Recommandation séquentielle à base de séquences fréquentes

Corentin Lonjarret, Marc Plantevit, Céline Robardet, Roch Auburtin

Recommandation séquentielle à base de séquences fréquentes

Corentin Lonjarret[*,***], Marc Plantevit[**]
Céline Robardet[*], Roch Auburtin[***]

[*]INSA Lyon, CNRS, LIRIS UMR5205, F-69621 France
corentin.lonjarret@insa-lyon.fr, celine.robardet@insa-lyon.fr
[**]Université Lyon 1, CNRS, LIRIS UMR5205, F-69622 France
marc.plantevit@liris.cnrs.fr
[***]Visiativ, France
roch.auburtin@visiativ.com

Résumé. La modélisation des préférences utilisateur et de leur dynamique est au cœur de la construction des systèmes de recommandation séquentielle. Les défis résident dans la combinaison réussie de l'historique des utilisateurs et de leurs actions récentes pour fournir des recommandations personnalisées. Les méthodes existantes s'appuient sur des chaînes de Markov d'ordre fixe, limitant la personnalisation. Nous proposons d'utiliser des séquences fréquentes de longueur variable, pour mieux identifier la dynamique séquentielle, et projetons les items dans un espace euclidien en fonction de la préférence utilisateur et de leur historique récent. Une étude empirique sur 13 jeux de données montre que notre méthode surpasse les performances des différentes méthodes de l'état de l'art. De plus, nous pouvons fournir des éclairages sur la recommandation.

1 Introduction

Notamment à travers les sites de vente en ligne ou les plateformes d'intermédiation, les utilisateurs sont de plus en plus confrontés à un nombre de possibilités rendant impossible l'analyse exhaustive de l'ensemble des choix possibles par l'utilisateur. Il est alors devenu indispensable de recourir à un système de recommandation dans ces situations. Dans cet article, nous abordons le problème de recommandation séquentielle dont le but est de prédire la prochaine action d'un utilisateur à partir de sa séquence d'actions passées. Dans la suite une action est assimilée à un item. Pour ce faire, la préférence à long terme de l'utilisateur et sa dynamique à plus court terme sont prises en compte. Considérons tout d'abord les méthodes récentes traitant ce problème.

Modéliser la préférence utilisateur. Les techniques de factorisation de matrices (Koren et Bell. (2011)) permettent de modéliser les interactions entre les utilisateurs et les items en décomposant la matrice d'interaction en un produit de deux matrices de rang k. La prédiction qu'un utilisateur u choisisse l'item i est estimée par le produit scalaire du vecteur de longueur k associé à u, par le vecteur de longueur k associé à i. Cependant, la matrice d'interaction est généralement creuse, ce qui rend la décomposition peu précise. Pour essayer de pallier ce problème, d'autres méthodes comme FISM (Kabbur et al. (2013)) décomposent une matrice

de similarité d'items en deux matrices de rang k. Plus un item i est similaire aux items déjà choisis par l'utilisateur, plus i a des chances d'être recommandé.

Modéliser la dynamique séquentielle. Une autre tendance dans les systèmes de recommandation est de prendre en compte les informations séquentielles présentes dans l'historique des utilisateurs. La dynamique à court terme est généralement modélisée à l'aide de chaînes de Markov. La matrice de transition est décomposée par le produit de deux matrices de rang k. Ainsi, la probabilité d'avoir l'item i sachant que l'item j appartient à l'historique de l'utilisateur est estimée par le produit scalaire des deux matrices.

Modèle unifié. Plusieurs approches récentes cherchent à unifier la préférence utilisateur et la dynamique séquentielle pour obtenir de meilleures performances, comme par exemple la méthode FPMC (Rendle et al. (2010)). Plus récemment, Fossil (He et McAuley (2016)) propose d'associer une approche de similarité entre items comme FISM avec des chaînes de Markov d'ordre L. PRME (Feng et al. (2015)) a amélioré FPMC et Fossil en remplaçant le produit scalaire par des distances Euclidiennes. En effet, les méthodes utilisant des distances permettent une meilleure généralisation. Dernièrement, TransRec (He et al. (2017)) unifie la préférence utilisateur et la dynamique séquentielle en utilisant des translations dans un espace Euclidien.

