UNIVERSITÉ DE YAOUNDÉ I
SAPIENTIA - COLLATIVA - COGNITIO
DEPUIS 1993

SORBONNE UNIVERSITÉ
CRÉATEURS DE FUTURS
DEPUIS 1257

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE YAOUNDÉ I ET DE SORBONNE UNIVERSITÉ

Spécialité **INFORMATIQUE**

Centre de Recherche et de Formation Doctorale en Sciences, Technologies et Géosciences (Yaoundé)
École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par
NZEKON NZEKO'O Armel Jacques

Pour obtenir le grade de
DOCTEUR/Ph.D de l'UNIVERSITÉ DE YAOUNDÉ I
et de DOCTEUR de SORBONNE UNIVERSITÉ

SYSTÈME DE RECOMMANDATION AVEC DYNAMIQUE TEMPORELLE BASÉE SUR LES FLOTS DE LIENS

Soutenue publiquement le 09 décembre 2019 devant le jury composé de :

Rapporteurs :	Emmanuel VIENNET	Professeur, Université Paris 13
	Jean-Loup GUILLAUME	Professeur, Université de la Rochelle
Examinateurs :	Clémence Magnien	Directrice de recherches, CNRS
	Marcel FOUDA NDJODO	Professeur, Université de Yaoundé I
	René NDOUNDAM	Associate Professor, Université de Yaoundé I
	Paulin MELATAGIA YONTA	Senior lecturer, Université de Yaoundé I
Directeurs :	Maurice TCHUENTE	Professeur, Université de Yaoundé I
	Matthieu LATAPY	Directeur de recherches, CNRS

RÉPUBLIQUE DU CAMEROUN
PAIX-TRAVAIL-PATRIE
****
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
****
UNIVERSITÉ DE YAOUNDÉ I
****
CENTRE DE RECHERCHE ET DE FORMATION DOCTORALE EN SCIENCES, TE*CHNOLOGIES* ET G*EOSCIENCES*
****

REPUBLIC OF CAMEROON
PEACE-WORK-FATHERLAND
****
MINISTRY OF HIGHER EDUCATION
****
THE UNIVERSITY OF YAOUNDE I
****
POSTGRADUATE SCHOOL OF SCIENCE, TECHNOLOGY AND GEO-SCIENCES
****

DÉPARTEMENT D'INFORMATIQUE
DEPARTMENT OF COMPUTER SCIENCE

ATTESTATION DE CORRECTION DE LA THÈSE DE DOCTORAT/PhD

Nous soussignés, Pr. **FOUDA NDJODO Marcel**, Pr. **TCHUENTE Maurice**, membres du jury de la thèse de Doctorat/PhD présentée par Monsieur **NZEKON NZEKO'O Armel Jacques**, Matricule **09U0272**, intitulée: « **Système de recommandation avec dynamique temporelle basée sur les flots de liens** » et soutenue le **09/12/2019** en vue de l'obtention du diplôme de **Doctorat/PhD en Informatique**, attestons que toutes les corrections demandées par le jury de soutenance en vue de l'amélioration de ce travail, ont été effectuées.

En foi de quoi la présente attestation lui est délivrée pour servir et valoir ce que de droit.

Examinateur

Pr. FOUDA NDJODO Marcel

Rapporteur

Pr. TCHUENTE Maurice

UNIVERSITÉ DE YAOUNDÉ I
SAPIENTIA - COLLATIVA - COGNITIO
DEPUIS 1993

SORBONNE UNIVERSITÉ
CRÉATEURS DE FUTURS
DEPUIS 1257

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE YAOUNDÉ I ET DE SORBONNE UNIVERSITÉ

Spécialité **INFORMATIQUE**

Centre de Recherche et de Formation Doctorale en Sciences, Technologies et Géosciences / *Laboratoire d'Informatique et Applications* (Yaoundé)

École doctorale Informatique, Télécommunications et Électronique / *Laboratoire d'Informatique de Paris 6* (Paris)

SYSTEME DE RECOMMANDATION AVEC DYNAMIQUE TEMPORELLE BASEE SUR LES FLOTS DE LIENS

THÈSE

Pour obtenir le grade de
DOCTEUR / Ph.D de l'UNIVERSITÉ DE YAOUNDÉ I
et de DOCTEUR de SORBONNE UNIVERSITÉ

Présentée par
NZEKON NZEKO'O Armel Jacques
MSc Informatique

Directeurs : TCHUENTE Maurice Professeur, Université de Yaoundé I
LATAPY Matthieu Directeur de recherches, CNRS

Année académique 2018/2019

Résumé

La recommandation des produits appropriés aux clients est cruciale dans de nombreuses plateformes de e-commerce qui proposent un grand nombre de produits. Les systèmes de recommandation sont une solution favorite pour la réalisation de cette tâche. La majorité des recherches de ce domaine reposent sur des notes explicites que les utilisateurs attribuent aux produits, alors que la plupart du temps ces notes ne sont pas disponibles en quantité suffisante. Il est donc important que les systèmes de recommandation utilisent les données implicites que sont des flots de liens représentant les relations entre les utilisateurs et les produits, c'est-à-dire l'historique de navigation, des achats et de streaming. C'est ce type de données implicites que nous exploitons.

Une approche populaire des systèmes de recommandation consiste, pour un entier N donné, à proposer les N produits les plus pertinents pour chaque utilisateur : on parle de recommandation top-N. Pour ce faire, bon nombre de travaux reposent sur des informations telles que les caractéristiques des produits, les goûts et préférences antérieurs des utilisateurs et les relations de confiance entre ces derniers. Cependant, ces systèmes n'utilisent qu'un ou deux types d'information simultanément, ce qui peut limiter leurs performances car l'intérêt qu'un utilisateur a pour un produit peut à la fois dépendre de plus de deux types d'information.

Pour remédier à cette limite, nous faisons trois propositions dans le cadre des graphes de recommandation. La première est une extension du Session-based Temporal Graph (STG) introduit par Xiang et al., et qui est un graphe dynamique combinant les préférences à long et à court terme des utilisateurs, ce qui permet de mieux capturer la dynamique des préférences de ces derniers. STG ne tient pas compte des caractéristiques des produits et ne fait aucune différence de poids entre les arêtes les plus récentes et les arêtes les plus anciennes. Le nouveau graphe proposé, Time-weight content-based STG contourne les limites du STG en y intégrant un nouveau type de nœud pour les caractéristiques des produits et une pénalisation des arêtes les plus anciennes.

La seconde contribution est un système de recommandation basé sur l'utilisation de Link Stream Graph (LSG). Ce graphe est inspiré d'une représentation des flots de liens et a la particularité de considérer le temps de manière continue contrairement aux autres graphes de la littérature, qui soit ignore la dimension temporelle comme le graphe biparti classique (BIP), soit considère le temps de manière discontinue avec un découpage du temps en tranches comme STG.

La troisième contribution de cette thèse est GraFC2T2, un framework qui intègre des graphes de recommandation et les enrichit par des informations sur les caractéristiques des produits, la dynamique des préférences des utilisateurs et la confiance entre ces derniers. Les mises en œuvre de ces trois contributions sur les jeux de données de CiteUlike, Delicious, Last.fm, Ponpare, Epinions et Ciao, confirment leur pertinence.

Abstract

Recommending appropriate items to users is crucial in many e-commerce platforms that propose a large number of items to users. Recommender systems are one favorite solution for this task. Most research in this area is based on explicit ratings that users give to items, while most of the time, ratings are not available in sufficient quantities. In these situations, it is important that recommender systems use implicit data which are link stream connecting users to items while maintaining timestamps *i.e.* users browsing, purchases and streaming history. We exploit this type of implicit data in this thesis.

One common approach consists in selecting the N most relevant items to each user, for a given N, which is called top-N recommendation. To do so, recommender systems rely on various kinds of information, like content-based features of items, past interest of users for items and trust between users. However, they often use only one or two such pieces of information simultaneously, which can limit their performance because user's interest for an item can depend on more than two types of side information.

To address this limitation, we make three contributions in the field of graph-based recommender systems. The first one is an extension of the Session-based Temporal Graph (STG) introduced by Xiang et al., which is a dynamic graph combining long-term and short-term preferences in order to better capture user preferences over time. STG ignores content-based features of items, and make no difference between the weight of newer edges and older edges. The new proposed graph Time-weight Content-based STG addresses STG limitations by adding a new node type for content-based features of items, and a penalization of older edges.

The second contribution is the Link Stream Graph (LSG) for temporal recommendations. This graph is inspired by a formal representation of link stream, and has the particularity to consider time in a continuous way unlike others state-of-the-art graphs, which ignore the temporal dimension like the classical bipartite graph (BIP), or consider time discontinuously like STG where time is divided into slices.

The third contribution in this thesis is GraFC2T2, a general graph-based framework for top-N recommendation. This framework integrates basic recommender graphs, and enriches them with content-based features of items, users' preferences temporal dynamics, and trust relationships between them. Implementations of these three contributions on CiteUlike, Delicious, Last.fm, Ponpare, Epinions and Ciao datasets confirm their relevance.

Dédicace

A ma mère, Wandji Suzanne
cette femme si patiente et douce,
cette femme toujours pleine de bon conseils,
cet exemple vivant de l'expression du respect d'autrui,
une femme comme je n'en n'ai plus rencontrée,
cette dame qui n'a jamais cessé de me pousser vers l'avant,
j'espère que pour elle, cette thèse sera comme l'accomplissement d'un rêve,
que cette thèse essuiera toutes les peines après le décès de son mari chéri,
cet événement tragique qui lui laissa un nourrisson de 3 mois
ce nourrisson qui lui dédicace ce manuscrit.

A la mémoire de :

mon papa, Nzekon Augustin
dont le travail et la défense des valeurs amicales et fraternelles a porté du fruit même au delà de sa vie, et dont je suis l'un des bénéficiaires directs.

ma grand mère, Yonkeu Anne
dont le seul sourire suffisait comme motivation pour que je me surpasse, et dont l'affection était si tendre et doux dans mon cœur.

papa Ntenda
dont l'éducation reste toujours gravée dans mon être et pour qui j'aimerai toujours chanter et danser "le coq chante".

Remerciements

Je débute cette section, en adressant un sincère merci au professeur Maurice Tchuente, qui a eu la patience et la bonté de m'encadrer depuis le Master 2. En planifiant et en organisant mon chemin à l'avance, il s'est arrangé à ce que mes années de thèse se déroulent dans un cadre très proche de l'idéal. De même, je remercie le directeur de recherche Matthieu Latapy qui n'a jamais cessé de contribuer à aplanir mon chemin, afin que j'y passe sans embûches. Mes contacts avec lui m'ont toujours permis de découvrir une nouvelle réalité dont le souvenir est une source d'inspiration pour une carrière de chercheur. Je remercie ces deux hommes dont les succès ne sont plus à démontrer, pour leur encadrement qui a rendu possible ce manuscrit de thèse.

Je remercie le laboratoire UMMISCO IRD-SU et le Centre d'Excellence Africain en Technologies de l'Information et de la Communication (CETIC) qui ont soutenu financièrement toutes mes années de thèse. Un remerciement particulier aux membres de l'équipe UMMISCO, notamment le directeur Jean-Daniel Zucker, le coordonnateur du programme doctoral international (PDI) Christophe Cambier, dont les œuvres ont contribué à la croissance de ma culture scientifique, et le chercheur Serge Stinckwich toujours prompt à me booster. Je n'oublie pas le personnel administratif de l'IRD dont la réactivité a toujours permis à ce que mes séjours à Paris et mon financement se déroulent très bien. Je pense à M. Pedro Verge-Depre, M. Napoléon Koagne, Mme. Colette Essono, Mme. Élisabeth Pereira, Mme. Kathy Baumont et Mme. Rolande Altemaire.

Je remercie mes camarades du PDI pour leur orientation durant mes séjours en France et leur participation à rendre l'ambiance détendu et fraternelle, de quoi permettre de travailler sereinement. Je pense à Cheikhou Ka, Ilhem Bouderbala, Sokhna Bessane, Nisrine Outada, Saad Touhbi, Ahmed Laatabi, Dorra Louati, Ghassen Hadded, Dounia Boufedji, Adil Khalil, Philippe Belmont-Guerron, Lauric Thiault, Malick Diop, Mouhamed Diop, Alain Kemgue, Bruno Kengne et Justin Noubissi.

Je remercie les membres des équipes complexnetworks et IDASCO, pour leurs conseils, leur encadrement et leur réactivité à me débloquer des situations difficiles. Je pense notamment au directeur de recherche Clémence Magnien, à Lionel Tabourier, Raphaël Fournier, Maximilien Danisch, Noé Gaumont, Tiphaine Viard, Marwan Ghanem, Thibaud Arnoux, Aurore Payen, Léo Rannou et Audrey Wilmet à Paris. Et mes collègues Thomas Messi, Vanessa Kamga, Priscille Zoua, Arielle Kitio, Robert Nsaibirni, Gérard Nzebop, Aboubakar Sidiki et Diane Tchuani.

Je ne saurais continuer sans remercier mes enseignants du département d'informatique de l'Université de Yaoundé I qui m'ont tenu depuis mes premières années en filière informatique et ont su inscrire dans mon esprit, les prémisses de cette thèse. Je pense notamment aux professeurs Marcel Fouda, Roger Atsa, René Ndoundam, aux docteurs Jean Nzali, Gilbert Tindo, Serge Moto, Paulin Melatagia, Norbert Tsopze, Hippolyte Tapamo, Etienne Kouokam, Jules Waku, Donatien Chedom, Eric Ngoko, Patrick Kamgueu, Valerie Monthe et aux assistants Serge Ebele, Adamou Hamza, Nadège Meyemdou, Rodrigue Domga et Fidel Jiomekong.

Je remercie mes camarades et amis qui n'ont jamais cessé de travailler avec moi, de me conseiller, de m'encourager, d'être présent pour des réflexions constructives, le partage de nos difficultés, la communion fraternelle à travers des moments de distraction. Les rencontres avec eux permettent d'avoir plus de force pour aller de l'avant. Je pense à Laetitia Mouafo, Léonie Tamo, Samuel Nyobe, Corneille Tchio, Christian Ngono, Emmanuel Moupojou, Samuel Tamene, Cavour Kamgang, Florentin Jiechieu, Darius Tetsa, Franck Keudem, Fabrice Tachago, Shamir Fetcheping, Francis Yuya, Michael Chirmeni et Roméo Mounchipou.

Ce paragraphe est dédié à des personnes qu'on ne remercie jamais assez, celle qui sont toujours si proches de nous que nous n'imaginons pas notre vie sans elle. Ces personnes qui représentent notre famille, notre "chez nous", ces personnes qui se battent pour nous et pour qui on se bat. Ces personnes qui nous réconfortent et nous maintiennent debout en être équilibré psychologiquement et prêt à travailler de manière sereine. Je parle ici de ma famille de sang, ma maman Suzanne, Viviane, Narcisse, Michelle, Eve-marie, Crédo et mes amis très proches William Ngongang, Cédric Djialeu, Carole Jiofack, Delphin Tchekoulong, Ornela Nganso et Dimitri Ngongang. Un merci particulier à ma chère et tendre fiancée Minette Tchamba, la mère de mon fiston Maurice Tchuente (junior), qui a su être patiente et a su m'encourager à aller jusqu'au bout de cet œuvre.

Je remercie tous ceux qui m'ont été d'un quelconque soutient durant mes années de thèse et dont les noms ne figurent pas dans les paragraphes précédents, je vous prie de recevoir mes sincères excuses et j'espère pouvoir vous adresser mes remerciement de vive voix par un autre canal.

Dans un récit d'un des livres qui construisent les justes, il est dit d'un jeune époux qu'il a gardé le bon vin pour la fin, dans cet ordre d'idée, je me permets de conclure cette section en remerciant Celui qui transforma l'eau en vin ; merci à Dieu mon Créateur.

Table des matières

Chapitre 1

Introduction

Sommaire

Depuis le milieu des années 1990, de nombreuses plateformes en ligne proposent de grands nombres de produits à leurs utilisateurs. Ces plateformes sont dédiées à des activités diverses comme la vente de produits physiques (Amazon), l'écoute des chansons et vidéos à la demande (Last.fm [1], Netflix [2]), le partage des liens vers des pages web favorites (CiteUlike [3], Delicious [4]) et la consultation des publications sur les réseaux sociaux (Facebook, Twitter). Le nombre de produits sans cesse grandissant implique que, pour un utilisateur, la sélection d'un produit qui l'intéresse est un problème réel. Pour y remédier, les systèmes de recommandation sont devenus l'une des solutions les plus utilisées car ils proposent à chaque utilisateur un petit ensemble de produits qu'il est le plus susceptible de consommer dans un avenir proche. Ceci permet de réduire le temps de recherche des utilisateurs et facilite leur orientation dans cette masse de données.

1. https ://www.last.fm
2. https ://www.netflix.com
3. https ://www.citeulike.org
4. https ://del.icio.us

L'intégration des systèmes de recommandation dans des plateformes qui mettent en relation leurs utilisateurs avec de grands nombres de produits, a l'avantage d'améliorer la relation client des entreprises propriétaires de ces plateformes et d'augmenter leur chiffre d'affaires. Par exemple, en milieu d'année 2012, suite à l'intégration des systèmes de recommandation dans ses processus d'achat, la société Amazon a enregistré une augmentation de 29% des ventes pour un total de 12,83 milliards de dollars, contre 9,9 milliards de dollars au cours de la même période en 2011 [5]. De même, la société Netflix estime en avril 2012 que 75% des activités de sélection de vidéos sur son site web, sont dûes à la présence des systèmes de recommandation [6]. Cette estimation est passée à 80% en 2016 [7].

Les atouts des systèmes de recommandation encouragent les compagnies à y investir davantage de ressources pour l'amélioration de leurs plateformes. Par exemple, la société Netflix a organisé un grand challenge en 2009 afin de motiver les chercheurs pour la conception des systèmes de recommandations qui seraient meilleurs que le sien d'au moins 10%. A la fin de ce challenge, le meilleur système de recommandation était celui de Koren [79] et il était basé sur la factorisation matricielle avec prise en compte des effets temporels liés au comportement de chaque utilisateur et au changement de la popularité des produits à recommander. L'intégration des algorithmes de Koren dans les processus de recommandation de l'entreprise Netflix a eu un impact positif significatif sur la sélection de vidéos par les utilisateurs comme présenté plus haut pour l'année 2012. Ces éléments justifient l'importance de l'amélioration des systèmes de recommandation qui tiennent compte des aspects temporels.

Dans ce chapitre, nous décrivons le problème du calcul des recommandations top-N dans un contexte particulier de données implicites (section 1.1). Ensuite, nous présentons les flots de liens comme modèle de représentation des données disponibles dans les plateformes de e-commerce, streaming et les réseaux sociaux (section 1.2). Pour terminer nous donnons un positionnement de notre travail dans la section 1.3 en présentant l'objectif de cette thèse, nos contributions et le guide de lecture de ce manuscrit.

1.1 Recommandations top-N

Les approches les plus populaires pour le calcul des recommandations sont la prédiction des notes et la recommandation top-N [34, 129]. Dans la première variante, le système conçu fait une prédiction de la note qu'un utilisateur va attribuer à un produit. Dans la seconde variante, pour chaque utilisateur, le système trie les produits que l'utilisateur n'a pas encore consommés dans l'ordre décroissant des préférences estimées, et seulement les N premiers sont recommandés. Dans cette thèse, nous nous focalisons particulièrement sur la recommandation top-N.

5. http ://fortune.com/2012/07/30/amazons-recommendation-secret/
6. http ://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html
7. http ://www.news.com.au

1.1.1 Importance des recommandations top-N

Les plateformes d'applications des recommandations top-N sont très fréquentes sur le web. Leur mise en œuvre concerne divers type d'objets comme les chansons sur Last.fm, Yahoo ! music, Apple music[146, 73], les films et les vidéos sur youtube, netflix, hulu [34, 38], les liens sociaux sur facebook, linkedin, twitter [30, 145], les produits physiques sur amazon, ebay [90], les journaux d'information et scientifiques sur Google news, Yahoo ! news, Springer [37, 139] et même les lieux intéressants sur Yelp, FourSquare [149, 86]. Dans ces contextes variés d'application des recommandations top-N, il y a deux principales catégories de bénéficiaires : d'une part les utilisateurs et d'autre part les entreprises propriétaires de ces plateformes.

Dans la première catégorie, on constate qu'un utilisateur qui est connecté par exemple sur Youtube et qui veut regarder une nouvelle vidéo intéressante pour lui ne saurait avoir la tâche facile sans l'assistance d'un système de recommandation top-N qui lui présente un ensemble réduit d'items proches de son historique et de l'historique d'autres utilisateurs. Cette démarche permet à l'utilisateur concerné de gagner en temps et d'être satisfait. La satisfaction des clients a pour conséquence de faire de ces derniers des utilisateurs fidèles qui vont davantage effectuer des actions sur cette plateforme. Cet usage régulier par un grand nombre de clients conduit à une croissance du volume des ventes et du chiffre d'affaire des entreprises propriétaires.

La recommandation top-N est en train de devenir le standard le plus pratique du domaine [34], en grande partie grâce à la disponibilité des données utilisées pour leur mise en œuvre.

1.1.2 Données explicites vs données implicites

Les premiers systèmes de recommandation étaient conçus à partir des données explicites qui sont en général des matrices de notes. Dans une matrice de notes $R^{|U|\times|I|}$ où U et I sont respectivement l'ensemble des utilisateurs et des items, chaque cellule $R[u, i]$ est soit nulle, soit contient une valeur réelle qui est la note que l'utilisateur u a attribuée à l'item i sur une plateforme en ligne. Par exemple la plateforme Netflix collecte des notes de 1 à 5 sur les vidéos, et la plateforme Yelp collecte des notes comprises entre 1 et 5 sur les restaurants.

La collecte des données explicites des utilisateurs se fait avec beaucoup de difficultés. En effet, cette démarche requiert une action supplémentaire des utilisateurs qui doivent prendre la peine d'attribuer une note à chaque produit qu'ils consomment. Cependant, les utilisateurs ignorent souvent l'étape qui consiste à noter les produits, par exemple sur Amazon après les avoir réceptionnés. Ce comportement fréquent conduit à une faible proportion d'éléments notés par rapport à l'ensemble des éléments consultés dont les traces sont présentes dans l'historique de navigation. On a donc un manque de données explicites et une abondance des données implicites.

L'historique de navigation d'un utilisateur le met en relation avec des produits qui ont suscité un intérêt de sa part et qu'il a consultés ou consommés. Dans cette situation, la plupart des systèmes de recommandation top-N qui utilisent des données implicites considèrent de nouveau la matrice $R^{|U|\times|I|}$ mais à la différence que $R[u,i]$ contient 1 ou 0 ; 1 quand l'item i est présent dans l'historique des actions de l'utilisateur u et 0 sinon. Cependant, cette représentation des données ignore la dimension temporelle qui est présente dans l'historique de navigation.

La figure 1.1 présente des exemples de matrices de données explicites et implicites. Dans ces matrices, chaque ligne correspond à un utilisateur et chaque colonne correspond à un item.

FIGURE 1.1 – Exemples de matrices des données explicites et des données implicites.

	i_1	i_2	i_3	i_4	i_5
u_1	5	3	4	0	3
u_2	3	0	0	4	5
u_3	0	1	0	5	4
u_4	2	2	5	0	0
u_5	5	3	4	4	0

(a) Matrice des notes explicites

	i_1	i_2	i_3	i_4	i_5
u_1	1	1	1	0	1
u_2	1	0	0	1	1
u_3	0	1	0	1	1
u_4	0	0	1	0	0
u_5	1	1	1	1	0

(b) Matrice des données implicites

1.2 Flots de liens et données implicites

Un flot de liens modélise une suite de triplets $\{(t_k, v_k, w_k)\}_{k=1..|L|}$ indiquant que les entités u et v ont interagi à l'instant t. L'approche par flot de liens est un modèle de représentation des interactions qui est défini dans [80, 136]. et qui est déjà utilisée pour la détection et la visualisation des évènements dans la plateforme Github [60], pour l'identification des rôles des machines et groupes de machines d'un réseau informatique [135] ainsi que la détection des communautés dynamiques [45] et la prédiction des liens dans les réseaux sociaux [10]. Ce modèle est adopté dans cette thèse comme formalisme de représentation des données.

