Huang, T., Li, D., Wang, S., Fei, F., ... & Dou, W. (2016, June). A temporal-aware hybrid collaborative recommendation method for cloud service. In *2016 IEEE International Conference on Web Services (ICWS)* (pp. 252-259). IEEE.
- [21] Afify, Y. M., Moawad, I. F., Badr, N. L., & Tolba, M. F. (2017). A personalized recommender system for SaaS services. *Concurrency and Computation: Practice and Experience*, 29(4), e3877.
- [22] Kumar, T. S., & Pandey, S. (2015). Customization of recommendation system using collaborative filtering algorithm on cloud using mahout. In *Intelligent Distributed Computing* (pp. 1-10). Springer, Cham.
- [23] Yan, S., Chen, C., Zhao, G., & Lee, B. S. (2012, October). Cloud service recommendation and selection for enterprises. In *2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualization management (svm)* (pp. 430-434). IEEE.
- [24] Han, S. M., Hassan, M. M., Yoon, C. W., & Huh, E. N. (2009, November). Efficient service recommendation system for cloud computing market. In *Proceedings of the 2nd international conference on interaction sciences: information technology, culture and human* (pp. 839-845).
- [25] Rodrigues, R. B. (2013). *A CLOUD-BASED RECOMMENDATION SYSTEM*. IADIS. <http://www.iadisportal.org/digital-library/a-cloud-based-recommendation-system>.
- [26] Ma, H., & Hu, Z. G. (2015). User preferences-aware recommendation for trustworthy cloud services based on fuzzy clustering. *Journal of Central South University*, 22(9), 3495-3505.
- [27] Yu, Q. (2015). CloudRec: a framework for personalized service Recommendation in the Cloud. *Knowledge and Information Systems*, 43(2), 417-443.
- [28] Aznoli, F., & Navimipour, N. J. (2017). Cloud services recommendation: Reviewing the recent advances and suggesting the future research directions. *Journal of Network and Computer Applications*, 77, 73-86.
- [29] Nagarajan, R., & Thirunavukarasu, R. (2019). A service context-aware QoS prediction and recommendation of cloud infrastructure services. *Arabian Journal for Science and Engineering*, 1-15.

- [30] Hao, F., Pei, Z., Park, D. S., Phonexay, V., & Seo, H. S. (2018). Mobile cloud services recommendation: a soft set-based approach. *Journal of Ambient Intelligence and Humanized Computing*, 9(4), 1235-1243.
- [31] Somu, N., MR, G. R., Kirthivasan, K., & VS, S. S. (2018). A trust centric optimal service ranking approach for cloud service selection. *Future Generation Computer Systems*, 86, 234-252.
- [32] *What are the possible job roles in cloud computing?* - Quora. (2018). Quora.Com. <https://www.quora.com/What-are-the-possible-job-roles-in-cloud-computing>.
- [33] Bohn, R. B., Messina, J., Liu, F., Tong, J., & Mao, J. (2011, July). NIST cloud computing reference architecture. In *2011 IEEE World Congress on Services* (pp. 594-596). IEEE.
- [34] Das, D., Sahoo, L., & Datta, S. (2017). A survey on recommendation system. *International Journal of Computer Applications*, 160(7).
- [35] Lawton, G. (2020, July 22). *data preprocessing*. SearchSQLServer. <https://searchsqlserver.techtarget.com/definition/data-preprocessing>.
- [36] Jang, B., Kim, M., Harerimana, G., Kang, S. U., & Kim, J. W. (2020). Bi-LSTM model to increase accuracy in text classification: Combining Word2vec CNN and attention mechanism. *Applied Sciences*, 10(17), 5841.
- [37] B, T. (2020, October 30). *Word2vec : NLP & Word Embedding | DataScientest*. Formation Data Science | DataScientest.com. <https://datascientest.com/nlp-word-embedding-word2vec>.
- [38] *Why TensorFlow*. (2021). TensorFlow. <https://www.tensorflow.org/about>.
- [39] Team, K. (2021). *Keras: the Python deep learning API*. Keras. <https://keras.io/>.
- [40] *gensim*. (2021, April 1). PyPI. <https://pypi.org/project/gensim/>.
- [41] *scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation*. (2021). Scikit-Learn. <https://scikit-learn.org/stable/>.
- [42] *What is NumPy? — NumPy v1.21 Manual*. (2021, June 22). Numpy. <https://numpy.org/doc/stable/user/whatisnumpy.html>.
- [43] *Amazon Fine Food Reviews*. (2017, May 1). [Dataset]. <https://www.kaggle.com/snap/amazon-fine-food-reviews>.

- [44] Davis, J., & Goadrich, M. (2006, June). The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning* (pp. 233-240).
- [45] *Classification: Accuracy | Machine Learning Crash Course*. (2020). Google Developers. <https://developers.google.com/machine-learning/crash-course/classification/accuracy>.
- [46] Wood, T. (2020, August 7). *F-Score*. DeepAI. <https://deepai.org/machine-learning-glossary-and-terms/f-score>.
- [47] *sklearn.metrics.f1_score — scikit-learn 0.24.2 documentation*. (2020). Sklearn. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html.
- [48] *Descending into ML: Training and Loss | Machine Learning Crash Course*. (2020). Google Developers. <https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss>.
- [49] Gunawardana, A., & Shani, G. (2015). Evaluating recommender systems. In *Recommender systems handbook* (pp. 265-308). Springer, Boston, MA.
- [50] Singh, V., & Verma, N. K. (2018). Deep learning architecture for high-level feature generation using stacked auto encoder for business intelligence. In *Complex systems: solutions and challenges in economics, management and engineering* (pp. 269-283). Springer, Cham.
- [51] Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, 52(1), 1-38.
- [52] Karatzoglou, A., & Hidasi, B. (2017, August). Deep learning for recommender systems. In *Proceedings of the eleventh ACM conference on recommender systems* (pp. 396-397).
- [53] DeepAI. (2020, June 25). *Hidden Layer*. <https://deepai.org/machine-learning-glossary-and-terms/hidden-layer-machine-learning>.
- [54] Kamath, U., Liu, J., & Whitaker, J. (2019). *Deep learning for NLP and speech recognition* (Vol. 84). Cham: Springer.