Ces méthodes combinent la dynamique de long et court terme en n'utilisant uniquement des chaînes de Markov d'ordre fixe. Pour pallier ce problème, nous proposons une nouvelle méthode, **REBUS**, qui utilise des séquences fréquentes pour identifier les items les plus pertinents des historiques des utilisateurs. Ces items permettent de mieux capturer la dynamique séquentielle. Notre contribution se résume au développement d'un nouveau modèle, **REBUS**, qui unifie la préférence utilisateur et la dynamique séquentielle en les plongeant dans un même espace euclidien. De plus, l'ordre personnalisé des chaînes Markov est déterminé grâce à l'utilisation de séquences fréquentes. La figure 1 résume le fonctionnement de **REBUS**. Dans une étude empirique sur 13 jeux de données, nous démontrerons que **REBUS** surpasse l'état de l'art des systèmes de recommandation séquentielle. Les données et le code sont disponibles [1].

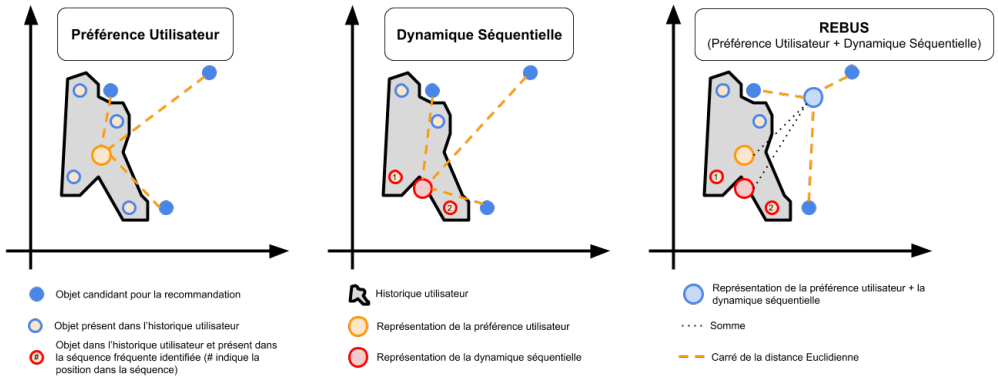

FIG. 1 – ***La préférence utilisateur*** *: plongement des items de l'historique de l'utilisateur ;* ***La dynamique séquentielle*** *: plongement des items de la séquence fréquente ;* **REBUS** *recommande l'item le plus proche de la somme des deux plongements.*

1. https://tinyurl.com/yaxve89j

2 Le modèle REBUS

2.1 Evaluer la dynamique séquentielle par un contexte personnel

Le point de vue défendu dans cet article, est que l'utilisation de séquences plus longues et variées permet de mieux représenter la dynamique séquentielle et donc d'améliorer la recommandation. C'est pourquoi nous proposons d'utiliser des chaînes de Markov d'ordre variable, contrairement aux méthodes existantes qui utilisent des chaînes d'ordre fixe. L'approche utilisée pour trouver le contexte le plus adapté pour un utilisateur u au pas de temps t, comporte deux étapes : (1) La construction d'un ensemble de contextes commun pour tous les utilisateurs et (2) l'identification du contexte le plus adapté à un utilisateur u au pas de temps t.

Identifier des contextes pertinents. Les contextes pris en compte sont ceux qui apparaissent dans au moins ***minCount*** séquences d'utilisateurs et ont une taille inférieure ou égale à ***L***. Nous proposons de les identifier à l'aide d'un extracteur de sous-chaînes fréquentes. L'ensemble généré est appelé F.