1.2.1 Flot de liens comme représentation des données implicites

Les grands volumes de données qui sont disponibles dans les plateformes en ligne (e-commerce, streaming, réseaux sociaux) au travers des historiques de navigation des utilisateurs peuvent être représentés par des flots de liens. A cet effet, dans un flot de

liens $(t_k, u_k, i_k)_{k=1,..,n}$, chaque triplet met en relation un utilisateur ou client u_k de ces plateformes avec un élément consommable i_k que nous appelons item ou produit tout en conservant l'instant t_k auquel l'utilisateur a sélectionné le produit. L'interprétation des triplets dépend du contexte d'application. Par exemple sur Last.fm, un triplet signifie que l'utilisateur u_k a écouté la chanson i_k à l'instant t_k, sur amazon cela signifie que le client u_k a acheté ou sélectionné le produit i_k à l'instant t_k.

Une fois que les données sont modélisées sous forme de flot de liens, on peut tirer avantage de cette nouvelle représentation en se servant des modèles d'analyse de données et de prédiction qui sont très proches des flots de liens, comme les graphes.

1.2.2 Flot de liens et graphes

Les flots de liens tout comme les graphes mettent en relation des entités qui peuvent être différentes et pour le cas spécifique des recommandations top-N, il s'agit des utilisateurs d'un coté et des items ou produits de l'autre coté. En se servant d'une représentation par graphe biparti [14], voir figure 1.2, on considère que chaque entité est un nœud et on crée systématiquement une arête entre deux entités qui sont impliquées dans le même triplet (t, u, i). Ceci est un formalisme simple et dont les interprétations sont faciles à comprendre [116].

Cependant, une représentation par de tels graphes fait perdre complètement la dynamique temporelle des interactions. Ainsi, ce formalisme doit être conservé mais enrichi par de nouveaux mécanismes qui permettent de mieux prendre en compte la dimension temporelle et d'en profiter pour enrichir le domaine des graphes dans leurs divers contextes d'application, et en particulier celui de la recommandation top-N.

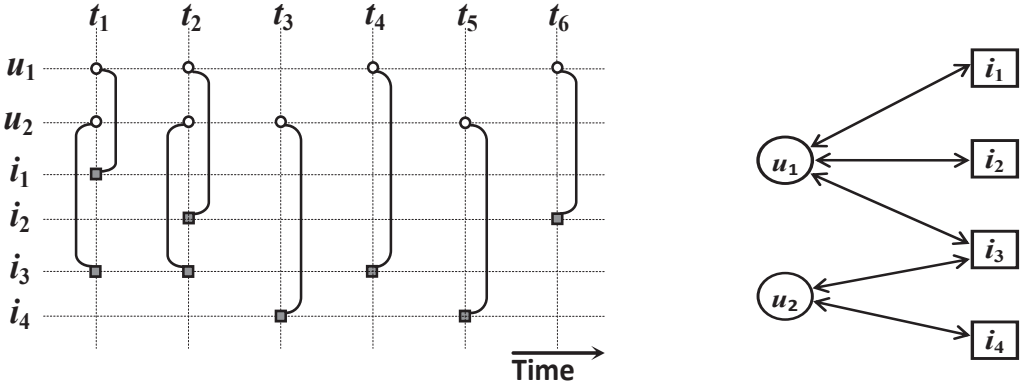
La figure 1.2 illustre une représentation d'un flot de liens et le graphe biparti classique associé. Dans le flot de liens, une entité est représentée par une ligne horizontale et les relations par des arêtes, et dans le graphe, chaque entité est représentée par un nœud et les relations par des arêtes.

1.3 Positionnement du travail

1.3.1 Objectifs

La conception des premiers systèmes de recommandation ne tenait pas compte de l'aspect temporel des informations utilisées [75, 57, 120]. L'une des hypothèses implicite était donc que les goûts et les préférences des utilisateurs ne changent pas au cours du temps. Cette hypothèse s'avère fausse dans la réalité : l'intérêt qu'un individu porte à un produit peut changer avec le temps. De plus, le profil d'un individu peut changer et impliquer systématiquement le changement de ses préférences. Pour prendre en compte de tels changements, la modification des anciennes conceptions des systèmes de recom-

FIGURE 1.2 – Représentation du flot de liens $E = \{(t_1, u_1, i_1), (t_1, u_2, i_3), (t_2, u_1, i_2), (t_2, u_2, i_3), (t_3, u_2, i_4), (t_4, u_1, i_3), (t_5, u_2, i_4), (t_6, u_1, i_2)\}$ et du graphe biparti associé.



(a) Représentation du flot de liens L

(b) Graphe biparti induit par L

mandation s'avère nécessaire. Cette tâche est un grand défi pour la recherche actuelle visant l'amélioration de la qualité des systèmes de recommandation [6, 26, 117].

La mise en œuvre de quelques unes de ces nouvelles techniques a augmenté considérablement la qualité de prédiction des systèmes de recommandation. Par exemple, le prix Netflix 2009 a été remporté par Yehuda Koren grâce à un système de recommandation basé sur la factorisation matricielle enrichi par des aspects temporels [77]. De même d'autres techniques sont enrichies par des concepts liés au temps, à l'instar des k-plus proches voisins (KNN) [82], des machines à vecteur de support (SVM) [89], la factorisation des tenseurs [61] et même des graphes de recommandation [142].

Dans cette thèse, nous nous intéressons à la conception d'un système de recommandation top-N à partir des données implicites. Notre objectif est d'exploiter des concepts temporels et topologiques extraits des flots de liens en vue d'une amélioration des systèmes de recommandation en général et des graphes de recommandation en particulier.

1.3.2 Contributions

Afin d'atteindre notre objectif, nous nous concentrerons sur les graphes de recommandation qui se déduisent facilement des flots de liens, et dont l'interprétation est relativement facile. Dans cette direction, des graphes dynamiques de recommandation ont été proposés à l'exemple du STG (*Session−based Temporal Graph*) de Xiang et al. [142]. L'objectif du STG était de prendre en compte la dynamique temporelle en combinant les préférences à long et à court termes des utilisateurs pour améliorer les recommandations top-N. Cependant, la mise œuvre de ce modèle révèle une limite dans la manière de considérer la dimension temporelle. En effet, lorsqu'on déploie ce modèle sur une longue période, on se rend compte que les nouveaux liens et les liens plus anciens ont le même poids. Cette dernière remarque nous a conduit à une amélioration du graphe STG par

insertion des fonctions de décroissance temporelle afin que les nouveaux liens aient des poids plus importants que les liens plus anciens [100]. Nous présentons cette contribution dans la section 6.2 du chapitre 6.

Le STG intègre toutefois la dynamique temporelle par le biais de sessions périodiques, ce qui nécessite une division du temps en tranches. Cette dernière opération a pour conséquence d'agréger des informations présentes dans une tranche et conduire à une perte de la dynamique des interactions. Afin de pallier à cette limite, nous proposons le graphe-flot de liens (LSG = *Link Stream Graph*) [102], un nouveau graphe de recommandation déduit de la représentation des flots de liens [136, 80]. Ce graphe n'agrège pas les interactions et fait en sorte que la dynamique des actions utilisateur ne soit pas perdue. Nous décrivons le graphe LSG dans la section 6.3 du chapitre 6.

D'autre part, la dimension temporelle n'est pas la seule qui n'est pas prise en compte dans les graphes basiques de recommandation et même dans les graphes qui intègrent le temps comme le STG ou le LSG ; d'autres informations ne sont pas considérées. En effet, les données implicites disponibles regorgent souvent d'autres éléments dont l'absence peut être un frein à l'obtention des bonnes qualités de recommandation. Nous citons par exemple, les propriétés liées aux produits ou aux utilisateurs ou même les relations de confiance qui existent entre les utilisateurs. Nous proposons donc GraFC2T2, un cadre général pour les graphes de recommandation qui facilite l'exploration des avantages de l'utilisation de diverses informations secondaires (*side information*) et trouve des combinaisons appropriées pour des contextes d'application particuliers [101]. Le chapitre 7 est consacré à la description et la mise en œuvre de GraFC2T2.

1.3.3 Guide de lecture

Ce manuscrit est structuré en deux parties, la première est consacrée au filtrage collaboratif et diverses extensions par des informations secondaires. Cette partie contient les quatre prochains chapitres. La seconde partie intitulée graphes de recommandation enrichis, contient deux chapitres qui présentent les contributions de cette thèse. Les paragraphes suivants présentent de brefs résumés des six prochains chapitres. Nous précisons que la dernière section de chaque chapitre est consacrée au résumé de ce chapitre.

Partie I : Filtrage collaboratif.

Filtrage collaboratif pour des recommandations top-N. Dans ce chapitre, nous présentons l'approche de recommandation par filtrage collaboratif. C'est l'approche de recommandation la plus utilisée et la plus étudiée dans la littérature. Les techniques de recommandation issues de cette approche peuvent être regroupées en deux principales catégories : l'une basée sur la mémoire et l'autre basée sur un modèle. Nous présentons des systèmes de recommandation de ces deux catégories dans un contexte où seule la matrice de notes binaires est disponible. Ensuite, nous nous focalisons sur les graphes de

recommandation et quelques limites du filtrage collaboratif classique. Nous clôturons le chapitre en présentant les méthodes d'évaluation des systèmes de recommandation top-N.

Filtrage collaboratif enrichi par des informations basées sur le contenu. Dans ce chapitre, nous décrivons les données qui permettent de décrire le contenu des produits proposés aux utilisateurs. Ensuite, nous montrons que leur intégration dans les techniques du filtrage collaboratif permet de pallier à certaines limites notamment le manque de données, fréquent dans les applications du filtrage collaboratif.

Filtrage collaboratif enrichi par des informations sur la confiance. Ce chapitre est focalisé sur la confiance entre les utilisateurs des plateformes de e-commerce en particulier et des médias sociaux en général. Nous commençons par définir la confiance et donner ses propriétés, ensuite nous présentons les différentes manières de mesurer la confiance. Le chapitre se termine par la présentation des mécanismes d'intégration de la confiance dans les techniques de filtrage collaboratif et leur impact sur ces techniques.

Filtrage collaboratif avec dynamique temporelle. Dans ce chapitre, nous présentons d'abord les différentes visions de la dimension temporelle dans les systèmes de recommandation, puis nous décrivons les principes déployés pour la prise en compte de la dynamique temporelle. Le chapitre se poursuit par la présentation de quelques systèmes qui intègrent la dynamique temporelle, avec éventuellement d'autres informations secondaires (contenu ou confiance). Le chapitre se termine par la description des mécanismes d'évaluation temporelle des recommandations.

Partie II : Graphes de recommandation enrichis.

Dynamique temporelle dans les graphes de recommandation. Dans ce chapitre, nous présentons une première contribution qui enrichit le domaine des graphes de recommandation par l'intégration des fonctions de décroissance temporelle. Puis nous décrivons une seconde contribution, un graphe qui considère le temps de manière continue dans sa structure, contrairement aux autres graphes de base qui soit ignorent le temps, soit le considèrent de manière discontinue.

GraFC2T2 : cadre général pour graphes de recommandation enrichis. Dans ce chapitre nous présentons GraFC2T2, le cadre général pour graphes de recommandation que nous proposons. Nous détaillons les modules de GraFC2T2 en présentant d'abord le module des graphes de base, puis le module qui permet d'enrichir ces graphes avec des attributs des produits et des fonctions temporelles. La description du cadre conceptuel s'achève par la présentation du processus de calcul des recommandations top-N avec l'algorithme du PageRank personnalisé en intégrant la confiance entre utilisateurs. Le chapitre se termine par la mise en œuvre et la présentation des résultats obtenus.

1.4 Résumé

Dans ce chapitre introductif, nous avons expliqué que les systèmes de recommandations top-N sont de plus en plus indispensables sur les plateformes qui présentent de grands nombres de produits à leurs utilisateurs. Ce constat est aussi vrai pour les utilisateurs, qui trouvent facilement des produits qui les intéressent, que pour les entreprises propriétaires, qui voient leur chiffre d'affaire augmenter considérablement. Nous avons également expliqué que la recommandation top-N est en train de prendre le pas sur la prédiction des notes à cause du manque de données explicites et l'abondance des données implicites disponibles à travers les traces de navigation des utilisateurs.

Dans la suite, nous avons montré que les traces de navigation sont facilement modélisées par des flots de liens qui conservent les interactions utilisateur-produit et les instants d'apparition de ces interactions. Ensuite, nous avons exprimé le fait que les flots de liens permettent aisément d'extraire des graphes, qui sont des modèles simples et faciles à interpréter. D'où l'objectif de notre thèse d'exploiter des concepts temporels et topologiques extraits des flots de liens en vue d'une amélioration des systèmes de recommandation top-N existants et particulièrement ceux basés sur des graphes de recommandation. Le chapitre se termine avec une brève description des améliorations que nous apportons à un graphe dynamique de recommandation existant et du cadre de conception des graphes de recommandation que nous proposons.

Première partie

Filtrage collaboratif

Chapitre 2

Filtrage collaboratif pour des recommandations top-N

Sommaire

Les premiers systèmes de recommandation étaient conçus pour prédire les notes que les utilisateurs allaient attribuer aux produits qu'ils n'avaient pas encore notés. Cependant, la présentation des recommandations sur les plateformes en ligne se fait toujours par le biais d'un ensemble réduit de produits. De plus, peu d'utilisateurs notent régulièrement les produits qu'ils consultent, ce qui implique que les données explicites nécessaires à la prédiction des notes ne sont pas suffisamment disponibles, contrairement aux données implicites qui sont les traces des actions des utilisateurs. Ces contraintes pratiques dans le déploiement des systèmes de recommandation renforcent la pertinence des systèmes de recommandation top-N.

L'approche de recommandation la plus utilisée est celle du filtrage collaboratif (CF - *Collaborative Filtering*) qui exploite les actions de l'utilisateur cible (celui pour qui les recommandations sont calculées) pour identifier ses préférences [12]. Les techniques

du filtrage collaboratif supposent que « les utilisateurs qui ont eu les mêmes préférences dans le passé auront les mêmes préférences dans le futur ». A cet effet, ces systèmes recommandent des produits sélectionnés par les utilisateurs les plus proches de l'utilisateur cible [120, 34].

Une classification courante des techniques du filtrage collaboratif considère deux catégories : les systèmes de recommandation basés sur la mémoire et les systèmes de recommandation basés sur un modèle de l'apprentissage automatique [4]. Les techniques basées sur la mémoire utilisent les données (likes, votes, clics) nécessaires pour établir des corrélations (similarités) entre des utilisateurs ou entre les produits afin de calculer les recommandations. Ces techniques reposent largement sur des mesures de similarité pour associer des utilisateurs ou des produits similaires. En revanche, les techniques basées sur un modèle construisent des modèles de classification et/ou de régression de l'apprentissage automatique à partir de l'ensemble des données connues. Le modèle résultant intègre des informations latentes déduites des données brutes de telle sorte qu'il devient possible de calculer de nouvelles recommandations sans utiliser l'ensemble des données à chaque fois.

Dans ce chapitre, nous nous focalisons sur les systèmes de recommandation top-N construits à partir d'information binaires : un utilisateur est oui ou non intéressé par un produit. Dans ce contexte on ne considère pas les informations relatives aux caractéristiques des produits, ni aux horodatages d'enregistrement des actions des utilisateurs. Ainsi, seules les pures techniques du filtrage collaboratif peuvent être appliquées.

Plan du chapitre. Ce chapitre est structuré comme suit : en section 2.1 nous présentons les techniques de filtrage collaboratif en commençant par les techniques basées sur la mémoire (section 2.1.1), puis les techniques basée sur un modèle (section 2.1.2) et mettons en avant les graphes de recommandation (section 2.1.3). Dans la section 2.2, nous présentons des limites du filtrage collaboratif, d'une part les limites liées au principe de l'approche et d'autre part les limites liées aux données utilisées. Nous terminons ce chapitre par la section 2.3 en décrivant des méthodes et des métriques d'évaluation des recommandations top-N.

2.1 Techniques du filtrage collaboratif

L'approche par filtrage collaboratif est la plus utilisée et également la plus étudiée dans la littérature, notamment au travers des techniques basées sur la mémoire comme les k-plus proches voisins [72], les graphes de recommandation [14] et les techniques basées sur un modèle d'apprentissage automatique comme la factorisation matricielle [62], les réseaux bayésiens [118], les machines à vecteurs supports (SVM) [89] et les réseaux de neurones [54].

2.1.1 Techniques basées sur la mémoire

Les techniques de recommandation basées sur la mémoire reposent sur deux principales étapes : la première pour déterminer les corrélations entre les utilisateurs (ou les items) et la seconde pour calculer les prédictions [22]. Dans cette catégorie, les techniques les plus utilisées sont les K-plus proches voisins (KNN) et les graphes de recommandation. Dans cette section, nous présenterons uniquement la technique KNN pour illustrer le fonctionnement des techniques basées sur la mémoire. Nous parlerons des graphes de recommandation en section 2.1.3.

Il existe deux variantes de KNN : la première basée sur les utilisateurs et le seconde basée sur les produits [72, 140]. Dans la technique KNN basée sur les utilisateurs, la première étape consiste à déterminer les K-plus proches voisins de l'utilisateur cible u, c'est-à-dire les K utilisateurs qui lui sont le plus similaires, et la seconde étape consiste à combiner les préférences de ces derniers pour déduire les produits à recommander. En revanche, dans la technique KNN basée sur les produits, pour chaque produit i que l'utilisateur cible u n'a pas encore sélectionné, on détermine les K-plus proches voisins de i dans la première étape, puis on combine les préférences de u pour ces K produits afin d'estimer le taux de préférence de u pour i. Dans la suite nous présentons comment se déroulent les deux principales étapes, la détermination des plus proches voisins et la déduction des recommandations.

Détermination des plus proches voisins

On veut avoir une estimation de la préférence de l'utilisateur cible u pour un produit i. Pour KNN basé sur les utilisateurs, l'objectif de cette première étape est d'avoir un ensemble restreint de K utilisateurs semblables à l'utilisateur cible u et l'objectif de KNN basé sur les produits est d'avoir les K produits les plus similaires à i. Dans les deux cas, on se sert des mesures de similarité qui sont généralement comprises dans l'intervalle $[0, 1]$ ou $[-1, 1]$, avec 1 comme similarité maximal, et plus on s'éloigne de 1, plus on est considéré comme dissimilaire. Pour des données implicites, on peut utiliser les mesures de similarité suivantes.

La mesure de Jaccard. L'hypothèse sous-jacente de cette mesure est que deux utilisateurs qui consomment en permanence les mêmes produits ont une préférence commune. De même, si deux produits sont sélectionnés par les mêmes utilisateurs, cela suppose une caractéristique commune aux deux produits. Pour l'exprimer, on calcule le ratio de la taille de l'intersection des profils des deux entités par rapport à la taille de l'union de leurs profils.

Le profil d'un utilisateur u (resp. produit i) est considéré ici comme étant l'ensemble des produits i' (resp. utilisateurs u') tel que $R_{u,i'} = 1$ (resp. $R_{u',i} = 1$). L'équation 2.1 donne la mesure de Jaccard entre deux utilisateurs et l'équation 2.2 représente la similarité

entre deux produits. I_u représente l'ensemble des items déjà consommés par l'utilisateur u et U_i représente l'ensemble des utilisateurs qui ont déjà sélectionné le produit i.

$$jaccard(u_1, u_2) = \frac{|I_{u_1} \cap I_{u_2}|}{|I_{u_1} \cup I_{u_2}|} \tag{2.1}$$

$$jaccard(i_1, i_2) = \frac{|U_{i_1} \cap U_{i_2}|}{|U_{i_1} \cup U_{i_2}|} \tag{2.2}$$

La mesure du cosinus. Une autre façon de calculer la similarité entre deux entités consiste à traiter chaque entité comme un vecteur dans un espace vectoriel à $|I|$ dimensions pour les utilisateurs et à $|U|$ dimensions pour les produits. Ensuite, on utilise le cosinus entre ces vecteurs comme mesure de la similarité. Formellement, si $\overrightarrow{R}_{u,*}$ est le vecteur de taille $|I|$ de l'utilisateur u et $\overrightarrow{R}_{*,i}$ est le vecteur de taille $|U|$ du produit i dans la matrice des notes binaires $R^{|U|\times|I|}$, alors la similarité entre les utilisateurs u_1 et u_2 (resp. produits i_1 et i_2) est estimée par le cosinus de l'angle formé par les vecteurs $\overrightarrow{R}_{u_1,*}$ et $\overrightarrow{R}_{u_2,*}$ (resp. $\overrightarrow{R}_{*,i_1}$ et $\overrightarrow{R}_{*,i_2}$) donné par l'équation 2.3 (resp. équation 2.4).

$$cos(\overrightarrow{R}_{u_1,*}, \overrightarrow{R}_{u_2,*}) = \frac{\overrightarrow{R}_{u_1,*} \cdot \overrightarrow{R}_{u_2,*}}{||\overrightarrow{R}_{u_1,*}|| \times ||\overrightarrow{R}_{u_2,*}||} = \frac{\sum_{i\in I} R_{u_1,i} R_{u_2,i}}{\sqrt{\sum_{i\in I} R^2_{u_1,i}}\sqrt{\sum_{i\in I} R^2_{u_2,i}}} \tag{2.3}$$

$$cos(\overrightarrow{R}_{*,i_1}, \overrightarrow{R}_{*,i_2}) = \frac{\overrightarrow{R}_{*,i_1} \cdot \overrightarrow{R}_{*,i_2}}{||\overrightarrow{R}_{*,i_1}|| \times ||\overrightarrow{R}_{*,i_2}||} = \frac{\sum_{u\in U} R_{u,i_1} R_{u,i_2}}{\sqrt{\sum_{u\in U} R^2_{u,i_1}}\sqrt{\sum_{u\in U} R^2_{u,i_2}}} \tag{2.4}$$

Il existe d'autres moyens de mesurer la similarité qui peuvent être appliqués sur des données implicites. On peut par exemple citer les travaux de Karypis [72] qui présentent une mesure de similarité basée sur les probabilités conditionnelles ou les travaux de Bernardes et al. [19] qui présentent une mesure de similarité basée sur les règles d'association. Chaque méthode diffère par les opérations qui sont appliquées pour le calcul des similarités et peut avoir un effet important sur la qualité des recommandations.

Après avoir choisi la mesure de similarité, on identifie les plus proches voisins de l'utilisateur cible u (resp. produit i), soit en fixant une valeur k comme nombre de voisins à considérer, soit en fixant un seuil de similarité s. Si on se fixe une valeur k, alors seuls les utilisateurs (resp. produits) associés aux k plus grandes similarités par rapport à u (resp. à i) sont considérés comme faisant partie du voisinage de u (resp. i). Dans le cas où c'est le seuil s qui est fixé, alors tout utilisateur (resp. produit) dont la similarité par rapport à u (resp. i) est supérieure ou égale à s est admis dans le voisinage de u (resp. i). Ensuite, il ne reste plus qu'à déduire la préférence de l'utilisateur u pour le produit i à l'aide du voisinage déterminé.

Calcul des recommandations

Pour le calcul des recommandations top-N de l'utilisateur cible u, si le modèle est basé sur les utilisateurs alors on fait collaborer les voisins V_u de u afin d'avoir une estimation de la préférence de u pour chacun des items qu'il n'a pas encore sélectionné $i_* \notin I_u$. En revanche, si le modèle est basé sur les produits alors on fait collaborer les voisins V_i de i. Ensuite les items $i_* \notin I_u$ sont classés dans l'ordre décroissant de préférence et seuls les N premiers sont recommandés à u. Le calcul des préférences est donné par l'équation 2.5 pour le KNN basé sur les utilisateurs, et par l'équation 2.6 pour le KNN basé sur les produits.

$$preference(u,i) = \frac{\sum_{u' \in V_u} R_{u',i} \times sim(u,u')}{\sum_{u' \in V_u} sim(u,u')} \tag{2.5}$$

$$preference(u,i) = \frac{\sum_{i' \in V_i} R_{u,i'} \times sim(i,i')}{\sum_{i' \in V_i} sim(i,i')} \tag{2.6}$$

Dans cette opération de calcul des recommandations, les utilisateurs qui sont les plus similaires à u ont une plus grande influence sur le choix des produits à lui recommander. Cette remarque montre que le choix de la mesure de similarité est d'une grande importance, de même que le choix du voisinage, car seuls les utilisateurs présents dans V_u ont une influence sur les recommandations faites à u. A cet effet, la plupart des travaux sur le KNN s'attardent soit sur la mesure de similarité, soit sur le choix du voisinage [57, 120].

