Approches hybrides pour la recommandation dans le domaine du pneumatique

Assaad Kenaan* , Khalid Benabdeslem*
Kilian Bourhis**, Bruno Canitia**

*Université Lyon 1, 43 Boulevard du 11 novembre 1918, Villeurbanne cedex 69622
**Lizeo IT, 42 Quai Rambaud, 69002 Lyon
assaad.r.kenaan@gmail.com

Résumé. De nombreuses approches de recommandation ont été exploitées pour recueillir les préférences des utilisateurs afin d'améliorer leur navigation et le taux de transformation. Dans certains cas, nous ne disposons pas d'information explicite sur les utilisateurs ou les produits. La reconstruction de leurs profils, à partir d'éléments indirects devient donc nécessaire. Notre objectif est donc d'exploiter des facteurs sous-jacents aux interactions entre les utilisateurs et les produits, afin d'enrichir notre moteur de recommandation. Nous proposons ici une nouvelle approche hybride - factorisation de matrices et réseau de neurones convolutif - qui répond au problème posé par un industriel dans le domaine du pneumatique. Notre algorithme a été évalué sur un ensemble de données réelles extrait d'un comparateur en ligne. L'étude a montré que notre modèle est plus performant que ceux de l'état de l'art et celui-ci nous a permis d'étudier l'impact des différents types d'information utilisés sur les résultats.

1 Introduction

De la croissance des sites commerciaux à la diversité de leurs produits, suivie de la montée en puissance des publicités en ligne, les systèmes de recommandation sont devenus des incontournables lors de notre navigation sur internet. Ces systèmes sont capables de proposer un ensemble de produits susceptibles d'intéresser un utilisateur. Dans le cas où ce dernier recherche un produit à acheter, le système devra le guider en affinant ses propositions au fur et à mesure des recherches effectuées.

De nombreux algorithmes de recommandation ont été mis en oeuvre, ceux-ci utilisent des données fournies explicitement et/ou implicitement par les utilisateurs. Cependant, dans notre cas applicatif (un comparateur en ligne), plusieurs contraintes s'imposent au niveau des données. D'une part, les utilisateurs ne peuvent pas se créer de profils et nous ne possédons aucune autre source d'informations explicites sur eux. D'autre part, nous ne pouvons pas reconnaître un utilisateur revenant après plusieurs jours d'inactivité. Enfin, les utilisateurs ne peuvent pas acheter de produits directement sur le comparateur, ils sont pour cela redirigés vers un site de vente et nous n'avons aucun moyen de vérifier qu'une transaction ait bien été effectuée. Cependant, nous pouvons reconstituer les actions des utilisateurs durant une session et nous possédons un ensemble d'informations expertes vis à vis des produits. Ces contraintes, nous

ont amenés à aborder les deux approches suivantes, le filtrage collaboratif et la recommandation par session.

Le filtrage collaboratif (Collaborative Filtering) (Schafer et al. (2007)) est une approche qui utilise les interactions des utilisateurs de type explicite ou implicite. Elle est fondée principalement sur la préférence historique des utilisateurs pour un ensemble d'éléments. Son hypothèse s'appuie sur le fait qu'un utilisateur ait tendance à acheter les mêmes produits que les autres utilisateurs ayant des goûts similaires.

L'idée de **l'approche basée sur les sessions (Session-Based)** (Hidasi et al. (2015); Godard (2017)) consiste à recommander aux utilisateurs, des éléments qui ont été présentés dans un contexte similaire. Ceci est réalisé en étudiant la séquence d'évènements ordonnées chronologiquement durant la session, ce qui permet la détection de patterns ou de comportements intéressants. Dans cette approche, on peut distinguer deux objectifs (Wang et al. (2019)) : la recommandation du «next-item» ou l'élément suivant de la session en cours ; et la recommandation du «next-basket» ou le panier suivant de la session entière. Dans le cas du «next-basket», on prend en compte deux éléments principaux. Soit $U = \{u_1, u_2, \ldots, u_m\}$, l'ensemble des utilisateurs et $I = \{i_1, i_2, \ldots, i_n\}$, l'ensemble des produits. Les interactions entre ces deux ensembles, durant une période précise, sont les principales informations utilisées par les systèmes de recommandation. Elles forment les sessions que l'on utilise dans la détection de comportements. Une session est donc une séquence chronologique d'interactions effectuées par un utilisateur, que nous pouvons représenter de la manière suivante $S = \{(u_a, i_x), (u_a, i_y), \ldots, (u_a, i_z)\}$. C'est cet ensemble de sessions S qui constituent le coeur des données utilisées dans cette approche.