Personnalisé le contexte de u au pas de temps t. L'objectif est de déterminer la sous-chaine $m_{s_u^{[1,t]}}$ la mieux adaptée à u au temps t. Pour ce faire, on sélectionne la séquence de F la plus longue et récente présente dans $s_u^{[1,t]}$, la sous-séquence de l'utilisateur u tronquée en position t. Dans le cas où aucune sous-chaîne ne peut être sélectionnée, nous définissons $m_{s_u^{[1,t]}} = i^\star$, où $i^\star$ est un item fictif.

2.2 Le modèle proposé

REBUS est un modèle qui plonge les items dans un espace euclidien de telle sorte que la projection d'un item est influencée par la préférence utilisateur et la dynamique séquentielle. L'utilisation d'une distance euclidienne possèdent deux avantages : (1) cela permet d'avoir une meilleure généralisation car les distances conservent l'inégalité triangulaire, (2) cela permet d'effectuer un seul calcul de distance tout en unifiant les dynamiques à long et court terme. L'objectif est d'apprendre un vecteur P_i pour chaque item i de I de telle sorte que la prédiction $\hat{p}_{u,i,t}$ que u choisisse i au temps t soit aussi proche que possible que ce qui est observé dans les données. Le modèle est alors défini par :

$$\widehat{p}_{u,i,t} \propto -(\beta_i + ||\Big(\frac{1}{|I_{s_u^{[1,t]}} \setminus \{i\}|^\alpha} \sum_{j \in I_{s_u^{[1,t]}} \setminus \{i\}} P_j + \sum_{r=0}^{|m_{s_u^{[1,t]}}|} \eta_r P_{m^r_{s_u^{[1,t]}}}\Big) - P_i||_2^2)$$

où (1) β_i est un terme de biais, (2) le premier terme, la dynamique à long terme, est la moyenne des vecteurs associés aux items j de l'historique de l'utilisateur, (3) le second terme modélise la dynamique à court terme à l'aide du paramètre η_r qui augmente avec r, le rang de l'item dans la séquence représentant l'historique, pour donner plus d'importance aux items récents [2].

REBUS apprend les paramètres **P** et β [3]. Le critère d'optimisation BPR (Rendle et al. (2009)) permet d'ordonner l'item vérité plus haut que tous les autres items. Il optimise les paramètres θ du modèle afin de maximiser la probabilité d'avoir i classé plus haut que j pour l'utilisateur u

2. η_r suit l'exponentielle normalisée de 1 moins la loi de répartition de Weilbull avec $x = 2$ et $y = 7$.

3. Les hyper-paramètres, tels que α, L, **minCount** et les autres paramètres liés à l'apprentissage sont déterminés à l'aide d'une recherche sur grille.

au pas de temps t, lorsque $i = s_u^t$. En supposant l'indépendance entre les items, les utilisateurs et les pas de temps, cela revient à estimer les paramètres du modèle par le maximum a posteriori (MAP). Les paramètres sont appris à l'aide d'une descente de gradient stochastique (SGD) qui a prouvé son efficacité (He et al. (2017); Feng et al. (2015)).

3 Expériences

	Datasets	#U	#I	$\sum$#seq.	avg (#seq.)	avg (#i occ.)
	Epinions	5015	8335	26932	5.3703	3.2312
	Foursquare	43110	13335	306553	7.1109	22.9886
	Visiativ	1398	590	16417	11.7432	27.8254
Amazon	Ama-Auto	34316	40287	183573	5.3495	4.5566
	Ama-Cell	68330	60083	429231	6.2817	7.1440
	Ama-Office	16716	22357	128070	7.6615	5.7284
	Ama-Toy	57617	69147	410920	7.1319	5.9427
	Ama-Game	31013	23715	287107	9.2576	12.1066
MovieLens	ML-5	6040	2848	30175	4.9959	10.5952
	ML-10	6040	3114	59610	9.8692	19.1426
	ML-20	6040	3324	111059	18.3873	33.4113
	ML30	6040	3391	152160	25.1921	44.8717
	ML50	6040	3467	215676	35.7079	62.2082

TAB. 1 – *Caractéristiques des jeux de données.*

Jeux de données et objectifs. Pour évaluer les performances de **REBUS**, nous avons utilisé les jeux de données présentés dans le tableau 1 [4]. Nous voulons répondre aux questions suivantes : Quelles sont les performances de **REBUS** comparé aux approches de l'état de l'art ? Est-ce que **REBUS** tire un bénéfice des contextes personnalisés ? Est-ce **REBUS** permet de mieux comprendre la recommandation séquentielle ?