Dans cette section nous avons présenté la technique KNN qui est l'une des plus célèbres techniques de filtrage collaboratif basées sur la mémoire. Cependant, cette technique utilise toutes les données disponibles à chaque fois qu'il faut recommander, et donc le modèle entier doit être en mémoire. Ce dernier constat peut être un souci si la mémoire est limitée. Dans ce cas, les techniques basées sur un modèle d'apprentissage peuvent être une bonne alternative.

2.1.2 Techniques basées sur un modèle

Les techniques de recommandation basées sur un modèle reposent sur deux principales étapes : la première est l'apprentissage du modèle de prédiction et la seconde est le calcul des recommandations [22]. De telles techniques définissent un modèle qui généralise la connaissance extraite des données (informations latentes) de telle sorte qu'on n'ait pas besoin de toutes les données disponibles pour recommander, contrairement aux techniques basées sur la mémoire.

Les systèmes de recommandation évoqués dans cette section reposent sur des modèles de réduction de dimensionnalité ou de clustering dans le but d'écarter les utilisateurs ou les items non représentatifs. Dans cette catégorie, le modèle le plus utilisé est la factorisation matricielle (MF) [62, 56]. Néanmoins, d'autres modèles sont également utilisées comme les réseaux bayésiens [118] et les machines à vecteurs de supports (SVM). Dans cette catégorie, il y a également les réseaux de neurones [55, 54] qui sont de plus en plus

fréquents dans le domaine des systèmes de recommandation top-N.

Factorisation matricielle

Dans cette section nous présentons la méthode de décomposition en valeurs singulières (SVD), qui est un modèle de factorisation matricielle très utilisée dans les systèmes de recommandation [121, 79]. Pour l'application de cette méthode, on part de la matrice des notes binaires R pour calculer un ensemble de k facteurs latents à partir desquels on caractérise à la fois les produits et les utilisateurs. La valeur de k (nombre de facteurs latents) est fixée en avance.

De manière générale, la valeur k est choisie de manière arbitraire, mais doit toujours être inférieure ou égale au rang de la matrice des notes ; pour $r = rang(A)$, $k \leq r$. Une fois que la valeur de k est fixée, la matrice $R^{|U|\times|I|}$ peut être approximée par $\widetilde{R}$ le produit de trois matrices construites à partir des k vecteurs propres de la matrice $R \cdot R^T$ (resp. $R^T \cdot R$) et des k plus grandes valeurs propres de R. L'équation 2.7 montre comment calculer $\widetilde{R}$.

$$\widetilde{R} = \widetilde{U} \times \Sigma \times \widetilde{I}^T \tag{2.7}$$

- $\widetilde{U}$ est une matrice de taille $|U| \times k$ qui décrit tous les utilisateurs par k facteurs latents. Les colonnes de $\widetilde{U}$ sont les vecteurs propres des plus grandes valeurs propres de la matrice $R \cdot R^T$.
- Σ est une matrice carrée diagonale de taille $k \times k$. Les valeurs sur la diagonale principale sont les plus grandes valeurs propres de la matrice $R \cdot R^T$ classées dans l'ordre décroissant.
- $\widetilde{I}$ est une matrice de taille $|I| \times k$ qui décrit tous les produits par k facteurs latents. Les colonnes de $\widetilde{I}$ sont les vecteurs propres des plus grandes valeurs propres de la matrice $R^T \cdot R$.

La décomposition de R en produit de trois matrices permet d'avoir les matrices $\widetilde{U}$ et $\widetilde{I}$ qui contiennent respectivement la description des utilisateurs et des produits par k facteurs latents. Un facteur latent correspond souvent à une caractéristique qu'on connait. Par exemple, pour la recommandation des vidéos, un facteur latent peut correspondre à un genre "Action" ou "Horreur". Cependant, il est possible qu'on n'arrive pas à interpréter un facteur latent. Dans la suite, $\widetilde{U}$ et $\widetilde{I}$ sont utilisés pour calculer les recommandations.

Calcul des recommandations. Le principe ici est de recommander à un utilisateur u les produits qui lui sont similaires dans le nouvel espace vectoriel à k dimensions. Ainsi, pour estimer la préférence qu'un utilisateur u peut avoir pour un produit i, on calcule le produit des vecteurs de u et i dans le nouvel espace vectoriel. Cette estimation est donnée

par l'équation 2.8.

$$preference(u,i) = \sum_{f=1}^{k} \widetilde{U}_{u,f} \times \widetilde{I}_{f,i} \tag{2.8}$$

Les N premiers produits que l'utilisateur cible u n'a pas encore sélectionnés et qui sont associés aux plus grandes valeurs de $preference(u, i_*)$ lui sont recommandés.

En dehors de la méthode SVD, d'autres modèles de factorisation matricielle ont déjà été utilisées dans la littérature pour le calcul des recommandations top-N. On peut citer les travaux qui portent sur l'analyse à composantes principales [155] et ceux qui portent sur l'analyse linéaire discriminante (LDA) [144]. Les techniques de factorisation matricielle sont les plus courantes dans la littérature, mais d'autres modèles sont aussi très utilisés.

Autres modèles d'apprentissage automatique

Outre la factorisation matricielle, d'autres modèles d'apprentissage automatique sont également fréquents dans la littérature des systèmes de recommandation top-N. La plupart de ces travaux transforment le problème de recommandation en un problème de classification binaire [55] ou en un problème de classification uni-classe [89]. Dans les deux cas, il est question pour un couple utilisateur-produit (u, i) non encore observé, d'exprimer à quelle proportion est-ce que (u, i) appartient à la classe 1, en supposant que 1 est la classe des liens utilisateur-produit positifs. La différence entre les deux types de classification, est que dans la classification binaire, on apprend le modèle avec des exemples positifs et des exemples négatifs, alors que dans la classification uni-classe, on apprend uniquement à partir des exemples positifs.

Une autre caractéristique de ces travaux est l'usage d'une représentation intermédiaire des données afin d'extraire des descripteurs des couples utilisateur-produit. Ces descripteurs sont utilisés en entrée des classifieurs considérés. Par exemple He et al. [55] appliquent la factorisation matricielle pour avoir k descripteurs des utilisateurs et des produits, puis ils entrainent le réseau de neurones à partir de ces descripteurs. On a également des systèmes de recommandation qui sont basés sur un modèle probabiliste de prédiction comme les réseaux bayésiens [22].

Techniques hybrides : basées sur la mémoire et sur un modèle. Notons que certains systèmes de recommandation combinent une technique basée sur la mémoire à une technique basée sur un modèle. Par exemple, Li et al. [89], représentent d'abord la matrice des notes binaires sous forme de graphe biparti classique. Puis décrivent les couples utilisateur-produit (u, i) à l'aide des propriétés du contexte topologique de l'utilisateur et du produit concerné. Enfin, ils entrainent un modèle de machines à vecteurs supports uni-classe et déduisent les recommandations.

Comme autre exemple, on a des travaux dans lesquels la factorisation matricielle est

combinée à KNN [16, 76]. Dans ces travaux, pour calculer la préférence de l'utilisateur cible u pour un produit i qu'il n'a pas encore sélectionné, on procède comme pour les KNN sauf que le calcul des similarités se fait dans l'espace réduit à k dimensions résultant de la factorisation matricielle. De même, le calcul peut être basé sur les similarités entre utilisateurs ou les similarités entre produits. La mesure de similarité à utiliser dans ce contexte repose sur des vecteurs à l'exemple de la mesure du cosinus.

Limites des techniques basées sur un modèle. Malgré le gain pour le stockage du modèle en mémoire, l'usage des modèles d'apprentissage automatique pour implémenter des systèmes de recommandation a quelques inconvénients :

- ces modèles peuvent être très complexes et même être basés sur des caractéristiques qui ne sont pas interprétables dans la réalité.
- l'optimisation des hyper-paramètres nécessite une quantité importante de données disponibles. Ceci peut être un problème lors du lancement d'une nouvelle plateforme de e-commerce.
- le temps d'apprentissage d'un modèle peut être très grand en fonction du nombre d'utilisateur et du nombre de produits.
- l'ajout d'un nouvel utilisateur ou d'un nouveau produit peut nécessiter de réapprendre le modèle.

Compte tenu de ces dernières remarques, il est compréhensible que de nombreuses alternatives aux systèmes de recommandation basés sur un modèle d'apprentissage automatique soient présentes dans la littérature. Par exemple, beaucoup utilisent les graphes de recommandation.

2.1.3 Graphes de recommandation

Les graphes sont très utilisés pour étudier la structure et la composition de divers systèmes. Un graphe est constitué d'un ensemble d'éléments appelés nœuds ou sommets et de connexions appelées arêtes ou liens. La recherche des associations transitives dans le contexte des systèmes de recommandation est généralement mise en œuvre à l'aide d'un modèle basé sur des graphes pour deux raisons : un graphe est facilement interprété et fournit un cadre plus naturel et intuitif pour différents types d'applications. Deuxièmement, de nombreux algorithmes basés sur des graphes peuvent être directement implémentés dans différents domaines comme le e-commerce ou les réseaux sociaux.

Les graphes de recommandation définis pour le calcul des recommandations top-N à partir des données implicites ont les mêmes étapes que les autres systèmes de recommandation basés sur la mémoire comme KNN : une étape pour établir les corrélations entre les entités et une seconde étape pour le calcul des recommandations. Dans ce contexte, la première étape est celle de la représentation des données par un graphe et la seconde

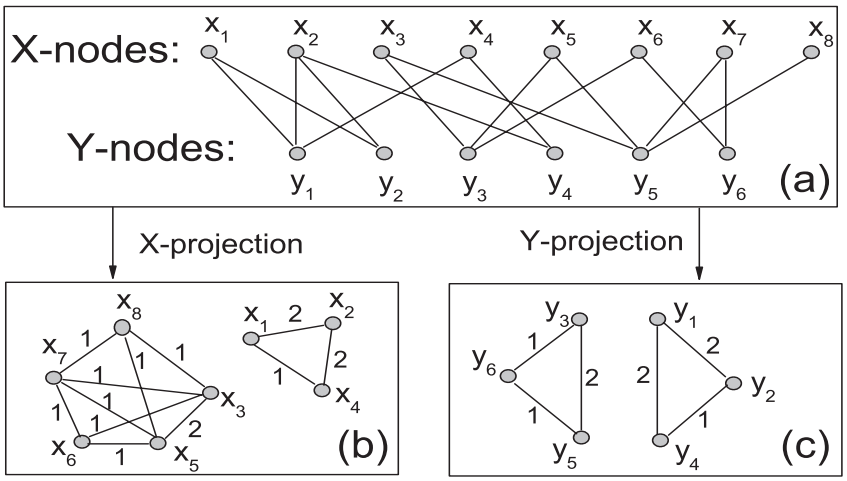
étape correspond à l'application d'un algorithme de calcul des recommandations à partir du graphe construit précédemment.

Construction du graphe

Le principal graphe de recommandation qui est construit à partir de la matrice des notes binaires est le graphe biparti classique. Dans ce graphe, chaque utilisateur (respectivement chaque produit) est représenté par un nœud. Lorsque $R_{u,i} = 1$, le nœud de l'utilisateur u est relié au nœud du produit i par une arête (u, i). Ce graphe est fréquent comme modèle de représentation des données dans plusieurs travaux sur les données implicites à l'exemple de ceux de Baluja et al. [14] sur la recommandation des vidéos sur Youtube et Yan et al. sur la recommandation des tweets [145].

D'autres travaux se servent des graphes projetés du graphe biparti classique [154, 19]. Le graphe peut être projeté sur la dimension utilisateur ou sur la dimension produit. Dans le cas du graphe projeté utilisateur, tous les nœuds correspondent à des utilisateurs. Il y a une arête entre deux nœuds si les deux utilisateurs associés ont sélectionné des produits en commun et le poids de cette arête est fonction du nombre de produits en commun. Dans le cas du graphe projeté produit, tous les nœuds correspondent à des produits et une arête entre deux nœuds signifie que les deux produits associés ont été sélectionnés par le même utilisateur. Le poids de l'arête dépend du nombre d'utilisateurs pour qui cela est vrai. La figure 2.1 présente un exemple de graphe biparti et les deux graphes projetés associés.

FIGURE 2.1 – Exemple de graphe biparti classique (a), sa projection sur la dimension X (b) et la dimension Y (c). Les poids des arêtes en (a) et (b), sont les nombres de voisins communs dans X et dans Y respectivement [154].



Une fois les graphes de recommandation construits, on s'en sert pour déterminer les recommandations. Pour chaque type de graphe utilisé, il existe divers algorithmes

de calcul des recommandations top-N. Nous présentons les principales variantes de ces algorithmes dans la section suivante.

Calcul des recommandations

Lorsque le graphe de recommandation est le graphe biparti, l'hypothèse de recommandation repose sur la proximité de l'utilisateur cible aux prochains produits qu'il va sélectionner. Ainsi, l'objectif est de recommander les N produits que l'utilisateur cible n'a pas encore sélectionnés et qui sont les plus proches de lui dans le graphe de recommandation. Suivant ce principe, la plupart des algorithmes de recommandation sur les graphes bipartis sont basés sur la marche aléatoire dans laquelle le nœud source est l'utilisateur cible [14, 142].

Les variantes de la marche aléatoire reposent sur la différence dans la diffusion des poids d'un nœud à l'autre et sur le nombre de pas à effectuer. On a par exemple l'algorithme Injected Preference Fusion (IPF) [142] dans lequel les poids sont calibrés dans la propagation et le nombre de pas limité à 3 ou le PageRank initial dans lequel le nombre de pas dépend d'un seuil de convergence [106]. D'autres algorithmes comme Katz [44] et HITS (*Hyperlink−induced Topic Search*) [64] peuvent également être utilisés.

Lorsque le graphe de recommandation est un graphe projeté, il s'agit du graphe projeté sur la dimension produit. Dans ce cas, pour recommander de nouveaux produits à un utilisateur u, c'est également un processus de marche aléatoire qui est appliqué ; tous les produits i tel que $R_{u,i} = 1$ sont des sources de la propagation [154]. Les produits à recommander sont classés suivant leur proximité aux précédentes sélections de l'utilisateur cible u. Plus un produit est proche des sélections de u, plus ce produit est dans les premières positions du top-N.

Notons que le graphe biparti conserve beaucoup plus d'information sur les préférences des utilisateurs comparé aux graphes projetés. Par exemple, dans un graphe projeté, on ne sait pas qui a sélectionné quoi ? Cette différence sous-entend que les graphes bipartis sont les plus adéquats pour une bonne qualité de recommandations.

D'autre part, le développement technologique atteint sur les tailles des mémoires encourage d'avantage l'usage des graphes de recommandation. Par exemple, le système de recommandation Pixie proposé récemment par Eksombatchai et al. [43], développé et déployé sur Pinterest [1], est un système de recommandations en temps réel basé sur les graphes. Eksombatchai et al. ont développé l'algorithme *Pixie Random Walk* qui calcule des recommandations sur le graphe de Pinterest constitué de 3 milliards de nœuds et de 17 milliards d'arêtes. Ces avancées font parties des raisons qui font que nous nous focalisons sur les graphes dans cette thèse.

1. Pinterest est un site web qui mélange les concepts de réseaux sociaux et de partage de photographies. www.pinterest.com

2.2 Limites du filtrage collaboratif

Bien que les techniques de filtrage collaboratif connaissent un grand succès, ces dernières ont des limites que nous regroupons en deux catégories : les limites liées au principe et les limites liées aux données.

2.2.1 Limites liées au principe

Dans le principe du filtrage collaboratif, lorsqu'il faut recommander de nouveaux produits à un utilisateur cible u, on lui propose des produits que des utilisateurs qui sont similaires à u ont aimé. Ce principe impose une notion de similarité qui donne naissance aux limites suivantes : le démarrage à froid, le problème des utilisateurs $Gray-sheep$ et la sensibilité aux attaques de faux profils.

Démarrage à froid

Pour calculer les similarités entre les utilisateurs ou entre les produits, il faut avoir suffisamment de liens utilisateur-produits qui impliquent ces utilisateurs ou ces produits. Par conséquent, les techniques de filtrage collaboratif ne peuvent pas recommander de nouveaux produits à un nouvel utilisateur qui n'a pas encore effectué un nombre significatif d'action dans le système. On parle de démarrage à froid des utilisateurs.

De plus, pour recommander un produit à un utilisateur, il faut que ce produit ait déjà été sélectionné par au moins un utilisateur similaire à l'utilisateur cible. Et donc un produit qui vient d'être ajouté au catalogue des produits de la plateforme ne sera pas recommandé par une technique pure du filtrage collaboratif. On parle de démarrage à froid des produits.

Grey-sheep users

Pour recommander de nouveaux produits à un utilisateur, on se sert des opinions des utilisateurs qui lui sont similaires. Cependant certains utilisateurs ont des préférences qui ont tendance à être divergentes de celles des autres ; ce sont des utilisateurs $Grey-sheep$ [48, 49]. En d'autres termes, ces utilisateurs ne sont pas similaires aux autres et donc les techniques pures du filtrage collaboratif ne sauraient leur proposer des produits qui les intéressent.

Sensibilité aux attaques de faux profils

Les attaques sont conçues pour que le système de recommandation agisse de la manière souhaitée par l'attaquant. Il pourrait soit recommander certains produits souhaités, soit empêcher de recommander d'autres produits. L'une des attaques à laquelle le filtrage collaboratif est sensible est l'insertion de faux profils dans la plateforme. Par exemple,

lorsqu'il faut recommander de nouveaux produit à un utilisateur u, s'il y a un faux profil très similaire à celui de u, alors les propositions faites à u seront biaisés par ce faux profil.

2.2.2 Limites liées aux données

Les techniques du filtrage collaboratif reposent généralement soit sur des notes explicites que les utilisateurs ont attribuées aux produits, soit sur les traces des actions des utilisateurs sur les produits (données implicites). Considérer uniquement ce type de données est à l'origine de plusieurs autres limites des techniques du filtrage collaboratif : le problème des données creuses et la non-utilisation des données supplémentaires.

Données creuses

Les techniques du filtrage collaboratif nécessitent le calcul des similarités entre les utilisateurs ou entre les produits. Cependant de nombreux algorithmes de calcul des similarités deviennent presque inutilisables lorsque la taille des vecteurs qui représentent les utilisateurs ou les produits augmente et dépasse un certain seuil [24]. Lorsque le nombre d'utilisateurs ou de produits augmente, la matrice utilisateur-produit devient extrêmement creuse.

Par exemple, si une plateforme a des millions de produits, aucun utilisateur ne pourra noter ou voir tous ces produits. De manière optimiste, même si chaque utilisateur consulte quelques milliers de produits, on aboutit toujours à une grande matrice creuse. Ce qui rend difficile le calcul des mesures de similarité d'une part à cause de la taille des matrices et d'autre part à cause du manque de données.

Données supplémentaires ignorées

Les techniques pures du filtrage collaboratif reposent soit sur des matrices explicites des notes que les utilisateurs accordent aux produits, soit sur des matrices binaires construites à partir de l'historique des actions des utilisateurs sur les produits. Dans les deux situations, plusieurs autres types de données sont ignorés. On peut citer par exemple les textes disponibles pour décrire les caractéristiques des produits, les données sur les relations d'amitié et de confiance entre les utilisateurs et même les horodatages des actions des utilisateurs sur les produits.

La prise en compte de telles données supplémentaires peut aider à réduire l'impact de certaines limites des techniques pures du filtrage collaboratif comme le manque de données. Par exemple, l'usage des données sur les caractéristiques des produits peut aider à proposer des produits qui ont des caractéristiques similaires à ceux que l'utilisateur cible a sélectionnés dans le passé. Ainsi, même les produits qui viennent d'être ajoutés au catalogue des produits de la plateforme peuvent être recommandés.

D'autre part, la prise en compte des données sur les relations de confiance ou d'amitié entre les utilisateurs peut permettre de proposer à l'utilisateur cible des produits

que ses amis aiment même s'il n'a jamais sélectionné un produit de la plateforme. Ceci peut avoir l'avantage de réduire l'impact des faux profils car les recommandations seront d'avantage axées sur les relations d'amitié et de confiance. De même, les utilisateurs $grey-sheep$ pourront recevoir plus de recommandations provenant des individus à qui ils font confiance, ce qui est mieux car dans le cas contraire ils auront des recommandations qui proviennent des utilisateurs avec qui ils n'ont rien en commun.

De plus, les techniques pures du filtrage collaboratif n'exploitent pas les horodatages qui sont très souvent enregistrés dans l'historique des actions des utilisateurs. Ceci peut être une limite car les profils et les préférences des utilisateurs changent avec le temps. Par exemple, les produits qu'un utilisateur aimait quand il était élève ou étudiant ne seront pas les mêmes une fois qu'il aura un travail. Un autre point important est le fait que les produits se démodent et donc la prise en compte du temps peut aider à réduire la recommandation de produits démodés.

2.3 Évaluation des recommandations top-N

La plupart des recherches sur les systèmes de recommandation portent sur la conception et l'amélioration de la performance des algorithmes de recommandation proposés [58, 124]. Afin de comparer et de sélectionner les algorithmes les plus performants parmi plusieurs algorithmes, il est nécessaire de mesurer et de comparer leurs performances. Cette comparaison est généralement faite empiriquement à partir d'expérimentation dans lesquelles les performances des algorithmes sont testées en ligne ou hors ligne. Ceci est fait en appliquant un protocole d'évaluation particulier, c'est-à-dire en utilisant une méthodologie d'évaluation qui définit comment évaluer et certains paramètres d'évaluation qui définissent quoi évaluer.

2.3.1 Méthodes d'évaluation des recommandations

Après la conception d'un système de recommandation, la qualité des recommandations produites peut être évaluée en-ligne ou hors-ligne. Dans le cas d'une évaluation en-ligne, le système de recommandation est déployé sur un site internet et des tests de recommandation sont appliqués sur des internautes qui évaluent les propositions qu'ils reçoivent. Ils peuvent remplir des questionnaires concernant leur expérience du système et les recommandations reçues [74]. Si les internautes sont satisfaits alors le système de recommandation concerné est dit satisfaisant, dans le cas contraire, ce système est jugé peu efficace.

En revanche, pour l'évaluation hors-ligne on se sert des jeux de données qui contiennent des traces des liens utilisateur-item passés pour simuler comment les utilisateurs se seraient comportés s'ils avaient utilisé le système évalué. Ainsi, les données sont divisées en deux partis disjointes, une parti comme jeu d'apprentissage (comportements passés

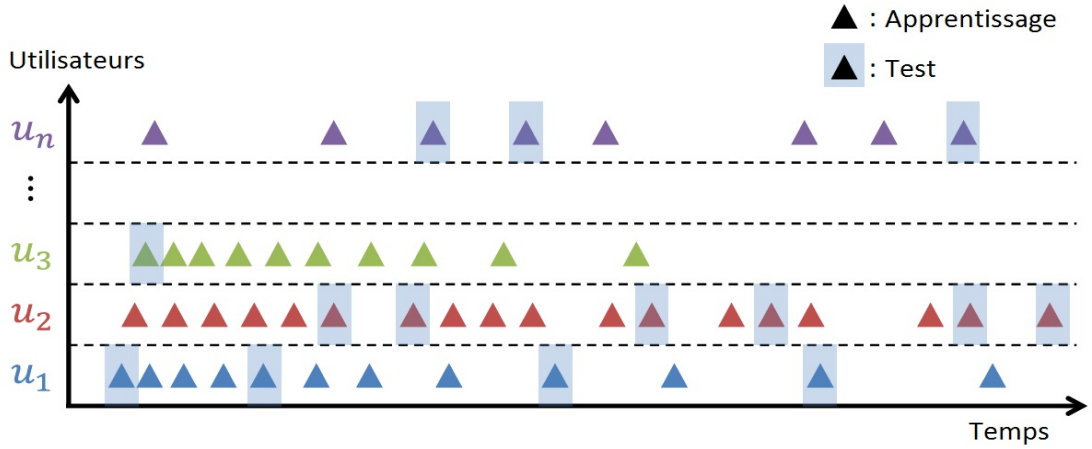
des utilisateurs que le système connaît) et la partie restante comme jeu de test (comportements futurs des utilisateurs qu'il faut prédire). Les résultats de l'évaluation sont obtenus en comparant les listes de produits recommandés aux listes réelles de produits sélectionnés dans le jeu de test [124].