En **hybridation** (He et al. (2017); Zhang et al. (2019); Kim et al. (2016)), il est tout à fait possible de combiner deux ou plusieurs approches de types différents ou du même type. Le but est de profiter des avantages des approches prises en compte et de surmonter ainsi certains inconvénients qu'un système seul ne peut surmonter. Parmi ces inconvénients, on peut citer : le démarrage à froid (Cold Start) ; le problème de l'éparpillement (Data Sparsity) ; et le passage à l'échelle (Scalability).

Nous proposons dans un premier temps, d'explorer les techniques d'hybridation puis de revenir sur les principales techniques liées au filtrage collaboratif et à la recommandation par session. Nous présenterons ensuite notre approche dans une vision générique indépendante de notre domaine applicatif, le pneumatique. Enfin, nous présenterons l'adaptation de notre modèle aux données applicatives ainsi que notre étude comparative grâce à une baseline commune avec l'état de l'art. Nous profiterons de cette étude comparative pour étudier l'impact des différents types de données que nous utilisons dans notre modèle.

2 Techniques d'hybridation

Dans cette section, nous explorons les différentes techniques d'hybridation initialement établie par (Burke (2002). Il est important d'avoir cette vue d'ensemble afin de comprendre l'intérêt de l'hybridation que nous avons sélectionnée. Celles-ci sont classées en 7 catégories, décrites ci-dessous.

Hybridation pondérée. Cette technique produit une seule recommandation par l'addition des scores provenant de plusieurs systèmes de recommandation fonctionnant en parallèle.

Chaque système génère une liste de produits avec un score pour chacun. Ensuite, une pondération des scores est produite et finalement, une liste de produits classés est ainsi recommandée.

Hybridation à bascule. En fonction de l'état actuel, le système va pouvoir alterner entre deux systèmes de recommandation différents. Cette technique est similaire à celle de la pondération, à la différence qu'au niveau de la sortie des systèmes de recommandation, chacun renvoie un ensemble de produits, I_1 et I_2. Une procédure de sélection des produits selon certains critères sera effectuée par la suite, et finalement le résultat sera soit I_1, soit I_2.

Hybridation mixée. À partir de deux systèmes de recommandation, on obtient une liste recommandée de produits en concaténant le résultat de ces deux systèmes. En partant sur le même principe que celui de la pondération et de la bascule, cette technique consiste à générer deux ensembles, I_1 et I_2. La différence se présente dans la deuxième étape, où I_1 et I_2 seront unis et la recommandation finale sera l'union des ensembles I_1 et I_2.

Hybridation par ajout de caractéristiques. Dans cette approche, le premier système de recommandation calcule un attribut ou un ensemble d'attributs, qui seront utilisés par un second système de recommandation. Cette technique est sensible à l'ordre car le deuxième modèle utilise uniquement la sortie du premier système. Cette astuce permet d'utiliser des caractéristiques que le second modèle n'aurait pas pu générer.

Hybridation par combinaison de caractéristiques. Cette approche est similaire à la précédente à la différence que le second modèle, en plus d'utiliser les caractéristiques issues du premier modèle, prend aussi en entrée d'autres sources de données qui peuvent être différentes de celles du premier modèle. Dans ce cas de figure, nous avons plus de possibilité dans le choix du premier modèle car le second ne dépend pas uniquement de lui.

Hybridation en cascade. Cette approche consiste à recommander une liste d'items produite par un système de recommandation et filtrée ensuite par un autre système. Le but est d'affiner le résultat à chaque fois.

Hybridation méta-niveau. Il s'agit d'un modèle généré par un système de recommandation, et réutilisé par le deuxième système. Cette technique consiste à produire un modèle appris grâce au premier système, qui servira d'entrée avec les données utilisateurs pour le second système.

Discussion. Notre objectif est d'hybrider deux systèmes issus du filtrage collaboratif et de la recommandation par session, afin de tirer profit de ces deux approches. La première nous permet de reconstruire des « **profils** » d'utilisateurs et de produits à partir d'interactions explicites ou implicites. La seconde ajoute une dimension **temporelle** aux interactions (les sessions) et permet de reconnaître des patterns communs à plusieurs sessions. En ce sens, les quatre premières techniques ne sont pas compatibles car on nous ne pouvons choisir ces deux approches sans perdre l'aspect temporelle des sessions. Les deux dernières techniques peuvent être appliquées, mais elles ne nous paraissent pas assez pertinentes au niveau du gain d'informations apporté au second modèle. En revanche, l'**hybridation par combinaison de caractéristiques** offre tout ce dont nous avons besoin. En utilisant le filtrage collaboratif comme premier modèle et un second basé sur les sessions en plus des profils utilisateurs et produits, nous pensons tirer pleinement partie de ces deux approches. Cette réflexion nous a menés à étudier les hybridations existantes ainsi que les principales techniques liées à ces deux approches en vue de les hybrider.