Méthodes de l'état de l'art. Nous avons utilisé comme point de comparaison 7 méthodes de l'état de l'art : Popularity (**POP**) qui recommande les items les plus populaires ; Bayesian Personalized Ranking (**BPR-MF**) (Rendle et al. (2009)) qui utilise la factorisation de matrices ; Factorized Markov Chains (**FMC**) (Rendle et al. (2010)) qui factorise la matrice item-item de transition ; Factorized Personalized Markov Chains (**FPMC**) (Rendle et al. (2010)) qui prend en compte la préférence utilisateur et la dynamique séquentielle avec chaînes de Markov d'ordre 1 ; Personalized Ranking Metric Embedding (**PRME**) (Feng et al. (2015)) qui représente la préférence utilisateur et la dynamique séquentielle par la somme de deux distances Euclidiennes ; Factorized Sequential Prediction with Item Similarity Models (**Fossil**) (He et McAuley (2016)) qui unifie des chaînes de Markov d'ordre L avec FISM ; Translation-based Recommendation (**TransRec**) (He et al. (2017)) qui unifie la préférence utilisateur et la dynamique séquentielle avec des translations dans un espace euclidien. Nous évaluons notre modèle **REBUS** avec deux configurations : (1) avec la dynamique séquentielle basée sur les sous-chaînes fréquentes, (2) avec la dynamique séquentielle basée sur les chaînes de Markov d'ordre 1, notée $\textbf{REBUS}_{1MC}$.

Protocole expérimental. Pour chaque jeu de données, nous avons séparé les séquences des utilisateurs en 3 parties : (1) l'item le plus récent qui sera utilisé pour évaluer les méthodes et

4. `http://epinions.com/`, `http://grouplens.org/datasets/movielens/1m/`, `https://foursquare.com/`

appelé *l'item vérité*, (2) le deuxième item le plus récent qui sera utilisé pour la validation des méthodes lors de la phase d'apprentissage et (3) tous les autres items qui seront utilisés pour entraîner les méthodes. La précision de l'approche est mesurée par l'**AUC** et le **HIT50**.

Études des performances. Les performances des différentes méthodes pour chaque jeu de données sont résumées dans le tableau 2. Nous pouvons observer que les résultats de la métrique AUC de **REBUS** surpassent les autres approches sur la plupart des jeux de données. **REBUS** obtient également de bonnes performances sur la métrique HIT50 avec un rang moyen de 3.2 sur 9 méthodes. PRME est très performant sur les jeux de données denses tels que ML30 et ML50. Cependant, il montre ses limites sur les jeux de données éparses. Cela permet de conclure qu'avoir des vecteurs latents indépendants n'est pas un avantage sur des jeux de données éparses. En utilisant des chaînes de Markov d'ordre personnalisé, nous remarquons que cela permet d'obtenir de meilleurs résultats par rapport à l'utilisation des chaînes de Markov d'ordre 1 (c.à.d. **REBUS**$_{1MC}$).