L'évaluation en-ligne peut être considérée comme préférable à l'évaluation hors-ligne, principalement en raison de sa capacité à prendre en compte l'expérience de l'utilisateur [74]. Malgré ses avantages, l'évaluation en-ligne est plus difficile et plus coûteuse à réaliser, car elle nécessite la mise en œuvre complète du système à évaluer. De plus, les utilisateurs doivent être recrutés et probablement payés pour tester le système. L'évaluation hors-ligne, par contre, ne nécessite que l'implémentation des algorithmes du système à tester. Grâce à la disponibilité des données, aucun recrutement d'utilisateurs n'est nécessaire. Ainsi, l'évaluation hors-ligne apporte un environnement expérimental peu coûteux et facile à reproduire pour tester de nouveaux algorithmes. Compte tenu de ces dernières remarques, la méthode d'évaluation hors-ligne est la plus utilisée dans la littérature. Nous suivons ce courant, et menons uniquement des évaluations hors-ligne dans notre travail.

2.3.2 Protocole d'évaluation hors-ligne : validation croisée

Plusieurs travaux sur les systèmes de recommandation utilisent un protocole d'évaluation qui procède à un fractionnement aléatoire (indépendant du temps) des jeux d'apprentissage et de test. Dans ces cas, une sélection aléatoire de données (liens implicites ou notes explicites) est effectuée pour avoir le jeu de test de chaque utilisateur et le reste est intégré au jeu d'apprentissage. Il est possible de se fixer un nombre de données ou de se fixer un pourcentage de données à sélectionner de manière aléatoire pour le jeu de test. La figure 2.2 montre un schéma de sélection aléatoire du jeu de données. Dans la figure, les triangles ombrés représentent les données attribuées au jeu de test.

FIGURE 2.2 – Sélection aléatoire des jeux d'apprentissage et de test. Les triangles ombrés représentent les liens utilisateur-produit sélectionnés pour le jeu de test et le reste est attribué au jeu d'apprentissage [27].



Les conditions de validation croisée indiquent si un ou plusieurs groupes de données (c'est-à-dire des paires d'ensembles jeu d'apprentissage et jeu de test) sont construits avec les données sur l'historique des actions des utilisateurs. Des recherches en statistiques [9] et en apprentissage automatique [39] ont montré que la variabilité des résultats d'une évaluation est réduite lorsqu'on répète plusieurs fois le processus d'évaluation en utilisant des répartitions de données différentes à chaque fois. Cette procédure est communément appelée validation croisée.

Les méthodes de validation croisée qui ne tiennent pas compte du temps utilisent une autre condition d'ordre indépendante du temps pour construire Z partitionnements des données en deux ensembles tel que $\Sigma = \{(Ja_z, Jt_z)\}_{z=1..Z}$. Dans chacun des partitionnements, les deux ensembles sont disjoints, c'est-à-dire $Ja_z \bigcap Jt_z = \phi$. Nous donnons un bref aperçu de ces méthodes dans les paragraphes ci-dessous.

Échantillonnage répété. Cette méthode effectue Z échantillonnages aléatoires en fonction de la taille désirée pour chaque jeu de données. Une séquence différente est générée à chaque répétition, en raison de l'utilisation d'un ordre aléatoire. Ceci permet de garantir que chaque division (jeu de test et jeu d'apprentissage) sera différente des autres. Cette méthode a été appliquée dans [46].

Ré-échantillonnage des utilisateurs. Cette méthode effectue Z échantillonnages aléatoires d'un sous-ensemble d'utilisateurs. Ensuite les données de ces utilisateurs sont réunies pour former le jeu de données qui sera ensuite divisé en jeu d'apprentissage et jeu de test. Cette méthode a été utilisée dans [152].

Z-fold cross validation. Il s'agit d'une méthode couramment utilisée qui prend l'ensemble global de toutes les données et le divise en Z sous-ensembles disjoints indépendamment du temps des données concernées. Ensuite, Z couples (jeu d'apprentissage, jeu de test) sont construits de telle sorte que chaque sous-ensemble de rang z du découpage précédent, représente le jeu de test de rang z et l'union des $(Z-1)$ sous-ensembles restants représente le jeu d'apprentissage de rang z. En général, les sous-ensembles sont de même taille et donc une fois que Z est fixé, on peut déduire la taille des Z sous-ensembles. Cette méthode est utilisée dans [5].

Leave-one-out. C'est un cas particulier de $Z-fold\ cross\ validation$ dans lequel Z est égal au nombre de liens du jeu de données global ; la taille de chaque sous-ensemble est de 1. Chaque note est considérée comme l'ensemble de tests et les notes restantes sont utilisés dans le jeu d'apprentissage. Bien que cette méthode ait la plus faible variabilité des résultats dans les problèmes de prédiction [9], son coût de calcul élevé le rend irréalisable dans de nombreuses situations. Cette méthode a été appliquée dans [35].

En général, les méthodes de validation croisée indépendantes du temps ont un inconvénient lors de l'évaluation des systèmes de recommandation. Ils peuvent produire des jeux d'apprentissage et de test qui se chevauchent du point de vue temporel. Il est donc possible d'apprendre un modèle avec des données plus récentes tandis que le jeu de test contient des données plus anciennes. Pour pallier à cette limite, des méthodes d'évaluation qui tiennent compte du temps on été proposées, nous les décrivons dans la section 5.3.1 du chapitre 5.

2.3.3 Métriques d'évaluation

Dans le cadre de l'évaluation hors-ligne, de nombreuses métriques d'évaluation ont été proposées dans la littérature. Chacune permet d'évaluer un aspect de la qualité des recommandations [50]. Dans cette thèse nous avons considéré 3 catégories de ces métriques d'évaluation : les métriques de la classification binaire, les métriques d'évaluation du classement dans les listes de recommandation et enfin les métriques de satisfaction des utilisateurs [11].

Dans la suite nous supposons qu'on a |U| utilisateurs et que le système de recommandation a recommandé N produits à chaque utilisateur. Nous présentons les équations de calcul des métriques d'évaluation dans ce contexte.

Métriques de la classification binaire

Le problème de recommandation est similaire aux problèmes de classification binaire. En effet, lorsque les données sont divisées en deux, une partie pour l'apprentissage, et la seconde pour le test, les liens utilisateur-produit présents dans la partie test représentent les futurs liens. Ainsi, lorsqu'un système de recommandation propose un lien (u, i), ce lien est soit de la classe *correct* (présent dans le jeu de test), soit de la classe *incorrect* (absent du jeu de test). Ainsi, les métriques d'évaluation de la classification binaire sont utilisées pour évaluer les recommandations top-N. Nous avons les métriques suivantes.

Précision. La précision est la proportion du nombre de bonnes recommandations sur le nombre total des recommandations. Soit $hit_N(u)$ le nombre de bonnes recommandations pour N produits recommandés. Le calcul de la précision est décrit par l'équation 2.9.

$$Precision@N = \frac{\sum_{u \in U} hit_N(u)}{|U| \times N} \tag{2.9}$$

Rappel. Le rappel est la proportion du nombre de bonnes recommandations sur le nombre total des liens à recommander dans le jeu de test. Soit $I_{new}(u)$ le nombre de nouveaux produits à recommander à l'utilisateur u dans le jeu de test. Le calcul du

rappel est effectué grâce à la formule de l'équation 2.10.

$$Rappel@N = \frac{\sum_{u \in U} hit_N(u)}{\sum_{u \in U} I_{new}(u)} \tag{2.10}$$

Mesure-F1. Le score-F1 est un compromis entre la précision et le rappel de sorte que l'optimisation du score-F1 est plus robuste que l'optimisation de la précision ou du rappel. Pour un utilisateur u, $Precision = \frac{hit_N(u)}{N}$, $Recall = \frac{hit_N(u)}{I_{new}(u)}$ et $F1 = 2 \cdot \frac{Precision \times Recall}{Precision + Recall} = 2 \cdot \frac{hit_N(u)}{I_{new}(u)+N}$. On déduit que le calcul du score-F1 pour l'ensemble des utilisateurs est donné par l'équation 2.11.

$$F1@N = \frac{\sum_{u \in U} 2 \times hit_N(u)}{\sum_{u \in U}(I_{new}(u) + N)} \tag{2.11}$$

Dans cette catégorie de mesure d'évaluation de la classification binaire, on peut ajouter l'usage de la courbe ROC et de la métrique AUC pour estimer la qualité d'un système de recommandation top-N. En revanche, il est possible d'utiliser d'autres métriques pour évaluer les recommandations et qui ne sont pas utilisées dans la classification binaire.

Métriques qui tiennent compte de l'ordre

Les métriques d'évaluation précédentes ne tiennent pas compte de l'ordre des éléments dans la liste des N produits recommandés à chaque utilisateur. Cependant, dans la pratique, lorsqu'un utilisateur est sur une plateforme en ligne, il parcourt les produits recommandés dans un ordre, et plus vite il retrouve un produit qui l'intéresse, plus il est satisfait des recommandations. Cette réalité motive l'usage des mesures d'évaluation qui tiennent compte de l'ordre des produits dans la liste recommandée. On peut citer les métriques ci-après.

Mean Reciprocal Rank (MRR) ou Average-Reciprocal Hit Rank (ARHR). Le rang réciproque d'une liste de recommandation top-N, est l'inverse du rang de la première bonne recommandation de la liste : 1 pour la première place, 1/2 pour la deuxième place, 1/3 pour la troisième place et ainsi de suite. Le score MRR est la moyenne des rangs réciproques des recommandations top-N proposées aux utilisateurs. En notant $rank_N(u)$ la position de la première bonne recommandation faite à u dans la liste des N produits recommandés, le calcul du MRR est donné par l'équation 2.12.

$$MRR@N = \frac{1}{|U|} \times \sum_{u \in U} \frac{1}{rank_N(u)} \tag{2.12}$$

Mean Average Precision (MAP). En appliquant cette mesure, on effectue des calculs de la précision en tenant compte de la position des produits pertinents parmi les N

recommandés, contrairement à la métrique de précision ordinaire. Le calcul de la MAP est donné par l'équation 2.13.

$$MAP@N = \frac{\sum_{u \in U} AP_N(u)}{|U|} \tag{2.13}$$

$$AP_N(u) = \frac{\sum_{k=1}^{N} \frac{hit_k(u)}{k} \times h(k)}{hit_N(u)} \tag{2.14}$$

où $AP_N(u)$ désigne la précision moyenne des recommandations top-N proposées à l'utilisateur u et $h(k) = 1$ si le produit à la position k est une bonne recommandation et 0 sinon.

En dehors de ces deux métriques, une métrique qui est également courante dans la littérature est la métrique NDCG (*N-Discounted Cumulative Gain*) [42]. La principale différence entre MAP et NDCG est que MAP prend en compte la pertinence binaire (un produit est d'intérêt ou non), tandis que NDCG permet des scores de pertinence sous forme de nombres réels positifs. En effet, pour calculer la qualité d'une liste de produits recommandés, avec NDCG, un score de pertinence est attribué à chaque produit de la liste, et les produits pour lesquels il n'y a pas de retour de l'utilisateur, leur score est fixé à zéro. Cette particularité justifie le fait que NDCG est généralement utilisée pour évaluer des recommandations calculées à partir des données explicites (notes explicites des utilisateurs).

Les métriques de cette catégorie tiennent compte de l'ordre des produits dans la liste des recommandations. Cependant, avoir une bonne performance suivant ces métriques, ne garantit pas qu'on ait une grande proportion d'utilisateurs satisfaits. Cette dernière remarque justifie l'usage des métriques de la section suivante.

Proportion de clients satisfaits.

La principale motivation d'un entrepreneur quand il utilise un système de recommandation est d'augmenter son chiffre d'affaire, et cela est possible si le système de recommandation lui garantit qu'une grande proportion d'utilisateurs seront satisfaits. C'est lorsque ces utilisateurs effectuent un achat et/ou sont fidèles à sa plateforme que l'entrepreneur a de bon chiffres d'affaires. Le Hit ratio [72] et ARHR [31] évaluent les performances sous cet angle.

Hit ratio. Le Hit ratio est la proportion d'utilisateurs à qui le système de recommandation a fait au moins une bonne recommandation. L'équation 2.15 présente le calcul de cette métrique.

$$HitRatio@N = \frac{\sum_{u \in U} (hit_N(u) > 0)}{|U|} \tag{2.15}$$

Mean Reciprocal Rank (MRR) ou Average-Reciprocal Hit Rank (ARHR). MRR de l'équation 2.12 est de nouveau mentionné car c'est une métrique qui permet d'évaluer simultanément la proportion d'utilisateurs à qui le système a fait au moins une bonne recommandation et de prendre en compte la rapidité d'accès à cette bonne recommandation par les utilisateurs cibles.

Dans cette section nous avons présenté des métriques d'évaluation qui permettent d'estimer les performances des systèmes de recommandation top-N sous plusieurs angles : l'exactitude des prévisions, la précision du classement et la satisfaction des clients. Outre ces aspects, d'autres propriétés de recommandation font l'objet de travaux récents. C'est le cas de la nouveauté et de la diversité [134], au moyen de métriques telles que la métrique SI (*Self Information*) [153] et ILS (*Intra List Similarity*) [157]. Les paramètres de nouveauté visent à déterminer dans quelle mesure des produits inconnus (pour un utilisateur en particulier ou pour l'ensemble de la communauté) sont recommandés, tandis que les paramètres de diversité évaluent à quelle mesure les produits d'une liste de recommandations sont similaires entre eux.

2.4 Résumé

Dans ce chapitre nous avons présenté une synthèse sur les techniques de filtrage collaboratif construites à partir d'une matrice des notes binaires. Ces techniques peuvent être scindées en deux catégorie, d'une part celles basées sur la mémoire et d'autre part celles basées sur un modèle d'apprentissage automatique. Les techniques basées sur le modèle permettent de limiter les besoins de mémoire, cependant ces techniques ont des limites comme le temps d'apprentissage, les difficultés de mise à jour du modèle et le problème du démarrage à froid. Ces dernières remarques, ajouté au fait que les avancés technologiques permettent d'avoir de plus grandes mémoires, justifient l'importance que nous accordons aux graphes de recommandation.

Une fois que les systèmes de recommandation sont conçus, il est nécessaire de les évaluer et de les comparer pour ne retenir que les plus performants. A cet effet, nous avons présenté les principales méthodes d'évaluation : en-ligne et hors-ligne. Notre choix a été porté sur l'évaluation hors-ligne car moins coûteuse et plus facile à dupliquer pour comparer les systèmes de recommandation. Enfin, nous avons présenté quelques métriques d'évaluation hors-ligne.

Les systèmes de recommandation présentés dans ce chapitre ont des limites comme le démarrage à froid, la sensibilité aux attaques par de faux profils, et ignorent également les informations secondaires liées aux caractéristiques des produits, aux relations de confiance entre les utilisateurs et même à la dynamique temporelle. Cependant, ces aspects sont pris en compte dans de nombreux travaux pour améliorer les performances des techniques du filtrage collaboratif. Ces travaux font l'objet des prochains chapitres.

Chapitre 3

Filtrage collaboratif enrichi par des informations basées sur le contenu

Sommaire

Les techniques du filtrage collaboratif abordées dans le chapitre précédent utilisent les corrélations entre les actions passées des utilisateurs pour déduire des recommandations. Ces techniques n'utilisent pas les caractéristiques des produits pour étudier les goûts des utilisateurs. Cependant, si un utilisateur aime un film de super-héros comme "*Avengers*", il y a de fortes chances qu'il aime également un autre film de super-héros comme "*Iron man*". Dans ce cas, les opinions des autres utilisateurs peuvent ne pas être nécessaires pour proposer des recommandations pertinentes. De plus, ce type d'information est particulièrement utile lorsqu'on a des nouveaux produits sur lesquels il y a peu d'actions des utilisateurs. Cette dernière remarque montre que la prise en compte des informations sur le contenu peut aider à pallier au manque de données et au démarrage à froid (des produits) dont les techniques de filtrage collaboratif sont victimes.

Les systèmes de recommandation qui prennent en compte des informations sur le contenu sont conçus pour les cas dans lesquels les produits peuvent être décrits par des

caractéristiques diverses. Cette condition est vérifiée dans de nombreuses plateformes dédiées aux films, chansons et livres, pour lesquels on a des données sur les genres, les auteurs et les mots-clés. Il est donc judicieux d'intégrer les informations du contenu dans les techniques du filtrage collaboratif pour atténuer le problème de manque de données et améliorer la description des goûts et préférences des utilisateurs. Dans ce chapitre nous donnons un aperçu de ces techniques hybrides en décrivant d'abord comment les informations sur le contenu sont considérées et enfin comment ces informations sont combinées au filtrage collaboratif.

Plan du chapitre. Nous entamons ce chapitre par la présentation des informations basées sur le contenu des produits dans la section 3.1. Ensuite nous donnons une description des systèmes de recommandation basés sur le contenu dans la section 3.2. Le chapitre s'achève par la section 3.3 avec la description des systèmes de recommandation hybrides qui combinent les filtrages collaboratif et basé sur le contenu.

3.1 Informations sur le contenu des produits

Les systèmes de recommandation basés sur le contenu (CBF - $Content{-}Based\ Filtering$) analysent et exploitent le contenu des produits afin de trouver des produits similaires à ceux connus et préférés par l'utilisateur cible, en supposant que ces produits similaires sont également intéressants pour l'utilisateur cible [5, 113]. Cependant, la similarité entre deux produits i_1 et i_2 ici, n'est plus liée au fait que les mêmes utilisateurs sélectionnent i_1 et i_2, mais plutôt aux informations sur le contenu des produits.

3.1.1 Types d'informations du contenu

Au niveau le plus élémentaire, les systèmes basés sur le contenu dépendent de deux sources de données : la première est une description formelle des produits en termes d'attributs centrés sur le contenu. Un exemple d'une telle représentation pourrait être la description textuelle d'un article par le fabricant ou par les propriétaires de la plateforme d'e-commerce. La deuxième source de données repose sur les retours des utilisateurs, qui peuvent générer des commentaires ou attribuer des étiquettes aux produits comme pour les logiciels de marquage social (*social bookmarking*) à l'exemple de CiteUlike et Delicious.

Quelle que soit la source des données, ces dernières peuvent être de deux types : structurées ou non structurées. On parle de données non structurées par exemple lorsque des mots-clés sont extraits des textes qui décrivent les produits. Ces textes peuvent être des descriptions fournies par le fabricant ou les commentaires des utilisateurs. En ce qui concerne les données structurées, les produits sont décrits par des catégories ou des taxonomies connues, par exemple le nom du fabricant, le genre, la taille ou le prix.

Dans la suite de cette section, nous décrivons les processus d'extraction et de sélection des attributs des produits, et dans les sections suivantes nous décrivons comment ces attributs sont utilisés dans les techniques du filtrage basé sur le contenu et leur intégration pour améliorer les techniques du filtrage collaboratif.

3.1.2 Extraction des attributs

La première phase des techniques basées sur le contenu consiste à extraire des caractéristiques discriminantes pour représenter les produits. Les caractéristiques discriminantes sont celles qui prédisent fortement les intérêts des utilisateurs. Cette phase dépend fortement de l'application à laquelle on est confronté. Par exemple, un système de recommandation de page web sera très différent d'un système de recommandation de chansons ou de vidéos.

Lors de cette phase, on doit pouvoir extraire des attributs qui permettent de décrire objectivement les produits du système. L'approche la plus courante consiste à extraire des mots-clés de données textuelles [20]. Ce choix s'explique par le fait que les descriptions textuelles non structurées sont souvent largement disponibles dans divers domaines et que l'usage du texte reste le moyen le plus naturel pour décrire les produits. Cependant, d'autres techniques peuvent être utilisées, notamment dans des cas comme la recommandation de chansons illustrée dans l'un des exemples ci-après.

Recommandation de produits qui ont des descriptions textuelles. Dans de nombreux cas, un produit peut avoir plusieurs caractéristiques qui permettent de décrire les différents aspects de ce produit. Par exemple, pour un livre on peut avoir son titre, son auteur, ses mots-clés et le résumé du livre ; pour un film on peut avoir son genre, le réalisateur, la liste des acteurs principaux et une brève description du film.

Parmi les systèmes de recommandation conçus pour des produits qui ont des descriptions textuelles [111, 29], certains systèmes réunissent un sous-ensemble de mots-clés utilisés pour décrire les caractéristiques des produits dans un sac de mots-clés, tandis que d'autres travaillent directement à une représentation multidimensionnelle des produits de telle sorte que chaque caractéristique correspond à une dimension. D'autres systèmes utilisent un sac de mots mais attribuent des poids différents aux mots en fonction des caractéristiques dans lesquels ces mots ont été extraits. Cette dernière catégorie repose sur le fait que les noms des acteurs devraient avoir beaucoup de poids dans la recommandation de films par exemple.

Recommandation de chansons. Dans le cas de la recommandation de chansons, en plus des textes descriptifs, certains travaux utilisent des attributs extraits du fichier audio. Les attributs concernés ici peuvent être liés au timbre, au rythme et à la tonalité [28, 65]. Cette démarche vise à recommander des chansons similaires pas seulement concernant

les étiquettes mais également concernant les sonorités. Cependant, même les caractéristiques extraites des fichiers audio sont catégorisées en mots-clés. Ainsi, une chanson peut toujours être représentée par un sac de mots-clés qui proviennent des caractéristiques différentes ou par une représentation multidimensionnelle comme évoqué précédemment.

3.1.3 Principe de sélection des attributs pertinents

Les mots-clés extraits des caractéristiques des produits constituent les attributs de description de ces produits. Les mots-clés courants, qui apparaissent dans une grande majorité des produits, sont statistiquement moins discriminants. Par conséquent, on accorde moins de poids à ces mots et plus de poids à ceux qui sont plus discriminants et qui permettent de mieux apprendre les goûts des utilisateurs [111].

3.2 Systèmes de recommandation basés sur le contenu

Les techniques du filtrage basé sur le contenu supposent que les catégories de produits préférées d'un utilisateur dans le passé seront les mêmes dans le futur. Ainsi, la recommandation basée sur le contenu tente de comparer des produits en utilisant leurs caractéristiques (genre, acteurs, éditeur, auteur) pour recommander de nouveaux produits similaires à ceux précédemment sélectionnés par l'utilisateur cible [96].

Le filtrage basé sur le contenu impose des étapes de base qui restent présentes dans les différentes techniques. Ces étapes sont la phase de pré-traitement des données, la phase d'apprentissage des profils des utilisateurs et des produits, et enfin la phase de recommandation de nouveaux produits aux utilisateurs.

3.2.1 Étapes des systèmes de recommandation basés sur le contenu

Pré-traitement et extraction des attributs. Les systèmes basés sur le contenu sont utilisés dans une grande variété de domaines, tels que la navigation web, le streaming des chansons et des films et la diffusion des articles sur l'actualité. Comme évoqué dans la section précédente, dans ces cas, il est nécessaire d'extraire des mots-clés ou des informations qu'on peut catégoriser à partir des données disponibles ; ça peut être par exemple la fouille de texte [111, 29] ou de l'extraction d'information à partir des fichiers audio [65]. Cependant, ce processus peut générer un grand nombre de mots-clés, auquel cas il faut les réduire en un nombre raisonnable, de manière à ne retenir que les plus pertinents. Le choix de ces attributs influence la qualité des systèmes de recommandation basés sur le contenu, car ces attributs constituent la base des étapes suivantes.

Apprentissage des profils des utilisateurs et des produits. Dans les systèmes de recommandation basés sur le contenu, il est courant de représenter le profil d'un

produit par un sac de mots-clés et leurs fréquences dans une représentation en espace vectoriel [111]. De même le profil d'un utilisateur peut être assimilé à un vecteur pondéré des caractéristiques du contenu, en donnant plus de poids aux caractéristiques présentes dans les produits préférés de cet utilisateur.