Systèmes hybrides de recommandation.

3 Travaux liés

Dans cette section, nous présentons certaines des techniques qui nous paraissent les plus efficientes pour chacune des deux approches que nous voulons hybrider puis nous étudierons quelques systèmes hybrides de la littérature.

3.1 La factorisation de matrices

Avec cette technique très largement utilisée dans le **filtrage collaboratif**, les utilisateurs vont, tout au long de leurs actions, fournir au moteur de recommandation des données explicites et/ou implicites. Ces données représentent l'intérêt qu'ils portent envers un ou plusieurs produits et peuvent être représentées sous la forme d'une matrice $R^{m \times n}$. Dans la plupart des cas, la matrice R est très éparse, car l'utilisateur n'évalue qu'une partie des produits qu'il a déjà rencontré, ce qui représente une infime partie de l'ensemble des produits. L'objectif est de chercher la manière dont un utilisateur évalue un élément. Ceci nous permet d'extraire des facteurs latents, afin de reconstituer les profils d'utilisateurs et de produits. L'idée est de décomposer la matrice R dont les lignes représentent les utilisateurs et les colonnes représentent les produits, en deux ou plusieurs sous matrices $P^{U \times k}$ et $Q^{I \times k}$ tel que leur produit reconstruit la matrice initiale R, suivant l'équation : $R \approx P \times Q^T = R$. Avec k le nombre de facteurs latents, $P^{m \times k}$ la matrice latente des utilisateurs et $Q^{n \times k}$ la matrice latente des produits.

Décomposition en valeurs singulières - SVD. Dans le cadre de la recommandation, cette technique vise à reconstruire la matrice d'évaluation utilisateur-produit qu'on note R avec des valeurs cibles, et de réduire ainsi sa dimension en extrayant ses facteurs latents. Cependant, comme la plupart des ensembles de données du monde réel sont éparses, notre matrice d'évaluation R le sera aussi. Dans ce cas, la méthode des moindres carrés n'est appliquée que pour les entrées observées de la matrice cible R, ce qui peut entraîner un problème d'optimisation non convexe (Mnih et Salakhutdinov (2008)).

Element-Wise Alternating Least Square - eALS. (He et al. (2016a)) Ce modèle prend en entrée une matrice R construite à partir d'interactions implicites :

$$r_{ui} = \begin{cases} 1 & \text{si } \{u, i\} \in S. \\ 0 & \text{sinon.} \end{cases} \quad (1)$$

Avec r_{ui} le score de l'utilisateur u envers le produit i.

L'eALS fonctionne suivant deux modes différents : *offline* pour pré-entraîner le modèle, et *online* pour traiter les nouvelles données en temps réel. Ce modèle se distingue par sa capacité à gérer les produits populaires. En effet, un produit est dit populaire lorsqu'il est présent souvent sur l'ensemble des interactions utilisateur-produit. L'eALS attribue un poids aux produits selon leur popularité, cela indique que s'il n'a pas été vu par un utilisateur, alors il y a de forte chance qu'il ne l'intéresse pas. Ainsi, ce modèle calcule la confidence d'un produit qui désigne la probabilité que celui-ci n'ait pas été vu par l'utilisateur.

3.2 Les réseaux de neurones profonds

Les réseaux de neurones et l'apprentissage profond font partie des meilleures solutions à de nombreux problèmes de recommandation. Il s'agit d'un ensemble d'algorithmes inspirés de la biologie qui permettent d'apprendre des relations complexes à partir de données d'observation.

Le **GRU4Rec.** est un réseau de neurones récurrents (Hidasi et al. (2015)) qui traite les séquences d'évènements grâce à ses mémoires et ses portes. Son objectif est de prédire selon les séquences d'évènements effectuées auparavant par l'utilisateur, le prochain produit susceptible de l'intéresser. Il rentre dans le cadre de la **recommandation par session** et permet donc de garder l'information temporelle. Cependant, ce modèle est un peu limité quant aux données qu'il peut recevoir en entrée.

3D Convolutional Neural Network. ou 3D CNN (Tuan et Phuong (2017)) est un réseau de neurones convolutif ayant plusieurs objectifs : capturer les **relations spatio-temporelles** ; reconnaître un ou plusieurs objets dans une vidéo et recommander. À la différence du CNN classique, il peut recevoir en entrée une séquence d'images, donc une suite de matrices, et applique via des couches intermédiaires (convolutions, pooling, etc.) de dimension 3, des opérations de deux catégories : les filtres pour extraire les informations ; et le classifieur pour analyser ces informations. L'avantage de ce modèle est sa capacité à pouvoir traiter n'importe quel type d'information du moment que l'on garde une structure cohérente durant l'encodage.