Models	Metric	Epinions	Foursq	Visiativ	Auto	Cell	Office	Toy	Video	Avg (Amazon)	ML-5	ML-10	ML-20	ML-30	ML-50	Avg(ML)	Avg(All)
Pop	AUC	0.4575	0.9169	0.7864	0.5870	0.6959	0.6428	0.6240	0.7497	0.6599	0.7352	0.7722	0.7919	0.7981	0.8032	0.7801	0.7201
	HIT50	3.42%	55.65%	48.24%	3.84%	4.43%	1.66%	1.69%	5.17%	3.36%	12.65%	12.93%	12.82%	12.72%	13.13%	12.85%	14.49%
BPRMF	AUC	0.5359	0.9527	0.8384	0.6342	0.7611	0.7054	0.7280	0.8562	0.7370	0.7817	0.8309	0.8446	0.8514	0.8576	0.8332	0.7829
	HIT50	3.58%	66.65%	56.85%	3.80%	5.49%	4.33%	3.65%	12.47%	5.95%	17.80%	18.43%	17.39%	17.06%	16.54%	17.44%	18.77%
FMC	AUC	0.5476	0.9471	0.8341	0.6442	0.7548	0.6865	0.6943	0.8423	0.7244	0.7234	0.8012	0.8341	0.8441	0.8550	0.8116	0.7699
	HIT50	2.88%	63.47%	57.13%	2.49%	7.14%	3.02%	4.30%	14.30%	6.25%	14.44%	18.99%	20.55%	21.49%	22.22%	19.54%	19.42%
FPMC	AUC	0.5518	0.9480	0.8247	0.6415	0.7376	0.6859	0.7194	0.8524	0.7274	0.7405	0.8286	0.8476	0.8654	0.8802	0.8325	0.7787
	HIT50	2.93%	64.64%	55.34%	2.24%	2.81%	2.97%	4.42%	11.95%	4.88%	12.20%	19.32%	18.25%	25.83%	26.89%	20.50%	19.22%
PRME	AUC	0.6138	0.9604	0.8618	0.6523	0.7987	0.7155	0.7411	0.8763	0.7568	0.7761	0.8354	0.8681	**0.8786**	**0.8884**	0.8493	0.8051
	HIT50	2.88%	65.94%	64.01%	3.77%	7.03%	6.43%	5.25%	**16.29%**	7.75%	16.81%	23.48%	**26.49%**	**27.84%**	**28.18%**	**24.56%**	**22.65%**
Fossil	AUC	0.5974	0.9607	0.8429	0.6887	0.7980	0.7219	0.7620	0.8776	0.7696	0.7916	0.8478	0.8624	0.8677	0.8699	0.8479	0.8068
	HIT50	3.51%	63.21%	55.63%	5.21%	7.50%	5.44%	4.77%	13.87%	7.36%	19.01%	23.36%	22.80%	22.27%	21.64%	21.82%	20.63%
TransRec	AUC	0.6030	0.9632	**0.8649**	0.6767	0.7997	0.7230	0.7478	0.8755	0.7646	0.7856	0.8430	0.8645	0.8689	0.8743	0.8473	0.8069
L1	HIT50	3.02%	65.99%	**64.23%**	5.14%	7.88%	**6.78%**	4.80%	15.39%	8.00%	20.30%	23.60%	23.86%	24.08%	24.26%	23.22%	22.26%
TransRec	AUC	0.6084	0.9621	0.8639	0.6902	0.8049	0.7258	0.7646	0.8844	0.7740	0.7936	0.8488	**0.8711**	0.8750	0.8791	0.8535	0.8132
L2	HIT50	**5.02%**	67.15%	63.08%	**5.56%**	**8.99%**	5.12%	**5.60%**	15.98%	**8.25%**	20.85%	**24.67%**	24.09%	24.42%	23.94%	23.60%	**22.65%**
REBUS	AUC	0.6350	0.9676	0.8630	**0.7108**	0.8266	**0.7521**	**0.7775**	**0.8888**	**0.7912**	**0.8030**	**0.8510**	0.8682	0.8731	0.8757	**0.8542**	**0.8225**
	HIT50	4.83%	68.79%	63.87%	4.57%	8.44%	6.35%	4.46%	15.91%	7.95%	**22.04%**	23.07%	24.24%	22.44%	21.46%	22.65%	22.34%
REBUS$_{1MC}$	AUC	**0.6392**	**0.9684**	0.8643	0.7101	**0.8271**	0.7441	0.7765	0.8876	0.7891	0.8026	0.8501	0.8671	0.8730	0.8768	0.8539	0.8221
	HIT50	3.97%	**69.17%**	63.08%	5.34%	8.90%	6.47%	4.76%	15.38%	8.17%	21.20%	22.39%	24.03%	22.87%	21.54%	22.40%	22.24%
Rang	AUC	1	1	2	1	1	1	1	1	1	1	1	2	3	4	2.2	1.5
	HIT50	2	1	3	2	2	2	5	3	2.8	1	5	2	5	7	4.2	3.2
Amélioration	AUC	4.14%	0.54%	-0.07%	2.98%	2.76%	3.62%	1.69%	0.50%	2.22%	1.12%	0.26%	-0.33%	-0.63%	-1.31%	0.08%	1.14%
	HIT50	-3.78%	3.01%	-0.56%	-3.96%	-1.00%	-4.57%	-15.00%	-2.33%	-0.97%	3.67%	-6.49%	-8.49%	-17.85%	-23.56%	-7.78%	-1.37%