En d'autres termes, le système s'appuie sur les attributs des produits qu'un utilisateur a aimés dans le passé : c'est l'ensemble de ces attributs qui constitue son profil. Le profil d'un utilisateur peut être explicite ou implicite ; un profil est implicite lorsque les attributs sont déduits des précédents choix de l'utilisateur et explicite lorsque l'utilisateur a fourni lui même une liste des attributs qui l'intéressent. Dans cette étape, des techniques de classification peuvent être utilisées pour attribuer des étiquettes au profil de chaque utilisateur [7].

Filtrage et recommandation. Dans cette étape, pour recommander des produits à un utilisateur, on utilise son profil construit à l'étape précédente. L'objectif est de lui recommander des produits qui correspondent à son profil et qu'il n'a pas encore sélectionné dans le passé. A cet effet, les techniques du filtrage basé sur le contenu calculent pour chaque couple (u, i), un score $F(u, i)$ qui indique à quel point les caractéristiques de i correspondent aux préférences de u [12, 113].

Pour avoir ce score, certaines techniques calculent directement la similarité entre le contenu des produits et le profil de l'utilisateur cible. Par exemple, en utilisant les vecteurs pondérés des caractéristiques du contenu, l'application de la mesure du cosinus peut permettre de trouver les produits similaires à ceux préférés par l'utilisateur cible.

Une autre manière de procéder est celle des techniques basées sur un modèle, ces dernières créant un modèle des préférences utilisateur basé sur le contenu des produits, et utilisant le modèle construit pour calculer le score $F(u, i)$. Par exemple, dans [111] les auteurs utilisent un modèle bayésien pour classer les pages web comme "intéressantes" ou "non intéressantes" pour un utilisateur, à partir d'un ensemble de pages précédemment notées par l'utilisateur concerné. Ce classifieur Bayésien est utilisé pour calculer la probabilité qu'une page web appartient à une classe ("intéressant" ou "non intéressant"), étant donné le vecteur des caractéristiques du contenu qui décrit cette page.

3.2.2 Atouts et inconvénients des systèmes basés sur le contenu

Atténuer le démarrage à froid des produits. Le principal avantage du filtrage basé sur le contenu est sa capacité à recommander des produits auxquels aucune note n'a été attribuée. Ceci permet de résoudre le problème du démarrage à froid des nouveaux produits qu'on rencontre avec le filtrage collaboratif. Les recommandations avec le filtrage basé sur le contenu sont donc utiles lorsque le manque de données est très élevé ou lorsque l'ensemble des produits change rapidement, mais pas l'ensemble de leurs étiquettes, comme dans le domaine des recommandations des articles sur l'actualité [12].

Sensibilité au manque de données. Les techniques du filtrage basé sur le contenu nécessitent d'autres types de données en dehors de l'historique des actions des utilisateurs ; on peut citer la description des produits et les étiquettes des catégories de ces produits. Ainsi, ces techniques peuvent être limitées, lorsque les données sur le contenu sont en quantité insuffisante ou ne contiennent pas suffisamment d'informations utiles qui permettent de bien discriminer les produits et bien exprimer les préférences des clients.

Manque de surprise et de diversité des recommandations. Les techniques de filtrage basé sur le contenu souffrent de la sur-spécialisation des recommandations, car ces techniques suggèrent toujours des produits similaires à ceux déjà connus de l'utilisateur et ne peuvent donc pas fournir de recommandations nouvelles ou diverses qui puissent surprendre l'utilisateur [23]. De plus, ces techniques requièrent que l'utilisateur cible ait au nombre suffisant de produits qu'il a aimés dans le passé, afin de bien connaître ses préférences [5, 113]. Ceci montre clairement que l'usage des informations sur le contenu ne résout pas le problème du démarrage à froid des utilisateurs ; au contraire, les techniques du filtrage basé sur le contenu en souffrent également.

3.3 Combinaison des filtrages collaboratif et basé sur le contenu

Lorsqu'un utilisateur sélectionne un produit pour la première fois sur une plateforme de e-commerce, cela peut être dû aux attributs de ce produit. Par exemple, une personne qui aime un artiste musicien peut écouter le nouvel album de cet artiste dès l'instant que cet album est inséré dans la plateforme. Cette remarque laisse penser que l'intégration des informations du contenu dans les techniques de filtrage collaboratif peut avoir un impact positif sur la qualité des recommandations. Cette idée a inspiré la conception de nouveaux types de systèmes de recommandation dits hybrides, dans lesquels l'objectif est de combiner le meilleur des deux approches pour créer des systèmes de recommandation encore plus robustes [12, 15, 122].

En effet, malgré les inconvénients des techniques basées sur le contenu, notamment le démarrage à froid des utilisateurs et le problème de diversité et de nouveauté des recommandations, ces techniques peuvent compléter les techniques du filtrage collaboratif en raison de leur capacité à exploiter les connaissances basées sur le contenu dans le processus de recommandation. Ce comportement complémentaire est souvent mis à profit pour pallier au démarrage à froid des produits et au manque de données dont souffrent les techniques du filtrage collaboratif, mais également de produire des recommandations diverses contrairement aux techniques de filtrage basé sur le contenu. Dans la suite, nous présentons quelques méthodes de conception de tels systèmes de recommandation.

3.3.1 Méthodes de combinaison des deux approches

Il y a plusieurs manières de combiner le filtrage collaboratif et le filtrage basé sur le contenu dans un système hybride. Conformément aux travaux de Adomavicius et Tuzhilin [5], les systèmes hybrides qui combinent le filtrage collaboratif et le filtrage basé sur le contenu peuvent être classés comme suit : (1) implémenter séparément les deux approches et combiner leurs recommandations, (2) incorporer certaines caractéristiques basées sur le contenu dans une technique du filtrage collaboratif, (3) incorporer certaines caractéristiques de collaboration dans une technique basée sur le contenu, et (4) construire un modèle qui intègre à la fois des caractéristiques de collaboration et du contenu. Toutes ces quatre méthodes ont été utilisées par des chercheurs en recommandation.

Implémentations séparées. Un moyen de créer des systèmes de recommandation hybrides consiste à mettre en œuvre deux techniques distinctes de filtrage collaboratif et de filtrage basé sur le contenu. Nous pouvons avoir deux scénarios différents. D'une part on peut combiner les résultats obtenus à partir des systèmes de recommandation individuels en une recommandation finale en utilisant soit une combinaison linéaire [32], soit un schéma de vote [112]. D'autre part, on peut utiliser l'un des systèmes en choisissant à chaque fois celui qui est meilleur en fonction de certaines mesures de la qualité des recommandations [20].

Ajouter des caractéristiques basées sur le contenu dans le filtrage collaboratif. Plusieurs systèmes de recommandation hybrides reposent sur des techniques traditionnelles du filtrage collaboratif, mais maintiennent également les profils basés sur le contenu pour chaque utilisateur [12, 112]. Ces profils basés sur le contenu et pas uniquement sur les notes explicites ou les notes binaires implicites, sont ensuite utilisés pour calculer la similarité entre les utilisateurs. Comme mentionné dans [112], cela permet de surmonter certains problèmes liés au manque de données des techniques purement collaboratives, car il y a des paires d'utilisateurs qui ont peu de produits sélectionnés ou notés en commun.

Un autre avantage est qu'il est possible de recommander un produit pas seulement lorsque celui-ci est très apprécié par les utilisateurs similaires, mais aussi lorsque ce produit correspond au profil de l'utilisateur [12]. Dans [94], les auteurs utilisent une approche du filtrage collaboratif dans laquelle le vecteur des notes des utilisateurs est complété par des notes supplémentaires calculées à l'aide d'un modèle de prédiction basé sur le contenu.

Notons que c'est dans cette catégorie que s'inscrivent les graphes de recommandation qui intègrent les attributs du contenu des produits comme des nouveaux types de nœuds du graphe. Par exemple, pour la recommandation des films, Phuong et al. [114] et Bogers [21] construisent un graphe dans lequel ils intègrent un nouveau type de nœuds dédié aux caractéristiques basées sur le contenu : les acteurs, les genres et les étiquettes que les utilisateurs attribuent aux films. Ensuite, ils appliquent des algorithmes de marche

aléatoire pour calculer les recommandations à partir du graphe résultat. D'autre part Huang et al. [63] construisent un graphe des produits dans lequel deux produits sont reliés lorsqu'ils ont en commun des caractéristiques basées sur le contenu.

Ajouter des caractéristiques de collaboration dans le filtrage basé sur le contenu. L'approche la plus populaire dans cette catégorie consiste à utiliser une technique de réduction de dimensionnalité sur un groupe de profils basés sur le contenu. Par exemple, dans [127] les auteurs utilisent l'indexation sémantique latente pour créer une vue collaborative d'un ensemble de profils d'utilisateur, où les profils d'utilisateurs sont représentés par des vecteurs de mots-clés, ce qui permet d'améliorer les performances par rapport à la technique purement basée sur le contenu.

Développer un système de recommandation qui unifie les deux approches. De nombreux chercheurs ont suivi cette approche. Par exemple, [15] proposent d'utiliser des caractéristiques basées sur le contenu et sur la collaboration dans un seul classifieur basé sur des règles d'association. Dans [122], les auteurs proposent une méthode d'analyse sémantique probabiliste latente pour combiner des recommandations collaboratives et basées sur le contenu. Une autre approche est proposée dans [33] et [8], qui utilisent des modèles de régression bayésiens.

3.3.2 Avantage des combinaisons des deux approches

Plusieurs articles, tels que [12, 127, 94] comparent empiriquement les performances des systèmes hybrides aux méthodes pures du filtrage collaboratif ou du filtrage basé sur le contenu, et démontrent que les systèmes de recommandation hybrides peuvent fournir des recommandations plus précises que les techniques pures de filtrage collaboratif ou de filtrage basé sur le contenu. Cette conclusion peut être justifiée grâce aux avantages des systèmes de recommandation hybride.

Atténue le démarrage à froid des produits. Les systèmes de recommandation hybrides permettent de recommander des produits qui sont nouveaux dans le système grâce aux caractéristiques basées sur le contenu de ces produits. En d'autres termes, il est possible de recommander un produit qu'aucun ou très peu d'utilisateurs ont sélectionné ou noté dans le passé.

Atténue le problème du manque de données. Pour construire le profil des utilisateurs, les techniques des filtrages collaboratif et basé sur le contenu ont besoin d'avoir suffisamment de données sur chaque utilisateur ; un grand nombre de produits sélectionnés ou notés dans le passé. En combinant les deux approches, on a plus de données disponibles pour décrire les goûts des utilisateurs, car les caractéristiques basées sur le contenu s'ajoutent aux caractéristiques liées au filtrage collaboratif et vice-versa.

Propose des recommandations diverses et surprenantes. En combinant le filtrage collaboratif au filtrage basé sur le contenu, ce ne sont pas uniquement les recommandations liés aux attributs des produits qui sont proposées mais également des recommandations liés aux avis des autres utilisateurs qui sont proches de l'utilisateur cible. Ceci permet d'éviter la sur-spécialisation des recommandations qui empêche de surprendre les utilisateurs, et favorise des recommandations diversifiées.

Augmente la couverture des recommandations. Dans le filtrage collaboratif classique, on a accès uniquement aux produits qui ont déjà été sélectionnés par au moins un voisin de l'utilisateur cible, tandis que dans les systèmes de recommandation hybride, on peut atteindre des produits qui n'ont pas été sélectionnés par des voisins de l'utilisateur cible, mais qui ont des caractéristiques proches de celles des anciennes sélections de l'utilisateur cible.

Malgré ces nombreux avantages des systèmes de recommandation hybrides, le problème de démarrage à froid des utilisateurs demeure. Les techniques des filtrages collaboratif et basé sur le contenu ont toutes deux ce problème.

3.4 Résumé

Le filtrage collaboratif, malgré son efficacité à produire de bonnes recommandations, reste limité par des problèmes tels que le manque de données et le démarrage à froid des utilisateurs et des produits. De plus, le filtrage collaboratif ignore les données qui permettent de décrire les produits proposés aux utilisateurs et qui sont disponibles dans de nombreuses plateformes en ligne dédiées par exemple au streaming de chansons ou des films, ou à la diffusion d'articles sur l'actualité. Au contraire, le filtrage basé sur le contenu se sert de ces données disponibles pour extraire des attributs capables de décrire au mieux les différents produits dans le but de proposer à chaque utilisateur des produits qu'il n'a pas encore sélectionnés et qui sont similaires aux produits qu'il a sélectionnés dans le passé (du point de vue des attributs).

Les attributs considérés par le filtrage basé sur le contenu devraient être pris en compte dans le filtrage collaboratif pour améliorer la qualité des recommandations, car un utilisateur peut sélectionner un produit pour la première fois à cause de certaines caractéristiques de ce produit qui sont décrites par des attributs. Pour ce faire, de nombreux travaux combinent les techniques des filtrages collaboratif et basé sur le contenu pour construire des systèmes de recommandation hybrides. Ces derniers tirent avantages des deux approches, et permettent de pallier aux problèmes du manque de données et du démarrage à froid des produits. Cependant, le problème de démarrage à froid des utilisateurs demeure, d'où l'importance des notions abordées dans le chapitre suivant.

Chapitre 4

Filtrage collaboratif enrichi par des informations sur la confiance

Sommaire

L'approche du filtrage collaboratif est la plus étudiée et la plus utilisée dans la littérature. Cependant, cette approche a des limites qui peuvent être des freins à l'obtention de meilleures performances de recommandation, notamment le démarrage à froid et le manque de données, comme nous l'avons mentionné dans la section 2.2 du chapitre 2. D'autre part, en plus des matrices des données utilisateur-produit, on dispose souvent de données qui décrivent les relations sociales des utilisateurs et même parfois des données dans lesquelles certains utilisateurs précisent clairement qui sont ceux à qui ils font confiance dans le système. C'est le cas des plateformes comme Epinions [1], Ciao [2] et FilmTrust [3].

Toutes ces remarques donnent lieu à des travaux sur les systèmes de recommandation basés sur la confiance qui veulent d'une part résoudre certains problèmes du filtrage

1. http ://www.epinions.com
2. https ://www.ciao.co.uk/
3. http ://www.filmtrust.eu/

collaboratif, et d'autre part exploiter les informations liées aux relations sociales ou à la confiance. Pour donner un aperçu de ces travaux, ce chapitre est divisé en trois parties : la première est dédiée à la définition et aux propriétés de la confiance, la seconde présente des méthodes de mesure de la confiance entre utilisateurs, et la troisième décrit l'intégration de la confiance dans les systèmes de recommandation.

Plan du chapitre. La section 4.1 présente la définition et les propriétés de la confiance. Ensuite, la section 4.2 est dédiée aux méthodes de mesure des différentes catégories de la confiance : explicite, inférée et implicite. Pour finir, dans la section 4.3, nous décrivons l'intégration des informations sur la confiance dans le filtrage collaboratif.

4.1 Définition et propriétés de la confiance

Dans cette section, nous fournissons un aperçu de la confiance, ses propriétés dans les médias sociaux en général et les systèmes de recommandation en particulier. Ces informations sur la confiance serviront de base pour améliorer les systèmes de recommandation classiques, qui seront décrits dans la Section 4.3.

La confiance, dans le contexte social, désigne généralement le fait qu'une personne (*trustee*) se fie aux paroles et aux actes d'une autre personne (*trusted*). Dans les systèmes de recommandation, il est défini en fonction de la capacité d'un utilisateur (*trusted*) à fournir de bonnes recommandations à un autre utilisateur (*trustee*) [51].

4.1.1 Propriétés de la confiance

La valeur de la confiance peut être soit un nombre binaire, soit un nombre réel compris dans l'intervalle [0, 1], 0 et 1 signifient respectivement "*pas de confiance*" et "*confiance totale*". Lorsqu'on considère la méfiance (*distrust*), on ajoute les valeurs négatives de la confiance dans la plage [-1, 0[. Par conséquent, la confiance et la méfiance sont représentées dans l'intervalle [-1, +1]. L'usage des valeurs binaires est le moyen le plus simple d'exprimer la confiance ; soit un utilisateur fait confiance à un autre, soit il ne lui fait pas confiance.

Le réseau de confiance (*trust network*) est un graphe orienté et pondéré dans lequel les nœuds représentent les utilisateurs et les arcs représentent les relations de confiance entre utilisateurs. Le poids d'un arc indique le degré de confiance que l'utilisateur source accorde au destinataire.

Propriétés de la confiance. Lorsqu'on se fie à la confiance entre les personnes, il en ressort que la confiance a les propriétés suivantes : la transitivité, l'asymétrie, la

dépendance au contexte et la personnalisation [51]. Ces propriétés constituent la base de la création des algorithmes de mesure de la confiance.

- **Transitivité** : La transitivité est une propriété clé de la confiance. Elle permet de propager la confiance explicite le long des chemins du réseau de confiance pour atteindre d'autres utilisateurs. Par exemple, si A fait confiance à B, et B fait confiance à C, alors on déduire que A fait confiance à C.
- **Asymétrie** : La confiance est une relation subjective et personnelle entre utilisateurs. Par conséquent, c'est une relation orientée dans les réseaux sociaux. Ainsi, si l'utilisateur u fait confiance à l'utilisateur v, cela ne veut pas dire que v fait confiance à u.
- **Dépendance au contexte** : La confiance dépend du contexte, par exemple, un utilisateur digne de confiance en technologie peut ne pas l'être en gastronomie.
- **Personnalisé** : Le poids de la confiance qu'un utilisateur u accorde à un utilisateur v peut être différent du degré de confiance que l'utilisateur w accorde à v.

4.1.2 Confiance globale et confiance locale

Certaines métriques de la confiance permettent de calculer le poids de la confiance que les utilisateurs s'accordent les uns aux autres de manière individuelle. De ce point de vue, la confiance est un attribut local [92]. Outre cette vue locale, la confiance peut également être calculée comme une mesure globale ; chaque utilisateur ayant une valeur globale qui indique sa fiabilité dans l'ensemble du réseau [91].

Mesure de la confiance locale. Les métriques de la confiance locale utilisent l'opinion subjective de chaque utilisateur pour prédire le poids de la confiance qu'il accorde aux autres utilisateurs. Dans un réseau de confiance de taille n, pour une mesure de la confiance locale, chaque utilisateur accorde un poids de confiance aux $(n-1)$ autres utilisateurs.

Mesure de la confiance globale. La mesure de la confiance globale à un utilisateur u représente l'opinion que tous les utilisateurs du système ont à propos de u. Ainsi, chaque utilisateur n'a qu'une seule valeur qui représente son niveau de fiabilité dans le système. On conclut que la confiance globale viole la propriété de personnalisation.

Comparaison entre confiance locale et confiance globale. Les métriques de la confiance globale sont plus simples et moins coûteuses en temps de calcul que les métriques locales, car les métriques locales sont calculées pour chaque paire d'utilisateurs. Cependant, les modèles de la confiance locale apportent de la personnalisation sur des utilisateurs controversés (fiables pour certains, et pas fiables pour d'autres).

4.2 Mesure de la confiance

4.2.1 Confiance explicite et confiance inférée

Confiance explicite. Dans un réseau d'utilisateurs, il est possible de demander à chaque utilisateur d'évaluer les autres, en attribuant à chacun d'eux une valeur qui indique sa fiabilité. Cependant, sur les plateformes réelles, un utilisateur n'évalue qu'une faible proportion des autres utilisateurs. Cela peut être dû à la paresse, au manque de temps ou au fait qu'un utilisateur a une opinion directe uniquement sur un petit nombre d'utilisateurs. Par exemple, sur Epinions [4], le nombre moyen d'utilisateurs évalués par un utilisateur quelconque est de 1.7 [131, 92]. On conclut que la matrice associée au réseau de confiance explicite est creuse.

Confiance inférée. Pour étendre la couverture du réseau de confiance, des techniques sont conçues pour prédire la fiabilité des utilisateurs inconnus (pour qui on n'a pas donné d'opinion). De telles techniques visent la prédiction du score de la confiance de toutes les paires de nœuds du réseau à partir des informations existantes sur la confiance explicite. Il est question de déduire les valeurs de la confiance entre d'autres paires de nœuds, dont le lien n'est pas présent dans le réseau de confiance.

Étant donné un réseau d'utilisateurs, la transitivité est l'hypothèse courante dans toutes ces solutions pour propager la confiance [103]. La méthode la plus intuitive pour déduire le poids de la confiance qu'un utilisateur s accorde à un utilisateur d est la suivante : s interroge chacun de ses voisins pour obtenir le poids de la confiance qu'ils accordent à d. Chaque voisin répète le même processus. Chaque fois qu'un nœud reçoit des poids de plusieurs voisins, il utilise la moyenne pondérée par la confiance en ces voisins. Si on est dans un contexte de valeurs binaires, alors la moyenne est arrondie à 0 ou 1.

4.2.2 Confiance implicite

Dans les modèles de confiance implicite, il n'y a pas d'information sur la confiance explicite et l'objectif est d'utiliser d'autres données pour construire un réseau avec des valeurs de confiance entre utilisateurs. Dans [156], les auteurs montrent qu'il existe une relation entre les préférences similaires des utilisateurs et la confiance entre eux. Cela signifie que des personnes qui partagent les mêmes centres d'intérêts et les mêmes goûts ont tendance à avoir davantage confiance les unes en les autres. Il est donc raisonnable d'utiliser des mesures de la similarité des préférences des utilisateurs pour déduire les valeurs implicites de la confiance entre eux. Ces techniques sont en majorité basées sur la similarité des profils des utilisateurs et l'historique des notes explicites.

4. http ://www.epinions.com

Similarité des profils utilisateur. La similarité entre deux utilisateurs est définie en fonction du fait qu'ils sont liés dans un réseau social, qu'ils ont des amis en commun ou qu'ils aiment les mêmes produits ou catégories de produits [141, 150].

Historique des notes explicites. La similarité est grande entre deux utilisateurs, s'ils notent les mêmes produits de la même manière. Les utilisateurs qui attribuent des notes similaires sont plus susceptibles de se faire confiance [110, 51].

Il existe une différence entre la confiance implicite et les valeurs inférées de la confiance explicite. Dans la confiance inférée, on utilise la transitivité pour propager les informations de confiance explicite existantes et déduire les poids qui manquent dans le réseau de confiance. Par contre, pour la confiance implicite, on n'utilise pas la confiance explicite.

Cette section était consacrée à la mesure de la confiance dans les médias sociaux. Nous avons donné une vue d'ensemble sur les mesures de la confiance explicite, inférée et implicite. Dans la section suivante, nous décrivons comment ces mesures de la confiance sont utilisées pour améliorer les techniques du filtrage collaboratif.

4.3 Intégration de la confiance dans le filtrage collaboratif

Les utilisateurs peuvent être influencés par leurs amis et sont plus susceptibles d'accepter les recommandations de ces derniers que celles des étrangers. Des études confirment également que les gens ont tendance à se fier davantage aux recommandations de ceux à qui ils font confiance, qu'aux recommandations fournies par les systèmes de recommandation [126]. Cet argument est suffisant pour penser que l'intégration des informations sur la confiance dans les systèmes de recommandation peut améliorer la qualité des recommandations ainsi que la relation client.

Les systèmes de recommandation basés sur la confiance reposent sur les mesures de la confiance et les techniques du filtrage collaboratif [137]. Ces systèmes utilisent les informations sur la confiance pour générer des recommandations plus personnalisées dans l'espoir d'améliorer la qualité, la couverture des utilisateurs (proportion d'utilisateurs pour qui il est possible de calculer des recommandations), et de relever certains des défis du filtrage collaboratif (démarrage à froid des utilisateurs, données creuses). Comme le filtrage collaboratif est divisé en deux catégories (une basée sur la mémoire et l'autre basée sur un modèle), de même, les systèmes de recommandation basés sur la confiance conservent ces deux catégories.

4.3.1 Filtrage collaboratif basé sur la mémoire et sur la confiance

Cette approche est différente du traditionnel filtrage collaboratif basé sur la mémoire car le système se concentre plus sur les utilisateurs fiables et moins/pas sur les autres utilisateurs. Le but est de renforcer les recommandations des utilisateurs fiables et pénaliser les recommandations des autres utilisateurs. Les informations sur la confiance sont utilisées de deux manières : soit pour que le système ne considère que les utilisateurs fiables (filtrage), soit pour pondérer les recommandations des utilisateurs (phase de collaboration).