3.3 Les modèles hybrides

DMF (Deep Matrix Factorization). C'est un modèle d'hybridation (Xue et al. (2017)) entre une factorisation de matrices et un réseau de neurones DSSM (Deep Semantic Similarity Model). Un DSSM est un réseau de neurones qui vise à capturer une similarité sémantique entre deux phrases. L'idée est de produire via deux branches, les vecteurs latents des utilisateurs et les vecteurs latents des produits. Ensuite, l'objectif sera de reproduire la matrice en appliquant la fonction cosinus sur la concaténation de la sortie de chaque branche, c.à.d. sur les deux vecteurs latents. On peut voir le DMF comme une hybridation entre la structure du DSSM et la représentation des données de la FM. Cependant, cette approche n'est toujours pas prometteuse car il s'agit toujours d'une approche de type Collaborative Filtering, où l'aspect temporel n'est pas présent.

NeuMF (Neural Collaborative Filtering). Le NeuMF (He et al. (2017)) est une hybridation qui apprend des données implicites entre un modèle de factorisation de matrices noté GMF, Generalized Matrix Factorization, et un perceptron multicouche noté MLP. Son objectif est de pouvoir transformer une matrice d'interactions (utilisateur-produits) éparse, en une matrice d'interactions dense grâce aux valeurs prédites. D'une part, le GMF va pouvoir modéliser les facteurs latents des interactions utilisateur-produit d'une manière linéaire. D'autre part, le MLP va aussi modéliser ces facteurs latents mais d'une manière non linéaire. Cette approche est applicable mais elle n'est pas prometteuse dans notre cas, car en l'appliquant, nous perdons l'aspect temporel.

DeepFM (Deep Factorization Machine). Il s'agit d'une hybridation (Guo et al. (2017)) entre une machine de factorisation FM et un réseau de neurones MLP. Cet algorithme met l'accent sur la puissance de la factorisation de matrices, et combine cette technique avec l'apprentissage profond. L'objectif est d'obtenir à la fin, un modèle capable de prédire dans un contexte donné, la probabilité qu'un utilisateur choisisse un élément. D'une part, la branche FM capture les interactions linéaires d'ordre 1 et 2 entre les caractéristiques. D'autre part, la branche MLP capture les interactions non linéaires d'ordre supérieur. Cette approche est très efficace et ne nécessite pas une expérience d'ingénierie particulière. Cependant, le modèle ne tient pas compte du temps, donc nous perdons de l'information pour la prédiction.

Systèmes hybrides de recommandation.

ConvMF (Convolutional Matrix Factorization). ConvMF (Kim et al. (2016)) est un modèle hybride qui intègre le modèle CNN (Convolutional Neural Network) avec le PMF (Probabilistic Matrix Factorization) (Mnih et Salakhutdinov (2008)), dans le but de prédire un score inconnu. L'idée de cette hybridation est de pouvoir utiliser à la fois l'information collaborative (au niveau du modèle PMF) et l'information contextuelle (au niveau du modèle CNN). Le sous modèle CNN prend en entrée les informations sur les documents, ainsi qu'une variable ϵ en tant que bruit Gaussien afin de générer les facteurs latents des produits. Ce dernier joue le rôle d'un pont de connexion entre les deux sous-modèles dans le but d'utiliser et d'exploiter efficacement les deux informations, collaborative et contextuelle. Les facteurs latents des utilisateurs sont ensuite estimés par le maximum à posteriori. Finalement, le PMF est appliqué afin de reconstruire la matrice R. Ce modèle est parmi les modèles les plus proches de notre cas. Il s'agit d'une factorisation de matrices combinée avec le modèle CNN qui pourrait améliorer l'exactitude de la recommandation. Cependant, avec cette approche l'axe temporel n'est encore une fois pas exploité et cela nous a donc pousser à proposer notre propre approche d'hybridation.

Discussion. Comme les solutions hybrides déjà existantes ne répondaient pas à nos attentes, nous nous sommes tournés vers les techniques non-hybrides. C'est ainsi que nous avons retenus l'eALS et le 3D CNN du fait de leurs performances, déjà démontrées par leurs auteurs et dans l'étude de Bourhis et al. (2019), et de leur compatibilité avec la technique d'hybridation que nous avons retenue.

4 Approche proposée

L'approche que nous proposons, consiste en une hybridation entre un modèle de factorisation de matrices (eALS) et un réseau de neurones profond (3D CNN). Le premier modèle, pouvant être remplacé par une autre FM, permet de répondre à la problématique de l'éparpillement et nous aide à construire un profil pour l'utilisateur. Le deuxième, central dans notre approche, permet d'analyser les comportements desdits utilisateurs, à travers leurs sessions, et de détecter des patterns qui vont construire le modèle d'apprentissage final. L'idée est de pouvoir combiner les deux modèles eALS et 3D CNN via la technique d'hybridation par combinaison des caractéristiques (c.f. section 2). Cela nous permet de pré-calculer un profil pour chaque utilisateur et chaque produit via l'eALS, afin d'enrichir nos sessions tout en gardant leur aspect temporel via le 3D CNN. La figure 1 montre une vue globale de notre système.