TAB. 2 – *AUC et HIT50 pour les différentes expérimentations. La ligne Amélioration montre les gains/pertes de* **REBUS** *comparé à la meilleure des autres méthodes (en gras).*

Exemples de recommandations. La figure 2 montre quelques exemples de recommandation de notre approche sur les jeux de données Amazon-Games et Amazon-Office. **REBUS** capture la dynamique séquentielle et recommande des items qui sont similaires à l'item vérité. Par exemple, pour le premier utilisateur, **REBUS** recommande *Final Fantasy X-2* car l'utilisateur avait acheté les éditions précédentes de *Final Fantasy* (Les carrés rouges autour des items). L'item vérité, *The Legend of Dragoon*, est un jeu similaire. On voit dans cet exemple que la dynamique séquentielle est capturée par une séquence avec 4 items et un joker. Les 2 derniers exemples utilisent des séquences compactes de 2 items.

4 Conclusion

Nous avons proposé une nouvelle méthode **REBUS** utilisant des distances dans un espace Euclidien pour tenter de régler le problème de recommandation séquentielle. Notre approche utilise des chaînes de Markov d'ordre personnalisé grâce à l'exploitation de sous-chaines fréquentes. **REBUS** apprend une représentation de la préférence utilisateur et la dynamique sé-

FIG. 2 – *Exemples de recommandations pour les jeux de données Amazon Games (ligne [1]) et Office (lignes [2-3]). Les carrés rouges sont les items présents dans* $m_{s_u^{[1,t]}}$.

quentielle dans un même espace Euclidien. Nous avons effectué des expériences sur 13 jeux de données et démontré que **REBUS** surpasse les approches présentes dans l'état de l'art.

Références

Feng, S., X. Li, Y. Zeng, G. Cong, Y. M. Chee, et Q. Yuan (2015). Personalized ranking metric embedding for next new POI recommendation. *IJCAI*.

He, R., W. Kang, et J. McAuley (2017). Translation-based Recommendation. *RecSys*.

He, R. et J. McAuley (2016). Fusing similarity models with markov chains for sparse sequential recommendation. *ICDM*.

Kabbur, S., X. Ning, et G. Karypis. (2013). FISM: factored item similarity models for top-n recommender systems. *SIGKDD*.

Koren, Y. et R. Bell. (2011). Advances in collaborative filtering. *in Recommender Systems Handbook*.

Rendle, S., C. Freudenthaler, Z. Gantner, et L. Schmidt-Thieme. (2009). BPR: Bayesian personalized ranking from implicit feedback. *Uncertainty in Artificial Intelligence*.

Rendle, S., C. Freudenthaler, et L. Schmidt-Thieme (2010). Factorizing personalized markov chains for next-basket recommendation. *WWW*.

Summary

Modeling user preferences and user dynamics is of greatest importance to build efficient recommender systems. Existing methods capture the sequential dynamics of a user using fixed-order Markov chains. We propose to use frequent sequences to identify the important part of user history and use a unified metric model to embed items based on user preferences and dynamics. Experiments demonstrate the advantages of this approach.