Filtrage basé sur la confiance. Dans le filtrage basé sur la confiance, pour calculer les recommandations d'un utilisateur u, on ne calcule pas son voisinage, mais on utilise directement l'ensemble des utilisateurs fiables de u. Le système les fait collaborer pour déduire les recommandations à fournir à u. Dans le cas où les valeurs de la confiance sont des nombres réels, le système sélectionne uniquement les utilisateurs dont le poids de la confiance est supérieur à un certain seuil τ. L'équation 4.1 illustre le calcul de la préférence d'un utilisateur u pour un produit i dans une technique orientée utilisateurs, τ étant le seuil de la confiance.

$$preference(u,i) = \frac{\sum_{t_{u,u'}>\tau} R_{u',i}}{\sum_{t_{u,u'}>\tau} 1} \tag{4.1}$$

Pondération basée sur la confiance. Dans cette option, le système utilise les informations sur la confiance pour pondérer les recommandations formulées par les utilisateurs à l'endroit de l'utilisateur cible. Les valeurs de la confiance sont utilisées pour renforcer les recommandations des utilisateurs fiables, et réduire l'influence des autres utilisateurs. Il est également possible de filtrer et pondérer grâce aux informations sur la confiance. L'équation 4.2 illustre la pondération basée sur la confiance et l'équation 4.3 présente un cas d'utilisation de la confiance pour le filtrage et la pondération dans une technique orientée utilisateurs.

$$preference(u,i) = \frac{\sum_{u'} R_{u',i} \times t_{u,u'}}{\sum_{u'} t_{u,u'}} \tag{4.2}$$

$$preference(u,i) = \frac{\sum_{t_{u,u'}>\tau} R_{u',i} \times t_{u,u'}}{\sum_{t_{u,u'}>\tau} t_{u,u'}} \tag{4.3}$$

4.3.2 Filtrage collaboratif basé sur un modèle et sur la confiance

Dans cette catégorie, les techniques de factorisation matricielle sont largement utilisées. La logique commune à ces solutions est que les préférences des utilisateurs sont similaires à, ou influencées par, ceux des utilisateurs fiables. Les systèmes de cette catégorie peuvent être divisés en trois groupes : les méthodes de co-factorisation, les méthodes ensemblistes et les méthodes de régularisation [133].

Méthodes de co-factorisation. L'hypothèse sous-jacente des systèmes de ce groupe est qu'un utilisateur doit partager le même vecteur des préférences utilisateur dans l'espace des notes et dans l'espace des relations de confiance. Ces systèmes procèdent à une co-factorisation de la matrice des notes utilisateur-produit et de la matrice de relation de confiance utilisateur-utilisateur en partageant les mêmes facteurs latents pour apprendre les préférences des utilisateurs [131].

Méthodes ensemblistes. l'idée de base des méthodes ensemblistes est que les utilisateurs d'un même réseau de confiance doivent attribuer des notes similaires aux produits, et une note manquante d'un utilisateur donné est prédite comme une combinaison linéaire des notes des utilisateurs de son réseau de confiance [132].

Méthodes de régularisation. les méthodes de régularisation se concentrent sur les préférences d'un utilisateur, et supposent que celles-ci doivent être similaires à celles de son réseau de confiance. Pour un utilisateur donné u, les méthodes de régularisation forcent les préférences de u à se rapprocher de celles des utilisateurs de son réseau de confiance. Par exemple, SocialMF [69] oblige les préférences d'un utilisateur à se rapprocher de la préférence moyenne de son réseau de confiance.

4.3.3 Avantages des recommandations basées sur la confiance

L'idée principale des recommandations basées sur la confiance est d'améliorer la qualité des recommandations en résolvant certaines limites du filtrage collaboratif, notamment le problème du démarrage à froid. Dans la suite, nous listons certains avantages de l'intégration de la confiance.

Atténuer le démarrage à froid. Les nouveaux utilisateurs n'ont que quelques données, voire aucune. Les systèmes de recommandation classiques peuvent échouer à faire des recommandations pour ces utilisateurs. Cependant, les systèmes de recommandation basée sur la confiance peuvent faire des recommandations tant qu'un nouvel utilisateur a des relations de confiance avec d'autres utilisateurs connus du système [92].

Robuste aux attaques de faux profils. En utilisant les informations sur la confiance, il est possible de filtrer des faux profils car personne ne fait confiance à un faux profil. Ainsi, le système n'utilisera pas ces utilisateurs pour calculer des recommandations [1].

Augmenter la couverture des recommandations. Dans le filtrage collaboratif classique, nous n'avons accès qu'aux voisins immédiats qui ont attribué des notes similaires aux mêmes produits. Mais les systèmes basés sur la confiance utilisent plus d'informations pour construire la matrice des similarités. Par conséquent, nous avons accès à plus d'utilisateurs [67].

4.4 Résumé

Les techniques du filtrage collaboratif ont des limites qui sont liées notamment au démarrage à froid, le manque de données et la corruption des recommandations par la présence de faux profils. Pour apporter des solutions à ces problèmes et grâce à la présence des données sur les relations sociales des utilisateurs, plusieurs techniques ont été proposées pour avoir des valeurs explicites, inférées et implicites de la confiance. Puis ces valeurs sont intégrées aux techniques du filtrage collaboratif pour constituer les systèmes de recommandation basés sur la confiance qui améliorent la qualité des prédictions et la couverture des utilisateurs.

Les systèmes de recommandation présentés dans ce chapitre ne tiennent pas compte de la dynamique temporelle des actions des utilisateurs. Ceci veut dire que ces techniques ne font aucune différence entre les données récentes et les données plus anciennes et ne font également aucune différence entre le comportement des utilisateurs en fonction des moments de la journée ou des saisons. La prise en compte de tels aspects a pourtant un impact considérable dans les systèmes de recommandation. Le chapitre suivant est dédié aux systèmes de recommandation qui considèrent la dimension temporelle.

Chapitre 5

Filtrage collaboratif avec dynamique temporelle

Sommaire

Les premiers systèmes de recommandation ne tenaient pas compte des aspects temporels des informations utilisées. L'une des hypothèses implicites était que les goûts et les préférences ne changent pas significativement au cours du temps. Cette hypothèse s'avère fausse dans la réalité. Par exemple, l'intérêt accordé à un nouveau film n'est pas le même plusieurs années après sa sortie. De plus, le profil d'un individu peut changer et impliquer systématiquement le changement de ses préférences. Pour prendre en compte de tels changements, de nouvelles conceptions de systèmes de recommandation ont été élaborées et contribuent à une amélioration de la qualité des recommandations.

Un grand nombre de méthodes sur la modélisation et l'exploitation des informations temporelles a été proposé dans la littérature sur les systèmes de recommandation. Les

premières approches prenant en compte les informations temporelles dans les systèmes de recommandation remontent à 2001. Par exemple les travaux de Zimdars, Chickering et Meek [158] ont considéré le problème des recommandations comme un problème de prédiction de séries chronologiques. Cependant l'usage des informations temporelles a surtout attiré beaucoup d'attention grâce aux travaux de Adomavicius et Tuzhilin [5], et d'avantage encore après la victoire de Koren, Bell et Volinsky [79] au grand challenge Netflix sur les systèmes de recommandation [1].

Les systèmes de recommandation qui tiennent compte du temps ont diverses façons de considérer la dimension temporelle et diverses manières d'intégrer la dynamique associée. Certains font varier l'importance des données utilisées par le système de recommandation : soit cette importance varie en fonction du contexte temporel, soit cette importance décroit dans le temps. D'autres ne font pas varier l'importance des données, mais procèdent à une configuration dynamique des paramètres du système de recommandation dans le temps, en fonction des performances. Ce chapitre porte sur les systèmes de recommandation qui intègrent la dimension temporelle.

Plan du chapitre. Nous commençons par présenter dans la section 5.1 les différentes façons d'intégrer la dynamique temporelle dans les techniques de filtrage collaboratif en prenant le soin à chaque fois de décrire les informations temporelles considérées. Dans la section 5.2, nous donnons un aperçu des systèmes de recommandation avec dynamique temporelle qui intègrent d'autres types d'informations secondaires (information sur le contenu des produits et sur la confiance). Pour finir, dans la section 5.3, nous décrivons les méthodes d'évaluation des systèmes de recommandation qui tiennent compte du temps.

5.1 Dynamique temporelle des systèmes de recommandation

Les informations sur le temps ont l'avantage d'être faciles à collecter, car presque toutes les plateformes peuvent enregistrer les horodatages des actions des utilisateurs sur les produits : les achats dans les sites de vente en ligne, la lecture des fichiers vidéos et audio sur les sites de streaming et l'ouverture des pages web. En effet, par exemple cette collecte ne nécessite pas d'efforts supplémentaires des utilisateurs, et n'exige pas de nouveaux systèmes ou périphériques particuliers.

Dans cette section, nous présentons d'abord les types d'informations temporelles couramment utilisés (section 5.1.1) et les stratégies employées pour prendre en compte ces données dans les systèmes de recommandation, à savoir la dynamique en fonction du contexte temporel (section 5.1.2), la décroissance de l'importance des données dans le

1. https ://www.netflixprize.com/

temps (section 5.1.3) et la configuration dynamique des systèmes de recommandation (section 5.1.4).

5.1.1 Informations sur le temps

A partir des horodatages enregistrés, on peut déduire des informations et des considérations liées au temps. Les systèmes de recommandation qui intègrent les aspects temporels utilisent ces informations dans au moins une étape de leur processus de recommandation, et sont ainsi capables de fournir des recommandations différentes en fonction de l'instant auquel les recommandations sont proposées. En d'autres termes, la liste des produits recommandés aujourd'hui dans l'après midi, peut être différente de celle d'hier ou même de celle de ce matin. De manière générale, les informations sur le temps sont soit liées au contexte temporel, soit liées à la durée des données.

Informations liées au contexte temporel

Dans la prise en compte de la dimension temporelle on considère les propriétés du contexte temporel telles que la période de la journée, le jour de la semaine et la saison de l'année. Ces différentes propriétés proviennent très souvent de la liste suivante.

- les étiquettes de la structure hiérarchique des unités de temps, par exemple : les "heure"-[00H00, 01H00, ... , 23H00], "jour"-[Lundi, Mardi, ... , Dimanche], "mois"-[Janvier, Février, ... , Décembre] ;
- la caractérisation des moments de la journée, par exemple : "matinée", "après-midi" et "soirée" ;
- les moments clés du fonctionnement habituel des humains, par exemple : "vacances", "jour ouvrable" et "week-end" ;
- les saisons de l'année, par exemple : "saison de pluie", "saison sèche" et "hiver".

Ces valeurs catégorielles sont liées soit par une relation de précédence (lundi précède vendredi), soit par une relation hiérarchique (un jour appartient à une semaine et à un mois), et le fait que le temps soit un continuum conduit à une conception cyclique du temps où ses valeurs se répètent périodiquement.

Considération des durées

Lorsque ce n'est pas le contexte temporel qui est utilisé, un grand nombre de systèmes de recommandation considèrent le temps de manière continue. Les horodatages sont conservés et convertis en nombre entier de manière à faciliter le calcul des durées. Dans ces cas, on se sert par exemple de la durée entre l'instant pour lequel on calcule les recommandations et la date :

– d'ajout du produit sur la plateforme ;

– de fabrication, ou de sortie ou de publication ;

– de la dernière action de l'utilisateur ;

– de la dernière action sur le produit concerné.

C'est en fonction des durées calculées qu'une donnée ou un produit peut être estimé comme pertinent ou pas, dans le processus de recommandation en cours.

Remarquons que d'autres sources d'informations temporelles peuvent être collectées et exploitées. Par exemple les instants des événements intéressants comme l'anniversaire de l'utilisateur, la date de son enregistrement dans la plateforme ou les dates d'ajout des produits dans le système.

5.1.2 Dynamique en fonction du contexte temporel

L'importance des données suivant le contexte temporel s'illustre facilement par des exemples : les vêtements les plus intéressants en hiver ne sont pas les plus intéressants en été. De même, les chansons qu'on aime écouter en période de travail ne sont pas les mêmes qu'on aime écouter dans la soirée, une fois la journée de travail achevée. Des variations similaires peuvent aussi être observées selon qu'on est un jour ouvrable ou en week-end. Au vu de telles situations, la prise en compte de l'influence du contexte temporel sur l'intérêt accordé à un produit est importante pour améliorer la qualité des recommandations.

Dans ce type de systèmes, la dimension temporelle est modélisée comme une ou plusieurs variables discrètes liées au contexte temporel, ce qui permet de traiter les données différemment en fonction des valeurs contextuelles associées. Dans cette formulation, pour recommander à un instant t donné, le contexte temporel de t est pris en compte dans le processus de déduction des produits à proposés. Les systèmes de recommandation qui intègrent les informations du contexte temporel exploitent le temps par pré-filtrage et post-filtrage contextuels pour calculer $F(u, i, t)$ qui est l'estimation de la préférence de l'utilisateur u pour le produit i à l'instant t.

Dans le pré-filtrage, une dimension du contexte temporel peut être représenté par T^1 = {matinée, après-midi, soirée} ou par T^2 = {jour-ouvrable, week-end}. Ainsi, il n'est pas possible de classer les données au moyen de leur horodatage, c'est-à-dire qu'il n'y a pas de données plus anciennes ou plus récentes, mais plutôt des données associées à des contextes particuliers. Dans ce cas, lorsqu'il faut calculer $F(u, i, t)$, un filtre $f(u, i, t)$ peut être appliqué pour pénaliser les données qui ne sont pas pertinentes pour le contexte temporel de t. Le filtre le plus trivial est celui qui ignore toutes les données dont le contexte temporel n'a rien en commun avec celui de t.

Par exemple, dans [13], les auteurs ont créé des micro-profils contextuels, chacun contenant les notes d'un utilisateur dans un contexte particulier. Seuls les micro-profils

correspondant au contexte temporel cible sont utilisés pour calculer les recommandations. Les auteurs ont testé plusieurs schémas contextuels, tels que : les moments de la journée (matin, soir), la période de la semaine (jour ouvrable, week-end) et les saisons (hiver, été). Ces différents schémas ont amélioré la qualité des recommandations.

En ce qui concerne le post-filtrage contextuel, le filtre est utilisé pour adapter la valeur de $F(u, i)$ qui est l'estimation de la préférence de l'utilisateur u pour le produit i lorsque le temps n'est pas pris en compte. La valeur de $F(u, i, t)$ est alors fonction de $F(u, i)$ et $f(u, i, t)$. Un exemple de post-filtrage utilisant des variables du contexte temporel est donné dans [109]. Dans leur travail $F(u, i, t) = F(u, i)$ si $f(u, i, t)$ est supérieur à un seuil et $F(u, i, t) = 0$ dans le cas contraire. Avec $f(u, i, t)$ le nombre de voisins de u qui ont été favorables à i dans le même contexte temporel que celui de t.

Il est important de noter que lorsque les horodatages de données sont disponibles, alors plusieurs attributs du contexte temporel peuvent être exploités. Par exemple, dans [84], les auteurs déduisent les saisons (automne, hiver, printemps, été), les jours de la semaine (lundi, mardi, mercredi, jeudi, vendredi, samedi, dimanche), et moments de la journée (minuit, matinée, midi, après-midi, soir), et utilisent tous ces attributs simultanément pour le calcul des recommandations. De plus, compte tenu de la souplesse de la représentation discrète du contexte temporel, il est facile d'intégrer d'autres aspects en dehors des unités du temps ou des moments de la journée, à l'exemple des étiquettes comme "pâques" et "noël" [107].

Techniques basées sur un modèle avec prise en compte du contexte temporel

Les systèmes de recommandation basés sur un modèle et qui considèrent le contexte temporel sont construits à partir de données sur les préférences des utilisateurs et des attributs discrets du contexte temporel. L'un des premiers systèmes de ce type est celui proposé dans [104]. Les auteurs considèrent plusieurs dimensions contextuelles, notamment les moments de la journée et les saisons sont incorporées dans un modèle de machine à vecteurs de supports pour la recommandation des restaurants.

On note cependant que la plupart des travaux de cette catégorie n'utilisent pas uniquement les attributs liés au contexte temporel mais également des attributs du filtrage basé sur le contenu. Ainsi, on a un système de recommandation qui intègre simultanément plusieurs informations secondaires à savoir sur le contenu et sur le contexte temporel. Nous y reviendrons dans la section 5.3. Néanmoins, on peut citer [71] qui utilisent un modèle de *Tensor Flow* pour une modélisation multidimensionnelle des informations contextuelles.

Graphe de recommandation avec prise en compte du contexte temporel

Pour construire les graphes de recommandation de cette catégorie, on procède comme pour les graphes de recommandation qui intègrent les attributs du filtrage basé sur le contenu de la section 3.3.1 : on ajoute des nœuds qui correspondent aux attributs discrets

du contexte temporel. C'est le cas dans les travaux de Lee et al. [85] qui intègrent les saisons (automne, hiver, printemps, été) pour la recommandation de films, mais également d'autres attributs comme les positions géographiques des utilisateurs lorsqu'ils agissent. Une fois le graphe construit, les auteurs appliquent une personnalisation du PageRank [106] pour calculer les recommandations top-N.

5.1.3 Décroissance de l'importance des données dans le temps

Les systèmes de recommandation de la section précédente permettent de capturer des effets liés au contexte temporel. Ces systèmes ont un point de vue périodique des actions des utilisateurs. L'importance qu'un utilisateur accorde à un produit peut également diminuer avec le temps, en conséquence de la croissance et du changement de profil de l'utilisateur, mais également du fait qu'un grand nombre de produits se démodent. En d'autres termes, les sélections ou les achats de produits les plus récents d'un utilisateur reflètent mieux ses préférences actuelles. Afin de capturer ces effets de décroissance de l'importance au cours du temps, plusieurs stratégies ont été proposées.

Dans ce type de systèmes de recommandation, le temps est considéré comme étant une variable continue. La fonction $F(u, i, t)$ qui permet d'estimer la préférence de l'utilisateur u pour le produit i à l'instant t, est une fonction explicite de t, avec t un entier naturel qui représente le temps en secondes ou en nombre de jours, semaines, mois ou années. Les systèmes de cette catégorie peuvent être classés en trois sous-catégories : découpage du temps en tranches (préférence à court terme), combinaison des préférences à long et à court terme et enfin l'usage des fonctions de décroissance temporelle.

Découpage du temps en tranches

Dans l'usage du découpage du temps en tranches, on considère que l'importance des données est éphémère et ces données deviennent obsolètes au bout d'un certain temps. Ainsi, une fois que la taille des tranches de temps est fixée, chaque donnée est utilisée durant une seule tranche de temps et la tranche suivante est ignorée. Cette stratégie ne capture que les préférences à court terme [25, 138].

Par exemple Vinagre et Jorge [138] se servent du découpage du temps en tranches pour exprimer un mécanisme d'oubli d'un système de recommandation basé sur la mémoire. En d'autres termes, le système est capable d'ignorer des données anciennes et potentiellement obsolètes, tout en évitant des pertes de mémoire inutiles et du temps de calcul. Leurs résultats montrent que ce mécanisme permet de réduire les coûts des techniques de filtrage collaboratif basé sur la mémoire, tout en conservant la qualité des recommandations. Ce résultat est important pour le passage à l'échelle des techniques basées sur la mémoire.

Cependant, ces techniques on une limite liée au choix de la taille des tranches de temps. Pour faire simple, certains auteurs choisissent une taille qui correspond au fonctionnement périodique des humains dans la réalité. Par exemple Campos et al. [25] considère

uniquement les données qui datent d'une semaine pour le processus de recommandation des films. Leurs résultats montrent que l'utilisation des données proches de la date de recommandation peut aider à améliorer la qualité des recommandations, avec une fois de plus l'avantage de faciliter le passage à l'échelle.

Combinaison des préférences à long et à court termes

En considérant uniquement les préférences à court terme, le système de recommandation ignore le fait que les sélections effectuées par les utilisateurs sont également l'expression de préférences stables qui perdurent dans le temps : les préférences à long terme. Pour pallier à cette limite, de nouveaux systèmes de recommandation sont construits à la fois avec des données anciennes qui contiennent les préférences stables des utilisateurs, et les données récentes qui représentent leurs préférences à court terme. Ensuite, le système combine les deux types de préférences tout en accordant des poids différents aux données en fonction de leur catégorie, long terme ou court terme.

Par exemple les travaux Song et al. [128] proposent un système de recommandation basé sur un réseau de neurones dont l'architecture permet de modéliser la combinaison des préférences à long terme qu'ils considèrent comme statiques, avec les préférences à court terme qu'ils supposent dynamiques. De même, Yin et al. [147] étendent un modèle de factorisation matricielle en intégrant un modèle des préférences à court terme dans le modèle classique qui ne considère que les préférences à long terme. Dans ces travaux, on observe une amélioration des performances.

En ce qui concerne les graphes de recommandation, Xiang et al. [142] propose le *Session−based Temporal Graph* (STG) qui permet de capturer à la fois les préférences à long terme et les préférences à court terme. Il ajoute au graphe biparti classique entre utilisateurs et items un nouveau type de nœud qu'ils appellent nœud-session (*session node*), chacun de ces nœuds représente le comportement à court terme d'un utilisateur. Par exemple, le nœud (u, T) représente le comportement de l'utilisateur u durant la tranche de temps à court terme T. En effet, dans la mise en œuvre du STG, le temps est divisé en tranches de temps, et chaque tranche correspond à une session des préférences à court terme.

De manière générale, le processus de combinaison des préférences se fait par le biais d'une combinaison linaire. Par exemple, si on suppose que P_l et P_c sont respectivement les préférences à long terme et les préférences à court terme, la préférence d'un utilisateur u est représenté par $P(u) = \alpha \cdot P_l(u) + (1 - \alpha) \cdot P_c(u)$. Ce processus de combinaison donne naissance au problème du poids à accorder à chacun des types de préférences. L'autre problème dans cette approche est le choix de la taille des tranches de temps qui permettent de capturer les comportements à court terme.

Fonctions de décroissance temporelle

Dans l'usage des deux précédentes catégories de la décroissance de l'importance des données, les auteurs procèdent à un découpage du temps, imposant une conception discontinue du temps. Cependant, le processus suivant lequel les produits se démodent peut être progressif et continu. Ainsi, plusieurs systèmes de recommandation intègrent des fonctions de décroissance temporelle qui permettent de pondérer les données de telle sorte que leur poids diminue au fil du temps. Ces derniers restent fidèles à l'hypothèse selon laquelle les récentes sélections des utilisateurs reflètent mieux leurs goûts et préférences actuels [40, 78].

Supposons par exemple que $w(u,i)$ est le poids accordé au lien (u,i) dans un système de recommandation qui ne tient pas compte du temps. Dans un système de recommandation qui utilise une fonction de décroissance temporelle f, si on veut calculer une recommandation à l'instant t, le poids du lien (u,i) est typiquement modifié en $w_t(u,i) = w(u,i) \times f(t - t_{u,i})$, avec $t_{u,i}$ l'instant de création du lien (u,i). L'un des travaux de référence pour cette prise en compte du temps est celui de Ding et al. [40] qui propose une extension de la technique des K-plus proches voisins orientée produits du filtrage collaboratif par intégration des fonctions exponentielles décroissantes dans la phase de prédiction : $f(t - t_{u,i}) = e^{-\lambda \cdot (t - t_{u,i})}$. Dans cette formulation, la valeur de λ est calculée comme suit $\lambda = 1/\tau_0$, où τ_0 est la durée de demi-vie, c'est-à-dire que le poids de chaque donnée diminue d'environ $1/2$ après τ_0 unités de temps.

Comme exemple de système basé sur un modèle qui intègre les fonctions de décroissance, nous pouvons citer les travaux de Koren [78] sur la factorisation matricielle. Dans ce travail, Koren intègre lui aussi des fonctions exponentielles décroissantes pour pénaliser les poids des biais liés à chaque utilisateur et chaque produit qui sont présents dans son modèle de factorisation matriciel. Un autre exemple de système est celui proposé par Xiong et al. [143] qui est basé sur un modèle probabiliste bayésien qui repose sur la factorisation des *Tensor Flow* dans lequel les auteurs intègrent le temps comme vecteur de caractéristiques supplémentaire associé à chaque paire utilisateur-produit, au lieu de chaque utilisateur ou chaque produit comme dans le travail de Koren [78].