La figure 2 illustre la construction d'une session encodée qui servira d'entrée au 3D CNN. Une session peut être représentée comme un empilement de couches (matrices) formant un pavé. Chaque couche, représente une interaction utilisateur-produit et est empilée chronologiquement dans le pavé. Les couches sont composées de plusieurs bandes représentant chacune une information concernant l'interaction courante. Ces informations peuvent être encodées de plusieurs manières et auront un impact direct sur la taille des sessions (dimensions du pavé), le temps d'exécution du système ainsi que ses performances.

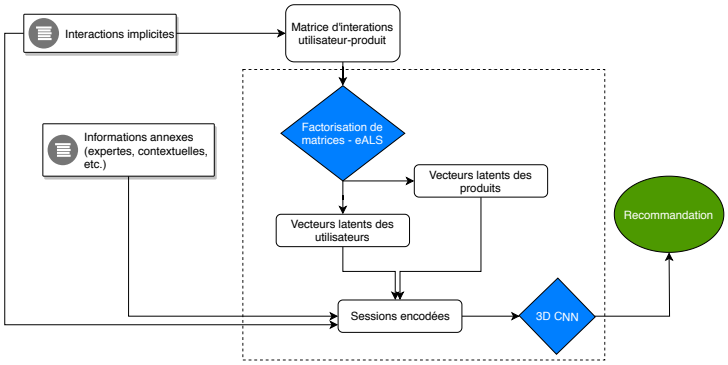

FIG. 1: Architecture globale de notre approche.

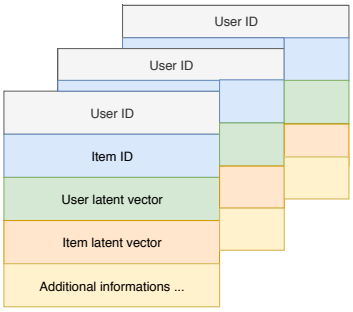

FIG. 2: Exemple de construction d'une session en entrée de 3D CNN.

Grâce à ces sessions (d'au moins 2 interactions), nous pouvons entraîner le 3D CNN itérativement, en essayant de prédire l'ID du produit (itemID) de l'interaction N+1 grâce aux informations des N premières interactions, avec N >= 1.

Encodage. Dans nos expérimentations, nous testerons deux types d'encodage. Le premier, « Character-Level Encoding » est proposé par les auteurs de l'article Tuan et Phuong (2017). L'idée est d'encoder en un vecteur « One Hot » chaque caractère d'une information. Le second, est l'encodage binaire (sur 32 bits), utilisé pour réduire la taille de nos sessions par rapport au premier encodage. L'encodage binaire nous permet d'encoder nos informations sur le même nombre de dimensions, sans avoir recours au padding (ajout de 0) qui rend le character-level encoding très éparse.

Architecture 3D CNN. L'architecture du 3D CNN étant fortement dépendante des données en entrée, nous présentons dans la figure 4 (en annexe), la version la plus complète de notre système parmi celles utilisées dans nos expérimentations. Cette architecture ainsi que les paramètres des couches, ont été fixés selon la même méthode employée par Tuan et Phuong (2017). La seule différence étant l'ajout de couches de convolution où le stride est à (1,1,1). L'ajout de ces couches suit le principe des blocs résiduels He et al. (2016b) et semble participer à l'amélioration des performances du système.

Systèmes hybrides de recommandation.

5 Données industrielles et protocole expérimental

Nous avons mené nos expérimentations sur des données réelles provenant d'un comparateur en ligne de pneumatique. Ces données se décomposent en deux sources distinctes :

1. **Les logs** : répertoriant l'ensemble des interactions (implicites) des utilisateurs sur le site web.
2. **Les données catégorielles (CD)** : qui décrivent l'ensemble des pneumatiques par des informations expertes.

Le tableau 1 récapitule la volumétrie de ces données selon deux périodes P1 (09/01/2017 - 11/04/2018) et P2 (09/01/2017 - 21/06/2019), avec P2 contenant P1. Le tableau 2 propose quelques statistiques sur ces mêmes périodes.

Période	Produits	Utilisateurs	Sessions	Interactions
P1	4117	22463	24350	63279
P2	7640	102613	114359	307231

TAB. 1: Volumétrie des données industrielles par période.

