Iterative Memory Network for Long Sequential User Behavior Modeling in Recommender Systems

Qianying Lin Wen-Ji Zhou Yanshi Wang Qing Da Qing-Guo Chen Bing Wang

GIANG Cécile - KHALFAT Célina - ZHUANG Pascal

Introduction

Dans le cadre des systèmes de recommandation modernes, la session d'un utilisateur est modélisée par une séquence des items avec lesquels il a interagit. L'enjeu est ainsi de prédire des items pertinents à lui recommander à partir de son historique d'activités. Cette pertinence est souvent mesurée par le taux de clics (CTR). Cependant, plusieurs problèmes se posent avec les modèles de recommandation séquentiels courants:

- il est difficile de capturer les intérêts utilisateur sur le long terme *(phénomène du fast forgetting)*
- pour les modèles basés sur le mécanisme de self-attention, la complexité est quadratique par rapport à la taille des séquences d'entrée
- pour ceux basés sur le mécanisme de target attention, les dépendances intra-séquences ne sont pas capturées

Le papier présente une nouvelle architecture appelée **Iterative Memory Network (IMN)**, qui permet de répondre aux limites citées, en introduisant le mécanisme de *Multi-way Attention*, ainsi qu'une mémoire permettant de modéliser les dépendances target-séquence et intra-séquences.

Formalisation

Une session utilisateur est modélisée par une séquence des items avec lesquels il a interagit, et la target correspond à un item candidat pour la prochaine interaction. L'objectif de notre modèle est alors de prédire si l'utilisateur cliquera sur l'item target. Nous présentons ci-dessous les différents blocs du modèle :

- la couche *Encoder Layer* convertit la séquence des items utilisateur, leur temps, leur position et l'item target dans un vecteur d'embedding
- le mécanisme de *Multi-Way Attention* et de *Memory Update* permettent de calculer à la fois les attentions entre la séquence et la target, ainsi qu'entre les différents items de la séquence
- le bloc de *Memory Enhancement* permet d'améliorer la mémoire avec l'item target

Encoder Layer

Les items d'une séquence utilisateur ainsi que l'item target sont projetés dans un espace de représentation *(embedding)*. L'information temporelle est importante afin de déceler des tendances. Les positions dans la séquence permettent de garder une notion d'ordre entre les items.

Pour le j^{ieme} item d'une séquence, dont l'embedding est e_i, le temps et la position sont encodés dans des espaces de même dimension que e_i (respectivement e_t et e_p).
Les embeddings sont ensuite sommés terme à terme et la représentation du j^{ieme} item dans la séquence devient alors :

$$e^j = e_i \oplus e_t \oplus e_p$$

Multi-Way Attention

Les différentes similarités entre la séquence, la target et la mémoire sont capturées à l'aide de la distance euclidienne (notée —) et de la distance d'Hadamard (notée ∘).

Les similarités sont concaténées de la manière suivante :

$$\alpha(e, v, m) = [e - m, e - v, e \circ m, e \circ v]$$

où e est l'embedding pour la séquence d'items , m est l'embedding pour la mémoire et v l'embedding pour l'item target.

Iterative Memory Update

Ce module permet de mettre en place une mémoire sur les dépendances target-séquence ainsi que sur les dépendances intra-séquences. Cette mémoire, valant v_T à l'initialisation, est mise à jour à chaque itération en appliquant un GRU à l'item target, avec comme état caché v^t qui est le produit entre le vecteur en sortie du bloc précédent modélisant l'intérêt utilisateur w et les embeddings de chaque item de la séquence. La quantité v^t permet de retrouver dans la mémoire les informations pertinentes.

Algorithm 1 Iterative Memory Update Algorithm

$m^0 \leftarrow v_T$
for t in 1...n **do**
  $v_u^t \leftarrow \sum_{j=1}^{L} = w_j e^j$
  $m^t \leftarrow GRU(v_u^t, m^{t-1})$
end for
return m^t

En sortie de ce bloc, nous obtenons ainsi une représentation des utilisateurs modélisée par une mémoire sur les dépendances target-séquence et intra-séquence.

Memory Enhancement Module

Ce bloc permet de renforcer la mémoire m^N (calculée après N itérations de l'étape précédente) avec l'item target v^T. Cette mémoire est mise à jour itérativement à l'aide d'une transformation linéaire W^u selon la formule suivante:

$$u^t = GRU([W^u u^{t-1}, v_T], u^{t-1}) \text{ avec comme initialisation } u^0 = m^N$$

où u_t est la représentation de l'utilisateur après t mises à jour.

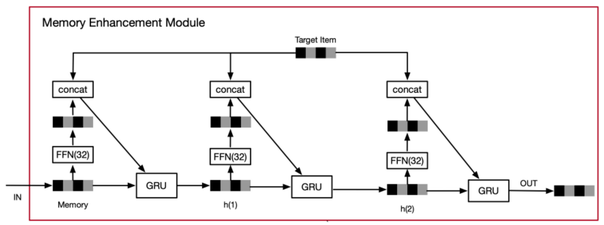

Figure 1. Memory Enhancement Module

Architecture

Figure 2. Architecture globale de IMN

Données et évaluation

- **Dataset :** Amazon Books Ratings (séquences d'items de taille L avec un décalage d)
- **Loss :** Cross-Entropy loss
- **Métrique d'évaluation :** AUC
- **Hyper-paramètres :** Taille d'embedding (10), Head (1), Memory iterations (3)
- **Optimisation :** Adam, Batch size (512), Learning rate (1e-5)
- **Régularisation :** BatchNorm, Dropout (0.2), Norme L2 (1e-6)

Résultats

Model	AUC	Impr
Youtube DNN	0.8374	0.00
DIN	0.8516	1.42
DIEN	0.8550	1.76
SASRec	0.8214	-1.60
MIMN	0.8523	1.49
UBR4CTR	0.8570	1.96
IMN 2P	0.8498	1.24
IMN 3P	0.8672	2.98
IMN 3P+	0.8692	3.18

Table 1. Résultats de l'article

Taille séquences \ Memory Iterations	0P+	3P+	10P+
40	0.672	0.964	0.964
100	0.658	0.937	0.939
500	0.654	0.681	0.683

Table 2. Nos résultats

Conclusion

Les expérimentations :

- le mécanisme de *Memory Update* permet une amélioration des performances
- au-delà de 3 itérations du module *Memory Update*, les performances stagnent
- plus les séquences sont longues, moins les performances sont bonnes

Limites de l'article :

- manque d'informations sur la création du dataset et l'entraînement du modèle
- pas d'information sur la loss utilisée et la taille des séquences pour les résultats

Référence

[1] Yanshi Wang Qing Da Qing-Guo Chen Bing Wan Qianying Lin, Wen-Ji Zhou. Iterative memory network for long sequential user behavior modeling in recommender systems. *ICLR*, 2022.