De manière générale, les systèmes de recommandation qui utilisent la décroissance de l'importance des données dans le temps, ont des problèmes de configuration relatifs aux choix des valeurs des paramètres ajoutés aux systèmes traditionnels qui n'intègrent pas la dynamique temporelle. Par exemple, le choix de la taille des tranches de temps qui correspondent à la durée du court terme ou encore le choix de la valeur de λ de la fonction $f(t - t_{u,i}) = e^{-\lambda \cdot (t - t_{u,i})}$. Ces paramètres sont très souvent choisis de manière empirique et parfois sans réelle justification.

5.1.4 Systèmes de recommandation avec configuration dynamique

Dans les précédentes sections, nous avons présenté des systèmes de recommandation qui modifient le poids des données utilisées pour tenir compte de la dynamique temporelle. Cependant, la qualité de prédiction d'un système de recommandation n'est pas seulement liée aux données utilisées mais dépend également de la configuration du système de recommandation. A cet effet, d'autres travaux se concentrent sur des mises à jour dynamiques des valeurs des paramètres de configuration.

Dans cette catégorie, on peut citer les travaux de Lathia et al. [82] dans lequel les auteurs divisent le temps en tranches et procèdent à un ajustement dynamique du nombre de voisins à utiliser dans la technique des K-plus proches voisins, tout ceci dans le but de diminuer les erreurs de prédiction précédentes.

Une autre forme de ce type de système adaptatif dans le temps est présentée par Jahrer et al. [66], où plusieurs modèles qui ne prennent pas le temps en compte sont construits à partir des jeux d'apprentissage différents. Les données sont réparties dans les jeux d'apprentissage grâce à des fonctions qui tiennent compte du temps ; des poids différents sont attribués à chaque jeu d'apprentissage. Ensuite tous les modèles construits sont utilisés pour la recommandation.

5.2 Intégration des autres informations secondaires

Dans les chapitres 2, 3 et 4, et dans les précédentes sections de ce chapitre, nous avons présenté des systèmes de recommandation de filtrage collaboratif, d'autres qui combinent les filtrages collaboratif et basé sur le contenu, sans oublier les systèmes de recommandations basés sur la confiance et ceux qui intègrent la dynamique temporelle. Cependant, nous ne nous sommes pas particulièrement attardés sur des systèmes de recommandations de filtrage collaboratifs qui intègrent à la fois plusieurs informations secondaires comme celles déjà mentionnées dans ce manuscrit, à savoir les attributs de description du contenu des produits, la confiance entre les utilisateurs et les informations extraites des horodatages des données.

Les systèmes de recommandation de cette catégorie ont pour objectif de tirer avantage de tous les types d'informations secondaires mentionnées dans le but de pallier aux limites du filtrage collaboratif et d'améliorer la qualité des recommandations. A notre connaissance, les systèmes de recommandation de filtrage collaboratif avec dynamique temporelle qui intègrent d'autres informations secondaires sont de deux catégories : filtrage collaboratif avec dynamique temporelle qui intègre des informations basées sur le contenu et le filtrage collaboratif avec dynamique temporelle qui intègre des informations sur la confiance entre les utilisateurs.

5.2.1 Informations temporelles et informations basées sur le contenu

Un grand nombre de travaux sur le filtrage collaboratif qui intègrent le temps en utilisant les informations sur le contexte temporel intègrent également des attributs basés sur le contenu. Ceci est simple car les informations utilisées sont matérialisées de la même manière. C'est le cas dans [71] où les auteurs utilisent une modélisation multidimensionnelle avec les *Tensor Flows*, ce qui leur permet d'intégrer les attributs du contexte temporel et les attributs de description des produits proposés aux utilisateurs. Cette extension a permis d'améliorer la qualité des recommandations.

D'autre part, certains travaux qui adoptent l'hypothèse de la décroissance de l'importance des données dans le temps, intègrent également des attributs basés sur le contenu des produits. On a par exemple les travaux de Song et al. [128] qui propose une architecture de réseau de neurones qui intègre les attributs liés à la description des produits mais aussi des propriétés liées au comportement statique à long terme des utilisateurs et d'autres liées au comportement dynamique (à court terme) des utilisateurs. Toujours dans la combinaison des préférences à long terme et à court terme, s'ajoutent les travaux de Yu et al. [148] qui proposent une extension du STG de Xiang et al. [142] pour la recommandation personnalisée des tweets en intégrant un nouveau type de nœud qui permet de représenter les catégories des sujets abordés dans les tweets.

En ce qui concerne le découpage du temps en tranches, on peut citer [98] où les auteurs utilisent un tel mécanisme et considèrent les attributs des chansons comme l'auteur de la chanson et la taxonomie des genres musicaux pour améliorer le processus de recommandation des chansons. Pour l'usage des fonctions de décroissance temporelle, [152] proposent de les utiliser tout en intégrant les étiquettes que les utilisateurs attribuent aux produits.

Notons que certains travaux avec dynamique temporelle qui incorporent les attributs de description des produits, utilisent simultanément plusieurs manières de prendre en compte le temps. En d'autres termes, ces systèmes peuvent intégrer le découpage du temps en tranches, la combinaison des préférences à long terme et à court terme, mais également des fonctions de décroissance temporelle. Parmi ces méthodes, on peut citer [88] pour la recommandation de pages web, et [87] pour la recommandation d'articles sur l'actualité.

5.2.2 Informations temporelles et informations sur la confiance

Les systèmes de recommandation de cette catégorie prennent en compte la dynamique temporelle et la confiance entre les utilisateurs. Ceci permet de pallier à des limites du filtrage collaboratif basé uniquement sur la confiance, notamment la prise en compte de l'évolution des préférences des utilisateurs, mais également l'intégration de l'évolution des

relations de confiance entre les utilisateurs.

En ce qui concerne la dynamique des préférences des utilisateurs dans les systèmes de recommandation basés sur la confiance, Zhang et al. [150] proposent une technique basée sur la factorisation matricielle pour les recommandations dans les réseaux sociaux. Ce système intègre des fonctions de décroissance temporelle pour pénaliser les données et les relations de confiance entre les utilisateurs pour personnaliser les recommandations. De même, Wei et al. [141] proposent un système similaire mais avec les K-plus proches voisins, dans un contexte de recommandation des pages web à partir des données du Web 2.0.

Pour la dynamique temporelle de l'évolution de la confiance entre les utilisateurs, nous citons par exemple [95, 119]. Dans [95], les auteurs font varier le poids de la confiance entre chaque paire d'utilisateurs en fonction de la date de création de leur relation d'amitié dans le système ; plus la relation d'amitié est vieille, plus la confiance est grande. Saito et al. [119] proposent un système de recommandation qui reflète des niveaux de confiance en incorporant un facteur de décroissance de la confiance dans une fonction temporelle.

5.2.3 Informations temporelles, sur le contenu et sur la confiance

A notre connaissance, il n'y a pas de systèmes de recommandation qui intègrent simultanément la dynamique temporelle, les attributs basés sur le contenu des produits et la confiance entre les utilisateurs. Pour tirer avantage de la combinaison de toutes ces informations secondaires, nous proposons dans le chapitre 7 un cadre conceptuel qui permet de construire des systèmes de recommandation basés sur les graphes et dans les lesquels on peut intégrer simultanément ces trois types d'information secondaires (attributs de description des produits, relations de confiance et dynamique temporelle).

5.3 Évaluation temporelle des recommandations

Dans ce chapitre, nous avons présenté un ensemble de travaux qui prennent en compte la dynamique temporelle des recommandations. Ainsi, le processus d'évaluation hors-ligne qui procède à une division des données en deux ensembles (jeu d'apprentissage et jeu de test) n'est plus adéquat car c'est une approche statique qui ne tient pas compte des dates des données. Grâce à la considération des horodatages des données utilisées, plusieurs protocoles et mesures ont été utilisés pour évaluer les recommandations temporelles. Dans cette section, nous présentons quelques exemples représentatifs de ces protocoles et mesures d'évaluation.

5.3.1 Stratégies de construction des jeux d'apprentissage et de test

Divers protocoles d'évaluation des systèmes de recommandation avec dynamique temporelle ont été mis au point. Les principales innovations sont au niveau de la construction des ensembles "jeu d'apprentissage" et "jeu de test", qui sont très importants dans le mécanisme d'évaluation des recommandations, car le premier est utilisé pour construire le système de recommandation et le second est utilisé pour évaluer les recommandations. Dans la suite nous présentons les différentes façons de construire le jeu d'apprentissage et le jeu de test en tenant compte du temps.

Taille fixe du jeu de test de chaque utilisateur

L'un des protocoles utilisés pour construire le jeu d'apprentissage et le jeu de test consiste à trier les liens (t, u, i) de chaque utilisateur dans l'ordre chronologique, puis de retenir un nombre fixe n_{jt} des liens les plus récents (pour chaque utilisateur) comme faisant partie du jeu de test et le reste des liens sont intégrés au jeu d'apprentissage. Par exemple, si n_{jt} est fixé à 3, alors les trois derniers liens de chaque utilisateur seront retenus dans le jeu de test et seront utilisés pour évaluer les recommandations qui seront faites. Ce protocole a été utilisé dans le cadre du concours Netflix [17]. La figure 5.1(a) présente un schéma de ce processus de fractionnement des données avec $n_{jt} = 3$, les triangles ombrés représentant les données affectées au jeu de test.

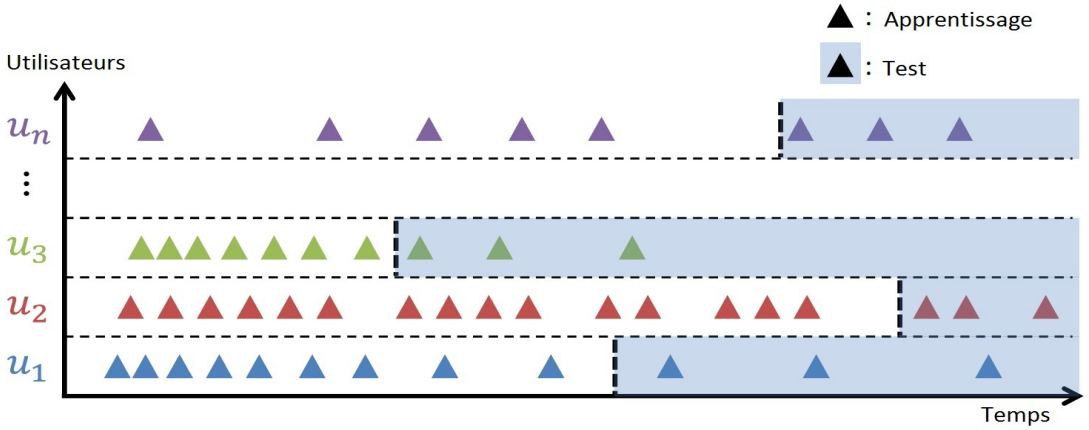
Cette méthodologie garantit que tous les utilisateurs ont un nombre égal de données pour l'évaluation. Cependant, les horodatages des données d'évaluation de certains utilisateurs peuvent avoir des dates inférieures (c'est-à-dire plus tôt) que les horodatages des données du jeu d'apprentissage des autres utilisateurs. Ce constat peut faire en sorte que des données du futur influencent des recommandations du passé, car en faisant collaborer les utilisateurs, les données du jeu d'apprentissage d'un voisin peuvent être plus récentes que celle du jeu de test de l'utilisateur cible. D'autre part, dans le cas où un utilisateur a peu de liens dans l'historique de ses actions, la sélection de n_{jt} liens de cet utilisateur dans le jeu de test peut faire en sorte que cet utilisateur ait peu ou pas de données dans le jeu d'apprentissage.

Séparation du jeu d'apprentissage et du jeu de test par une ligne temporelle

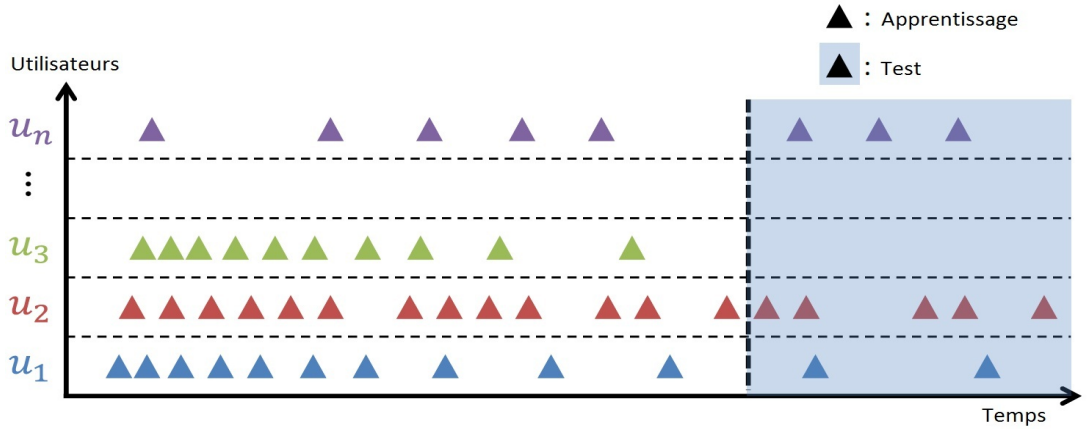
Un autre protocole très utilisé, notamment dans [82], tente de reproduire la division du jeu d'apprentissage et du jeu de test suivant le fonctionnement réel des systèmes de recommandation, c'est-à-dire en utilisant une division stricte du jeu de données en deux parties. Dans ce cas, une date particulière est fixée comme point de fractionnement et toutes les données antérieures à cette date sont utilisées comme données d'apprentissage, tandis que les données ultérieures à cette date sont utilisées comme données de test. Ceci

équivaut au déploiement du système de recommandation à la date fixée pour la division. Ainsi, toutes les données stockées par le système avant cette date sont utilisées pour calculer des prédictions, mais aucune donnée après cette date. La figure 5.1(b) montre un schéma de ce processus de division.

FIGURE 5.1 – Méthode de découpage des jeux d'apprentissage et de test en tenant compte du temps. Les triangles ombrés représentent les liens utilisateur-produit sélectionnés pour le jeu de test et le reste est attribué au jeu d'apprentissage [27].



(a) Découpage avec une taille fixe du jeu de test de chaque utilisateur, $n_{jt} = 3$.



(b) Découpage des jeux d'apprentissage et de test par une ligne temporelle.

Nous observons cependant qu'il existe un nombre variable de données de test pour chaque utilisateur. De plus, toutes les données des actions de certains utilisateurs peuvent être affectées au jeu de test s'ils ont sélectionné des produits uniquement après la date de la division. Inversement, toutes les données sur l'historique d'un utilisateur peuvent être insérées dans le jeu d'apprentissage s'ils ont sélectionné des produits uniquement avant la date de la division. Malgré ces remarques, nous observons que ce scénario est plus réaliste d'un point de vue temporel que celui décrit précédemment.

5.3.2 Métriques d'évaluation qui tiennent compte du temps

Dans la littérature, très peu de métriques d'évaluation prennent explicitement en compte le temps dans leur formulation. En général, les recommandations temporelles sont évaluées à l'aide de la moyenne d'une métrique traditionnelle d'évaluation comme celles présentées dans la section 2.3 du chapitre 2. Néanmoins, Lathia [83] propose des métriques qui contiennent le temps dans leur formulation et qui sont dédiées à la mesure de la précision, de la nouveauté et de la diversité des recommandations.

Pour le cas de la précision, Lathia propose un *Root Mean Square Error* (RMSE) pour l'évaluation des prédictions des notes et qui tient compte du temps. La nouvelle métrique est calculée comme le RMSE classique sur les notes attribuées jusqu'à un instant t donné. Cette métrique est destinée à être appliquée de manière itérative sur une longue période afin d'observer l'évolution de la précision des recommandations dans le temps.

Les métriques proposées par Lathia sont des alternatives aux métriques traditionnelles pour mesurer la qualité des recommandations temporelles. Cependant, ces métriques sont rarement utilisées probablement à cause de la difficulté à fournir une valeur unique qui résume la performance du système de recommandation dans le temps. Elles sont beaucoup plus adaptées à la configuration dynamique des systèmes de recommandation. C'est la raison pour laquelle, dans nos travaux, nous utilisons un mécanisme de combinaison linéaire des résultats des métriques traditionnelles d'évaluation, ce qui permet d'avoir une valeur unique qui résume les performances d'un système de recommandation au cours du temps. Nous présentons ce mécanisme dans la section 6.3.3 du chapitre 6.

5.3.3 Estimation de la fiabilité avec prise en compte du temps

Pour estimer la fiabilité des systèmes de recommandation, la division des données en deux jeux d'apprentissage et de test n'est pas suffisante. Il faut mettre en œuvre un autre mécanisme qui permettra d'évaluer plusieurs fois le système de recommandation afin d'avoir un meilleur aperçu de sa fiabilité. A cet effet, la méthode de validation croisée classique est déployée pour les systèmes de recommandation qui ne tiennent pas compte du temps (voir section 2.3 du chapitre 2). Cependant, la validation croisée classique peut conduire à des incohérences d'un point de vue temporel, notamment la prédiction des liens passés à partir des données du futur. Ainsi, de nouveaux mécanismes ont été proposés pour effectuer la validation croisée avec prise en compte du temps.

Les méthodes de validation croisée de cette catégorie visent à garantir le respect des conditions de dépendance temporelle entre les données des jeux d'apprentissage et de test. En d'autres termes, toutes les données du jeu de test doivent être plus récentes que les données du jeu d'apprentissage. Certaines de ces méthodes sont inspirées des méthodes de validation croisée indépendantes du temps et d'autres sont basées sur l'évolution dynamique des données dans le temps. Ces méthodes sont présentées dans les paragraphes ci-après.

Ré-échantillonnage en fonction du temps. Dans cette méthode, on effectue Z échantillonnages aléatoires en fonction de la taille du jeu de données qu'on désire. La particularité ici, est le fait que les données de chaque échantillon seront rangées dans l'ordre chronologique, puis une méthode de construction des jeux d'apprentissage et de test est appliquée. Cette méthode a été utilisée dans [59].

Ré-échantillonnage des utilisateurs en fonction du temps. Cette méthode est similaire au ré-échantillonnage des utilisateurs indépendant du temps, mais utilise une condition d'ordre des données en fonction du temps. On procède à Z échantillonnages aléatoires des utilisateurs qui seront considérés dans les jeux de données associés. Ensuite, les données de chaque échantillon sont rangées dans l'ordre chronologique. Cette méthode a été utilisée dans [36].

Fenêtres de temps de taille fixe. Cette méthode tient compte de l'évolution des données dans le temps. On fixe une durée pour le jeu d'apprentissage et une durée pour le jeu de test et la somme des deux durées donne la durée d_e de l'échantillon. En supposant que l'ensemble des données débute au temps t_0, le premier échantillon contient toutes les données de l'intervalle $[t_0, t_0 + d_e]$, le second échantillon est pris dans l'intervalle $]t_0 + d_e, t_0 + 2 \cdot d_e]$ ainsi de suite jusqu'au dernier échantillon qui est pris dans l'intervalle $]t_0 + (Z-1) \cdot d_e, t_0 + Z \cdot d_e]$. Cette méthode a été utilisée dans [115].

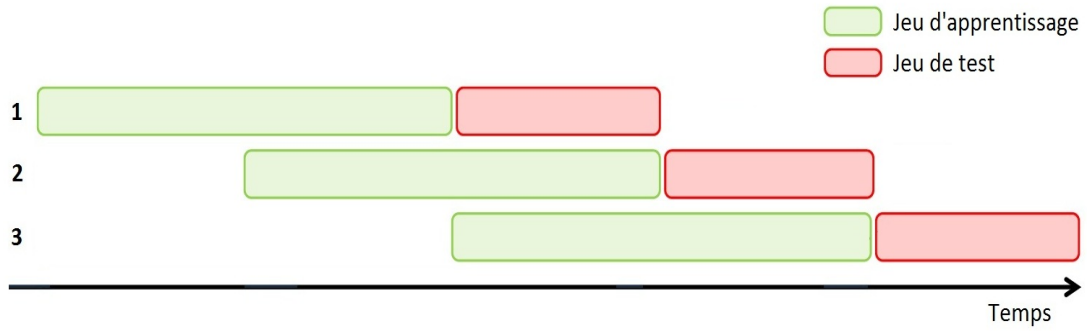
La figure 5.2(a) illustre ce processus de validation croisée, avec en vert le jeu d'apprentissage et en rouge le jeu de test. Contrairement aux méthodes précédentes, la durée de chaque jeu de données est fixe et une donnée appartient à un jeu de test au maximum. L'autre constat ici, c'est que les anciennes données sont ignorées au fur et à mesure. Ceci a le désavantage de faire perdre une partie des données d'apprentissage qui peuvent être précieuses pour certains systèmes de recommandation avec dynamique temporelle.

Fenêtres de temps de taille croissante. Cette méthode tient compte de l'évolution des données dans le temps, tout comme la précédente méthode. La principale différence est le fait que la taille du jeu d'apprentissage augmente avec le temps contrairement à la méthode précédente où cette taille est fixe. Pour mettre en œuvre cette méthode, on fixe la taille du jeu de test d_{jt} et éventuellement une taille initiale du jeu d'apprentissage d_{ja} (on peut utiliser celle du jeu de test), puis le premier échantillon est pris dans l'intervalle $[t_0, t_0+d_{ja}+d_{jt}]$, le second dans l'intervalle $]t_0+d_{ja}+d_{jt}, t_0+d_{ja}+2\cdot d_{jt}]$, ainsi de suite, et le dernier échantillon couvre les données de l'intervalle $]t_0+d_{ja}+(Z-1)\cdot d_{jt}, t_0+d_{ja}+Z\cdot d_{jt}]$. Cette méthode a été utilisée dans [82].

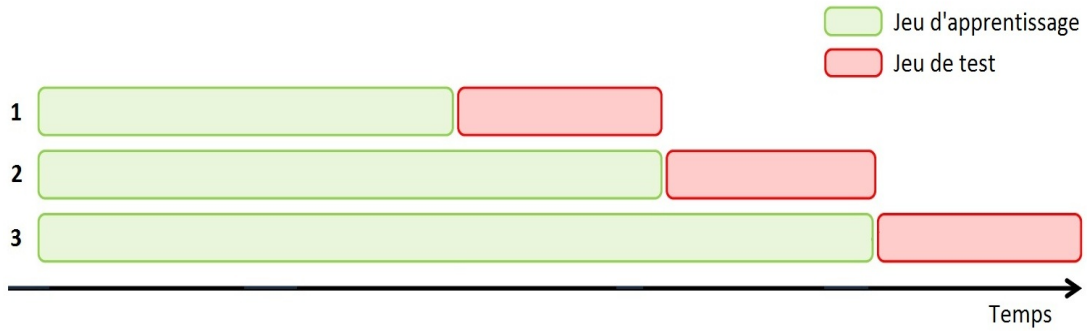
La figure 5.2(b) illustre cette méthode de validation croisée, avec en vert le jeu d'apprentissage et en rouge le jeu de test. On voit clairement la croissance de la taille du jeu d'apprentissage. Cette croissance du jeu d'apprentissage permet à chaque échantillon d'utiliser toutes les données disponibles à son rang et de simuler un comportement réel

des plateformes dans lesquelles le volume de données croit au fil du temps.

FIGURE 5.2 – Validation croisée avec prise en compte du temps. Les rectangles verts représentent les jeux d'apprentissage et les rectangles rouges représentent les jeux de test. Les jeux de données sont rangés dans l'ordre chronologique du haut vers le bas.



(a) Validation croisée avec fenêtre de temps de taille fixe.



(b) Validation croisée avec fenêtre de temps de taille croissante.

Les méthodes de validation croisée qui tiennent compte du temps que nous avons décrit dans cette section permettent de réduire la variabilité des résultats d'évaluation des systèmes de recommandation avec dynamique temporelle. Cependant la méthode qui est la plus proche de la réalité est celle des fenêtres de temps de taille croissante. En associant à cette méthode le mécanisme de séparation du jeu d'apprentissage et du jeu de test par une ligne temporelle comme décrit dans la section 5.3.1, on obtient un dispositif d'évaluation qui tient compte de l'évolution des données dans le temps et qui garantit à chaque fois que des données du futur ne sont pas utilisées pour prédire le passé. C'est ce dispositif que nous utilisons dans nos travaux dans les chapitres 6 et 7.