Période	Sessions/Utilisateur	Interactions/Session	Interactions/Sessions/Utilisateur
P1	$1,0955 \pm 0,5152$	$2,5987 \pm 0,8947$	$2,5737 \pm 0,9118$
P2	$1,1172 \pm 0,5453$	$2,6865 \pm 0,9502$	$2,6806 \pm 0,9543$

TAB. 2: Statistiques des données industrielles par période.

Notre objectif est d'évaluer les performances de notre système en fonction des différents types d'informations, de leur encodage et de la durée de la période. Nous serons ainsi en mesure de comparer l'impact de chacun de ces facteurs sur la performance. Pour ce faire, notre système sera décliné en quatre versions afin de l'évaluer selon les informations dont chacune des versions disposera en entrée. Ces versions sont les suivantes :

1. **Baseline** : le système est réduit à l'utilisation seul du 3D CNN avec uniquement les logs comme source de données.
2. **Baseline + CD** : idem au Baseline sauf que les données catégorielles sont cette fois-ci ajoutées.
3. **Hybrid** : l'hybride entre l'eALS et le 3D CNN avec uniquement les logs comme source de données. Dans cette version, nous n'utilisons que les vecteurs latents des utilisateurs (calculés par l'eALS) pour renforcer le 3D CNN.
4. **Hybrid + CD** : idem l'Hybrid mais avec l'ajout des données catégorielles.

La **baseline** nous permet de nous situer par rapport à l'état de l'art grâce aux travaux de Tuan et Phuong (2017); Bourhis et al. (2019). Pour pouvoir comparer équitablement nos quatre versions sur les périodes P1 et P2, dont le nombre de produits va presque du simple au double, nous procédons à un rééquilibrage au niveau de la taille du classement à recommander. En effet, dû à des contraintes techniques nous n'avons pas pu effectuer de recommandation uniquement sur les pneus compatibles avec la requête des utilisateurs. Nous nous sommes donc fixé une taille moyenne de 200 pneus compatibles pour chaque requête (cas déjà assez pessimiste). Sur

cette base, nous voulons proposer un top 5 sur ces 200 pneus. En transposant ces chiffres sur le nombre de pneus total dans nos deux jeux de données, nous arrivons à un top 100 pour P1 et un top 190 pour P2 en respectant les proportions. Cette astuce nous permet de nous rapprocher du cas d'utilisation réel d'un système de recommandation dans le domaine du pneumatique.

Les métriques d'évaluation que nous utilisons sont les suivantes :

1. HR (Hit Rank) qui mesure la capacité du système à donner toutes les solutions pertinentes (varie de 0 à 1 pour le meilleur).

$$HR = \frac{hits}{users} \tag{2}$$

$hits$: le nombre d'utilisateurs dont le produit a bien été recommandé.
$users$: le nombre total d'utilisateurs.

2. NDCG (Normalized Discounted Cumulative Gain) qui mesure la qualité d'un classement (varie de 0 à 1 pour le meilleur).

$$\begin{aligned} DCG_p &= \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} \\ IDCG_p &= \sum_{i=1}^{|rel_p|} \frac{2^{rel_i}}{\log_2(i+1)} \\ NDCG_p &= \frac{DCG_p}{IDCG_p} \end{aligned} \tag{3}$$

p : La position dans le classement pour laquelle on calcule le gain.
rel_i : La pertinence graduée du résultat à la position i.

6 Expérimentations

Le tableau 3 présente les résultats obtenus par notre système suivant les différentes déclinaisons décrites dans la section 5. On y retrouve en gras, le meilleur algorithme par ligne selon la métrique. Chaque valeur est obtenue par la moyenne de 10 expériences de 100 *epochs* chacune, où nous avons fait varier la graine de la descente de gradient stochastique (DGS) du 3D CNN. Notre *DGS* utilise un seul paramètre à savoir le taux d'apprentissage qui est déterminé via une phase de tuning. Cette moyenne est réalisée sur l'epoch obtenant la meilleure moyenne harmonique (HR+NDCG) sur les 10 expériences. L'intervalle de confiance utilisé est l'écart-type et le *Mean HR/NDCG* représente la moyenne des résultats du modèle (colonne).

Période - Encodage	Métrique	Baseline	Baseline + CD	Hybrid	Hybrid + CD
P1 - Character Level	**HR@100**	$0,8077 \pm 0,0056$	$0,6681 \pm 0,033$	$0,8514 \pm 0,037$	$\mathbf{0,8562 \pm 0,0099}$
	NDCG@100	$0,3816 \pm 0,0036$	$0,2269 \pm 0,0328$	$0,3854 \pm 0,049$	$\mathbf{0,4066 \pm 0,0049}$
P1 - Binary	**HR@100**	$0,797 \pm 0,0101$	$0,7089 \pm 0,0232$	$\mathbf{0,8408 \pm 0,0036}$	$0,8305 \pm 0,0423$
	NDCG@100	$\mathbf{0,385 \pm 0,0068}$	$0,2858 \pm 0,0134$	$0,3766 \pm 0,0039$	$0,3602 \pm 0,06$
P2 - Binary	**HR@190**	$0,8571 \pm 0,006$	$0,8692 \pm 0,0054$	$0,8843 \pm 0,0049$	$\mathbf{0,89 \pm 0,0115}$
	NDCG@190	$\mathbf{0,3558 \pm 0,0049}$	$0,3348 \pm 0,0057$	$0,3323 \pm 0,0066$	$0,3433 \pm 0,0104$
	Mean HR	$0,8206 \pm 0,0072$	$0,7487 \pm 0,0205$	$0,8588 \pm 0,0152$	$\mathbf{0,8589 \pm 0,0212}$
	Mean NDCG	$\mathbf{0,3741 \pm 0,0051}$	$0,2825 \pm 0,0173$	$0,3648 \pm 0,0198$	$0,37 \pm 0,0251$

TAB. 3: Résultats des 4 versions de notre système selon l'encodage et la période.

Systèmes hybrides de recommandation.

A partir de ces résultats, nous pouvons extraire les observations suivantes. Premièrement, nous remarquons le gain apporté au Hit Rank par **l'hybridation** et donc l'intérêt de reconstruire les "profils" des utilisateurs via les vecteurs latents de la factorisation de matrices. En effet, parmi nos deux métriques, le HR nous paraît être le plus important à condition d'avoir un NDCG correct. Le gain des hybrides sur le HR, bien que faible, montre donc que notre approche est pertinente. Quand au NDCG, les hybrides semblent équivalents voir très légèrement inférieures aux baselines. A l'inverse, l'utilisation des données catégorielles pour décrire les produits semble réduire les performances (baseline) ou n'apporter aucun gain notable (hybrid). On notera aussi l'augmentation de l'instabilité du système avec les données catégorielles. Deuxièmement, **l'encodage binaire** semble globalement équivalent à l'encodage par caractère. Cela nous permet de réduire grandement la taille des données une fois encodées et ainsi de réduire le temps d'apprentissage et de prédiction du système sans perte d'informations. Enfin, entraîner le système sur une plus **grande période** permet de bien mieux tirer partie des données catégorielles tout en augmentant les performances et la stabilité. Nous proposons également la figure 3 qui présente l'évolution des métriques sur 100 epochs et pour une expérimentation de chacune de nos versions.

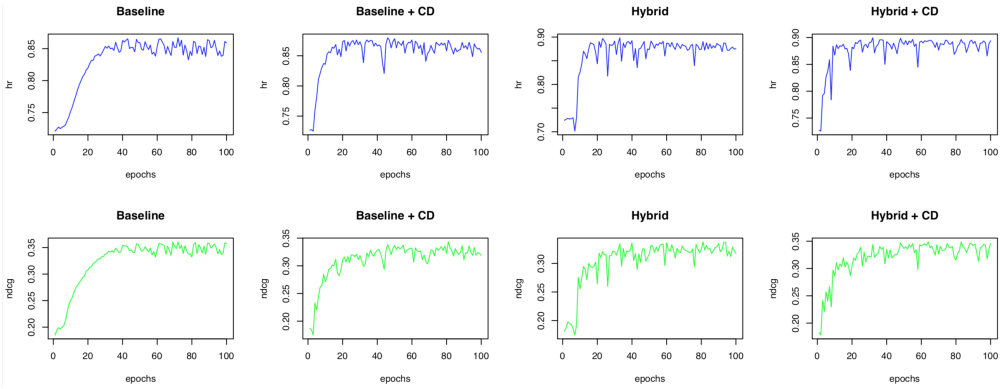

FIG. 3: Évolution du HR et du NDCG sur une expérience pour chacune des versions.

7 Conclusion

Dans ce papier, nous présentons notre système de recommandation, un hybride entre une factorisation de matrices (eALS) et un réseau de neurones convolutif (3D CNN). Ce système permet de s'adapter facilement à tout type de données comme nous l'illustrons avec les données du pneumatique qui sont très spécifiques. Ainsi, l'eALS nous a permis de recréer des profils d'utilisateurs et de produits, afin de les intégrer aux sessions qui sont exploitées par le 3D CNN tout en gardant l'aspect temporel des interactions. Nos expérimentations sur des données réelles montrent l'importance de notre hybridation sur les performances bien que nous n'ayons pas effectué de tests statistiques. Ces expérimentations utilisent une baseline (3D CNN) commune avec le reste de l'état de l'art, qui nous permet de nous situer dans les meilleurs approches. Nos expérimentations valident l'encodage binaire comme viable par rap-

port à l'encodage par caractère (one hot). Elles montrent l'impact de la volumétrie via nos deux périodes de tests et crédibilisent les résultats par l'ajout de données additionnelles à travers des informations expertes sur les produits. Nous prévoyons dans de futurs travaux, de réaliser des expérimentations en intégrant la liste des produits compatibles et d'étudier les approches multi-vues afin d'améliorer notre système de recommandation.

Références

Bourhis, K., K. Benabdeslem, et B. Canitia (2019). Entre factorisation de matrices et apprentissage profond pour la recommandation dans le domaine du pneumatique. In *Extraction et Gestion des Connaissances : Actes de la conférence EGC'2019*. EGC.

Burke, R. (2002). Hybrid recommender systems : Survey and experiments. *User modeling and user-adapted interaction 12*(4), 331–370.

Godard, P. (2017). Rnn-based sequence prediction as an alternative or complement to traditional recommender systems.

Guo, H., R. Tang, Y. Ye, Z. Li, et X. He (2017). Deepfm : a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv :1703.04247*.

He, K., X. Zhang, S. Ren, et J. Sun (2016b). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

He, X., L. Liao, H. Zhang, L. Nie, X. Hu, et T.-S. Chua (2017). Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pp. 173–182. International World Wide Web Conferences Steering Committee.

He, X., H. Zhang, M.-Y. Kan, et T.-S. Chua (2016a). Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 549–558. ACM.

Hidasi, B., A. Karatzoglou, L. Baltrunas, et D. Tikk (2015). Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv :1511.06939*.

Kim, D., C. Park, J. Oh, S. Lee, et H. Yu (2016). Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 233–240. ACM.

Mnih, A. et R. R. Salakhutdinov (2008). Probabilistic matrix factorization. In *Advances in neural information processing systems*, pp. 1257–1264.

Schafer, J. B., D. Frankowski, J. Herlocker, et S. Sen (2007). Collaborative filtering recommender systems. In *The adaptive web*, pp. 291–324. Springer.

Tuan, T. X. et T. M. Phuong (2017). 3d convolutional networks for session-based recommendation with content features. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pp. 138–146. ACM.

Wang, S., L. Cao, et Y. Wang (2019). A survey on session-based recommender systems. *arXiv preprint arXiv :1902.04864*.

Systèmes hybrides de recommandation.

Xue, H.-J., X. Dai, J. Zhang, S. Huang, et J. Chen (2017). Deep matrix factorization models for recommender systems. In *IJCAI*, pp. 3203–3209.

Zhang, S., L. Yao, A. Sun, et Y. Tay (2019). Deep learning based recommender system : A survey and new perspectives. *ACM Computing Surveys (CSUR) 52*(1), 5.

Annexe

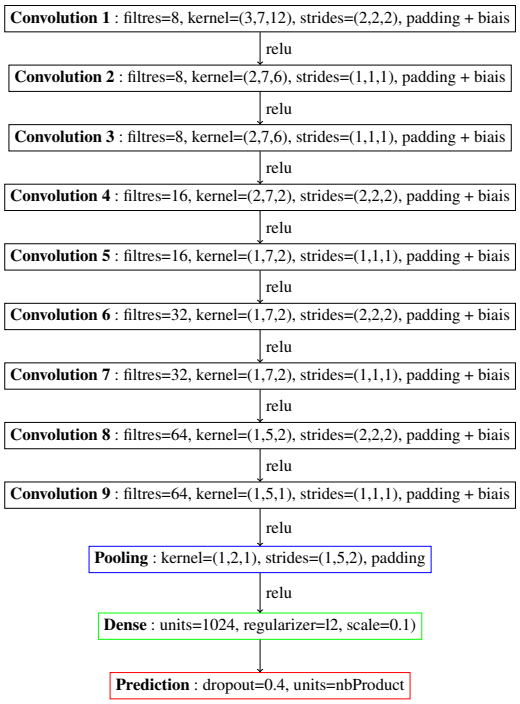

FIG. 4: Architecture 3D CNN - Hybrid + CD.

Summary

Many approaches to recommendation have been used to collect user preferences in order to improve their navigation and transformation rate. In some cases, we do not have explicit information about users or products. The reconstruction of a user or product profiles, based on indirect elements, therefore becomes necessary. Our objective is to exploit factors underlying user-product interactions to enrich our recommendation engine. We propose here a new hybrid approach - matrix factorization and convolutional neural network - that responds to the problem posed by an industrial company in the tire industry. Our algorithm was evaluated on a set of real data extracted from an online comparator. Our study shows that our model is more efficient than state-of-the-art models and it has allowed us to study the impact of the different types of information used on the results.