5.4 Résumé

De ce chapitre nous retenons que les premiers systèmes de recommandation de filtrage collaboratif ne tenaient pas compte des informations liées au temps, et donc ces systèmes ne pouvaient pas capturer des changements qu'on observe tant du coté utilisateurs, avec le changement de profil ou l'achat des produits selon le contexte temporel, que du coté produits car certains se démodent avec le temps. Pour pallier à ces insuffisances, de nouveaux systèmes de recommandation ont été proposés. Ces systèmes reposent souvent sur différentes considérations du temps, soit de manière catégorielle à travers des attributs du contexte temporel, soit de manière continue à travers des durées exprimées par exemple en nombre de jours, semaines ou mois. D'autre reposent sur la variation de l'importance des données.

La variation de l'importance des données est intégrée soit en fonction du contexte temporel, soit en supposant que cette importance décroit dans le temps. Lorsque l'importance des données varie en fonction du contexte temporel, on a par exemple des recommandations différentes selon que c'est l'hiver ou l'été. Lorsque l'importance des données diminue dans le temps, le système intègre le fait que les données récentes reflètent mieux les préférences actuelles des utilisateurs. Dans ce cas, soit le système considère uniquement les données récentes (préférences à court terme), soit le système combine les préférences à long terme et à court terme car certaines préférences de l'utilisateur sont stables dans le temps, soit le système intègre une fonction de décroissance temporelle pour diminuer progressivement les poids des données. D'autres systèmes de recommandation ne font pas varier les poids des données dans le temps mais procèdent plutôt à des configurations dynamiques de leurs paramètres.

Les systèmes de recommandation avec dynamique temporelle améliorent les techniques pures de filtrage collaboratif. La combinaison avec d'autres types d'informations secondaires comme les attributs basés sur le contenu et les informations sur la confiance permet de pallier à des problèmes de démarrage à froid, tout en augmentant la personnalisation des recommandations et par la même occasion la qualité des recommandations. Cependant, les travaux de ce type ne sont pas suffisamment poussés dans les graphes de recommandation : le domaine peut encore être enrichi par des stratégies de prise en compte de la dynamique temporelle et d'intégration simultanée de plusieurs informations secondaires. Ces observations motivent les contributions que nous présentons dans les prochains chapitres.

Deuxième partie

Graphes de recommandation enrichis

Chapitre 6

Dynamique temporelle dans les graphes de recommandation

Sommaire

Dans cette thèse nous abordons la question de dynamique temporelle des systèmes de recommandation basée sur les flots de liens. Cependant, les flots de liens et les graphes de recommandation mettent en relation les utilisateurs d'une part, et les produits d'autre part. La différence est le fait que les flots de liens conservent les horodatages des actions des utilisateurs sur les produits (sélection, achat, commentaire), tandis que la représentation par des graphes fait perdre la dynamique temporelle des actions des utilisateurs. Cette limite des graphes de recommandation peut être un frein pour l'obtention de meilleures qualités de recommandation car la prise en compte du temps dans les systèmes de recommandation a permis d'améliorer les performances comme présenté dans le chapitre précédent.

L'objectif de ce chapitre est de profiter de l'avantage des informations temporelles et des spécificités des flots de liens pour améliorer la prise en compte de la dynamique tempo-

relle dans les graphes de recommandations. A cet effet, nous présentons deux contributions de notre thèse. La première est une extension des graphes dynamiques de recommandation *Session−based Temporal Graph* (STG) de Xiang et al. [142] et *Topic−STG* de Yu et al. [148]. La seconde est un graphe de recommandation dont la structure considère le temps de manière continue contrairement aux graphes de recommandation antérieurs.

Plan du chapitre. Ce chapitre est structuré comme suit : en section 6.1 nous présentons les graphes de recommandation en fonction de leur considération du temps : d'une part ceux qui ignorent le temps et d'autre part ceux qui considèrent le temps de manière discontinue. Ensuite nous présentons notre première contribution dans la section 6.2. Le chapitre se termine par la proposition d'un nouveau graphe de recommandation dont la structure permet de considérer le temps de manière continue dans la section 6.3. Les deux dernières sections se déroulent suivant les points suivants : construction du graphe, point d'innovation du graphe et mise en œuvre.

6.1 Graphes de recommandation et dynamique temporelle

Dans cette section, nous présentons un état de l'art sur les graphes de recommandation en mettant un accent sur la considération des informations secondaires de manière générale et du temps en particulier. Nous présentons d'abord les graphes de recommandation qui ne tiennent pas compte du temps, ensuite, nous présentons ceux qui tiennent compte du temps.

6.1.1 Graphes sans prise en compte du temps

Sans prise en compte des informations secondaires

Le premier graphe à considérer dans cette catégorie est le graphe biparti classique, que nous désignons ici par BIP. C'est un graphe orienté biparti (U, I, E) où U est l'ensemble des utilisateurs, I l'ensemble des produits et $E \subseteq U \times I$ l'ensemble des arcs tel que les arcs (u, i) et (i, u) sont dans E si l'utilisateur u a sélectionné le produit i au moins une fois dans toute la période d'observation. Ce graphe est utilisé pour les recommandations dans [14].

A partir d'un graphe biparti, il est possible de déduire des graphes projetés sur les utilisateurs ou sur les produits. Dans un graphe projeté utilisateur, les nœuds représentent uniquement les utilisateurs et deux utilisateurs sont liés par une arête s'ils ont des produits en commun. De même dans un graphe projeté sur les produits, les nœuds représentent uniquement les produits et deux produits sont liés par une arête si au moins un utilisateur

a sélectionné ces deux produits. Les graphes projetés sont utilisés dans les systèmes de recommandation dans les travaux de Zhou et al. [154].

Prise en compte des informations sur le contenu

Les graphes bipartis et projetés dans leur représentation de base ne tiennent pas compte des caractéristiques des produits, alors que ces caractéristiques influencent souvent les choix des utilisateurs. Par exemple, il est fréquent d'écouter le nouvel album d'un musicien qu'on aime, comparé à un album d'un musicien qu'on ne connaît pas ou qu'on aime pas. Cette remarque a conduit à l'extension des graphes bipartis et des graphes projetés en tenant compte des attributs descriptifs des produits. La liste ci-après donne un aperçu de ces travaux.

- Huang et al. [63] proposent un graphe projeté sur les produits dans lequel les relations entre les produits ne sont pas définis par le fait qu'un utilisateur a acheté ces produits, mais par le fait que ces produits ont des caractéristiques descriptives en commun.
- Phuong et al. [114] proposent l'ajout d'un nouveau type de nœud "*content node*" pour représenter les attributs descriptifs des produits. Dans ce graphe, chaque "*content node*" est relié aux produits décrits par l'attribut correspondant.
- Bogers [21] proposent l'ajout de 3 types de nœuds pour la recommandation des films. Un nouveau type de nœud pour le genre de films (action, policier, romance), un nouveau type de nœud pour les acteurs des films et un nouveau type de nœud pour les étiquettes que les utilisateurs attributs aux produits. Les genres sont reliés aux films et aux étiquettes, les acteurs sont reliés aux films uniquement et les étiquettes sont reliées aux genres, aux films mais également aux utilisateurs.
- Shang et al. [123] proposent un graphe qui intègre des étiquettes qui permettent de décrire les produits, mais aussi des éléments des profils des utilisateurs. Les attributs des profils des utilisateurs sont reliés aux utilisateurs concernés, et les caractéristiques des produits sont liées aux produits concernés.
- Ostuni et al. [105] proposent de considérer les catégories des produits et de conserver la hiérarchie entre ces catégories. Ainsi, les nœuds qui représentent les catégories du plus bas niveau de la hiérarchie sont reliés aux produits de ces catégories, et les catégories sont reliées lorsque l'une est incluse dans l'autre.

Prise en compte des informations sur la confiance

Notons que très peu de travaux sur les graphes de recommandation considèrent les informations sur la confiance. Cependant, plusieurs recherches sur la mesure de la confiance inférée et la confiance implicite reposent sur l'analyse du réseau social formé par les utili-

sateurs [3, 70]. Néanmoins, Jamali et al. [67] proposent un graphe qui met en relation les utilisateurs et les produits, et qui relie les utilisateurs entre eux suivant leur affinité dans un réseau social. Dans cette représentation, les relations du réseau social sont assimilées aux relations de confiance entre les utilisateurs.

Dans les graphes de cette section, les horodatages présents dans les données sont ignorés et donc la dynamique temporelle des préférences des utilisateurs n'est pas prise en compte. Ce qui implique qu'il n'y a pas de différence entre les produits récents et les produits démodés.

6.1.2 Graphes avec prise en compte du temps

Sans prise en compte des autres informations secondaires

Pour tenir compte de la dynamique temporelle des préférences des utilisateurs, d'autres graphes de recommandation ont été proposés. Le plus populaire de ces graphes est le *Session−based Temporal Graph* (STG) de Xiang et al. [142], qui permet de modéliser à la fois les préférences à long terme et les préférences à court terme des utilisateurs. STG étend le graphe biparti classique et intègre la dynamique temporelle dans sa structure en y ajoutant des nœuds de type session (*session node*) qui représentent le comportement à court terme des utilisateurs.

Pour définir les nœuds session, l'intervalle d'observation T du flot de liens est divisé en tranches de temps de même taille Δ, avec Δ une durée fixée au préalable. On a donc les tranches de temps $T_k = [(k-1)\cdot\Delta, k\cdot\Delta]$. Dans le STG on a un nœud session (u, T_k) pour chaque utilisateur u et pour chaque tranche de temps T_k pendant laquelle u a été actif. Un nœud session $(u, T_k) \in S$ est relié à tous les produits que l'utilisateur u a sélectionnés durant la tranche de temps T_k. Un STG est donc un graphe biparti (U, I, S, E, w) où U, I, S et E sont respectivement l'ensemble des nœuds utilisateur, produit, session et l'ensemble des arcs du graphe. De plus, w est la fonction de pondération des arcs du STG.

Dans le graphe biparti classique, toutes les arêtes ont le même poids, mais dans le STG un arc $(x, y) \in E$, a un poids $w(x, y) = \eta_u$ si x est un nœud produit et y un nœud utilisateur, $w(x, y) = \eta_s$ si x est un nœud produit et y un nœud session et $w(x, y) = 1$ pour tous les autres cas. Xiang et al. simplifie la pondération en expliquant qu'on peut remplacer les valeurs de η_s et η_u respectivement par 1 et $\eta = \eta_u/\eta_s$ ou bien $\eta = \eta_s/\eta_u$ et 1. La figure 6.1 illustre un exemple de graphe STG.

D'autres graphes permettent de prendre en compte les informations sur le temps. Cependant, ces graphes considèrent uniquement le contexte temporel ce qui implique une représentation du temps de manière catégorielle. Par exemple Lee et al. [85] intègrent les saisons (automne, hiver, printemps, été) pour la recommandation des films, mais également d'autres attributs comme les positions géographiques des utilisateurs lorsqu'ils agissent. Les nœuds qui intègrent les saisons sont construits comme avec le STG, on a par

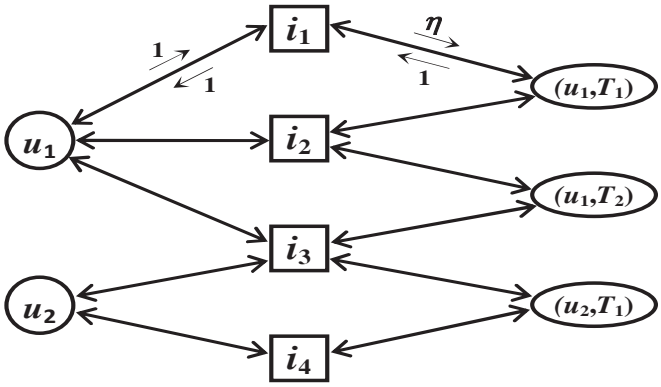


FIGURE 6.1 – Graphe STG (*Session−based Temporal Graph*) déduit du flot de liens $L = \{(t_1, u_1, i_1), (t_1, u_2, i_3), (t_2, u_1, i_2), (t_2, u_2, i_3), (t_3, u_2, i_4), (t_4, u_1, i_3), (t_5, u_2, i_4), (t_6, u_1, i_2)\}$, avec $\Delta = 3$, et donc deux tranches de temps T_1 et T_2.

exemple (u, "hiver") qui représente le comportement de l'utilisateur u pendant l'hiver.

Prise en compte des informations sur le contenu

Nous faisons le constat qu'il y a peu de travaux dans la catégorie des graphes de recommandation qui intègrent simultanément la dynamique temporelle et les informations sur le contenu. Néanmoins, nous pouvons citer *Topic*-STG proposé par Yu et al. [148], une extension du STG dédiée à la recommandation de *tweets*. Les auteurs ajoutent des nœuds de type *topic* dans le STG. Chaque nœud *topic* correspond à un thème abordé dans les *tweets*. Ces nœuds sont reliés à tous les *tweets* du thème correspondant. Les nœuds *topic* sont reliés aux *tweets* du thème correspondant, aux utilisateurs qui ont sélectionné ces *tweets* et aux nœuds session à court terme de ces utilisateurs.

Prise en compte des informations sur la confiance

A notre connaissance, il n'y a aucun travail sur les graphes de recommandation qui intègrent simultanément la dynamique temporelle et les informations sur la confiance. Pour aborder cette limite, nous proposons un mécanisme de calcul des recommandations qui tient compte des informations sur la confiance dans les graphes dynamiques. Nous y reviendrons dans le prochain chapitre consacré à la présentation de notre cadre conceptuel des systèmes de recommandation basés sur les graphes et qui peuvent combiner simultanément des informations secondaires sur le contenu, la confiance et le temps.

6.2 Time-weight Content-based Session-based Temporal Graph

Lorsqu'on met en œuvre les graphes de la section précédente, même ceux qui intègrent la dimension temporelle, on constate qu'il n'y a pas de différence de poids entre les arêtes les plus vieilles et les arêtes les plus récentes. Ceci veut dire qu'il n'y a pas une grande différence entre les actions anciennes des utilisateurs il y a un an et leurs récentes actions il y a une semaine. Ceci peut être un frein à l'augmentation de la qualité des recommandations car les actions récentes des utilisateurs reflètent mieux leurs préférences actuelles.

Pour pallier à cette limite, nous proposons de conserver la date de création de chaque arête et de faire diminuer les poids de ces arêtes en fonction de leur ancienneté. Pour ce faire, une fonction de décroissance temporelle peut être utilisée. Nous avons appliqué cette idée à une généralisation du *topic*-STG. Dans la suite de cette section, nous détaillons ce travail qui a été publié à la treizième édition de la conférence internationale *Web Information Systems and Technologies* (*WEBIST* 2017) [100].

6.2.1 Construction du graphe

Dans cette section, nous montrons d'abord comment construire le *Content−based STG* (CSTG), un STG basé sur le contenu similaire à *Topic*-STG de Yu et al. [148] mais qui n'est pas dédié uniquement à la recommandation des *tweets*. Ensuite, nous décrivons le *Time−weight Content−based STG* (TCSTG) que nous proposons.

Content-based Session-based Temporal Graph

Le modèle STG de base néglige les caractéristiques des produits, qui peuvent être des données importantes la recommandation. Pour prendre en compte ces données, nous ajoutons un type de nœud *content node* qui représente les attributs descriptifs des produits comme dans [114]. Chaque nœud qui représente une caractéristique des produits est relié aux nœuds des produits qui y sont associés et aux nœuds des utilisateurs qui ont déjà acheté au moins un produit ayant cette caractéristique.

Pour construire le STG basé sur le contenu, nous devons avoir des attributs descriptifs des produits dans les données manipulées. Par conséquent, nous n'utilisons pas un ensemble de triplets (t, u, i) mais un ensemble de quadruplets (t, u, i, c) qui exprime le fait que l'utilisateur u a sélectionné le produit i à l'instant t et que c est un attribut descriptif de i.

CSTG est un graphe orienté (U, I, S, C, E, w) obtenu à partir du STG (U, I, S, E, w) en ajoutant l'ensemble C des nœuds des attributs sur le contenu des produits et pour tout lien (t, u, i, c), six arcs supplémentaires sont ajoutés dans E du STG ; (u, c), (c, u),

$((u, T_k), c)$, $(c, (u, T_k))$, (i, c) et (c, i) avec des poids respectifs de 1, 1, 1, η, η_c et η_c. La figure 6.2 présente les nœuds, les arcs et la pondération qu'on a lorsqu'on a un nouveau quadruplet (t, u, i, c).

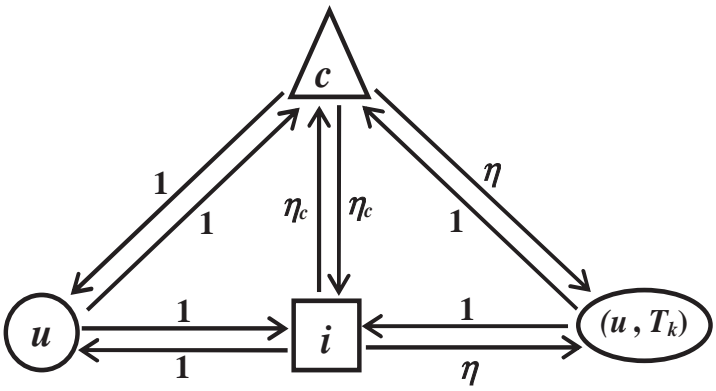


FIGURE 6.2 – Poids des arcs du graphe CSTG (*Content−based Session−based Temporal Graph*) suite à la présence du lien (t, u, i, c) avec $t \in T_k$.

Ajout des fonctions de décroissance temporelle dans les graphes de recommandation

Le CSTG intègre les attributs descriptifs des produits, cependant ce graphe accorde toujours la même importance aux arcs dans le temps. Il ne fait aucune différence de poids entre les arcs les plus vieux et les arcs les plus récents. Pour dépasser cette limite, nous avons fait le choix d'intégrer un mécanisme de pénalisation temporelle par le biais des fonctions décroissante comme l'ont proposé Ding et Li [40].

A cet effet, nous proposons de conserver la date t_e de création de chaque arc et lorsque nous calculons une recommandation à l'instant t, le poids initial de l'arc (x, y) est modifié comme défini par l'équation 6.1 :

$$w_t(x, y) = f(t - t_e) \cdot w(x, y) \tag{6.1}$$

où $w(x, y)$ est le poids initial de l'arc (x, y) et $w_t(x, y)$ est le poids du même arc à l'instant t. La fonction $f()$ est décroissante comme dans les travaux de Ding et Li [40]. Dans ce travail, nous avons utilisée une fonction très similaire à la fonction de décroissance de la radioactivité. En effet, $f()$ est défini ici par :

$$f(t - t_e) = e^{-\lambda \cdot (t - t_e)} \tag{6.2}$$

où λ est le taux de décroissance et $(t - t_e)$ est l'âge de l'arc concerné à l'instant auquel les recommandations sont calculées.

Le paramètre λ peut également être défini comme le ratio $1/\tau_0$ avec τ_0 un équivalent de la demi-vie radioactive, car lorsqu'un arc vieillit de τ_0 unité de temps, son poids diminue de moitié.

6.2.2 Calcul des recommandations

Dans les sections précédentes, nous avons présenté des graphes de recommandation et leurs extensions par intégration des informations basées sur le contenu et l'usage des fonctions de décroissance temporelle. Une fois que ces graphes sont construits, on applique un mécanisme de calcul des recommandations top-N. Dans cette section nous utilisons deux algorithmes de recommandation basés sur la marche aléatoire et proposés par Xiang et al. [142] : *Injected Preference Fusion* (IPF) et *Temporal Personalized Random Walk* (TPRW).

Temporal Personalized Random Walk

TPRW est une personnalisation du PageRank de Google proposé par Page et al. [106]. Initialement, le PageRank est proposé pour classer les pages web par ordre d'importance. En utilisant cet algorithme dans un système de recommandation, l'objectif est de classer les nœuds produits par ordre d'importance en fonction de l'utilisateur cible. Pour ce faire, un processus de personnalisation du PageRank doit être appliqué.

Dans le cas de TPRW, un processus de personnalisation du PageRank inspiré des travaux de Haveliwala [53] qui propose une configuration d'un vecteur du PageRank de manière à accroitre l'importance des pages web qui traitent d'un certain sujet. Ceci se traduit dans la configuration du vecteur d de l'équation suivante.

$$PR = \alpha \cdot M \cdot PR + (1 - \alpha) \cdot d \tag{6.3}$$

où PR est le vecteur PageRank qui contient l'importance de chaque nœud du graphe à la fin du processus de marche aléatoire, α est le facteur d'amortissement de la personnalisation, M est la matrice de transition du graphe considéré et d est le vecteur de personnalisation du PageRank. Le vecteur d indique les nœuds vers lesquels le marcheur aléatoire débutera une nouvelle marche après un redémarrage ; plus le poids d'un nœud est grand dans d, plus la probabilité que le marcheur aléatoire redémarre à ce nœud est grande, et donc les nœuds qui lui sont proches auront une grande importance.

TPRW personnalise le vecteur d de telles sortes que le marcheur aléatoire redémarre uniquement sur le nœud de l'utilisateur cible u et éventuellement sur son nœud session

le plus récent (u, T_k) (dans le cas du STG).

$$d_{BIP}(x) = \begin{cases} 1 & si\ x = u \\ 0 & sinon \end{cases} \tag{6.4}$$

$$d_{STG}(x) = \begin{cases} \beta & si\ x = u \\ 1 - \beta & si\ x = (u, T_k) \\ 0 & sinon \end{cases} \tag{6.5}$$

Ceci a pour conséquence de favoriser la recommandation des nouveaux produits proches du nœud u, qui représente les préférences à long terme, et du nœud (u, T_k) qui représente les préférences à court terme. A la fin du processus de convergence du PageRank personnalisé, les N produits les plus importants et nouveaux pour u lui sont proposés comme liste de recommandation top-N.

Injected Preference Fusion

IPF est un algorithme proposé par Xiang et al. [142] pour le calcul des recommandations top-N à partir des graphes de recommandation. C'est une extension de la marche aléatoire avec 2 particularités : premièrement, IPF ne considère que les plus courts chemins ; deuxièmement, le processus de propagation des poids n'est pas uniforme.

Pour recommander des produits à l'utilisateur cible u, IPF se déroule en 3 étapes :

- Injection des préférences à long terme β sur le nœud de l'utilisateur u et injection des préférences à court terme $(1 - \beta)$ sur le nœud session le plus récent (u, T_k) de l'utilisateur u.
- Propagation des préférences à travers une marche aléatoire qui ne considère que les plus courts chemins. La propagation d'un nœud v_k au nœud suivant v_{k+1} se fait suivant la formule ci-après.

$$\Phi(v_k, v_{k+1}) = \left(\frac{w(v_k, v_{k+1})}{\Sigma_{x \in out(v_k)} w(v_k, x)}\right)^{\rho} \tag{6.6}$$

  où $out(v_k)$ désigne l'ensemble des voisins sortants du nœud v_k, ρ est un paramètre pour ajuster le processus de propagation des poids, $w(v_k, v_{k+1})$ est le poids de l'arc (v_k, v_{k+1}) et $\Phi(v_k, v_{k+1})$ est la proportion de préférence de v_k qui est propagée à v_{k+1}.
- Recommandation des N premiers produits qui ont reçu les plus grandes valeurs de préférence et que l'utilisateur cible u n'a pas encore sélectionnés dans le passé.

Notons que dans [142], la longueur maximale de cette marche aléatoire est fixée à 3, car c'est à cette longueur que les meilleurs résultats sont atteints. Dans la suite, nous

utilisons IPF avec une longueur maximale de propagation toujours fixée à 3.

6.2.3 Expérimentations et résultats

Nous avons mené plusieurs expérimentations pour évaluer les performances de TCSTG comparé au STG et au CSTG. Pour chaque graphe, nous considérons plusieurs valeurs de paramètres et nous conservons les meilleures performances. Nous avons également implémenté le graphe biparti classique BIP pour montrer les effets de la prise en compte des préférences à court et à long termes dans les graphes de recommandation.

Description des données

Notre objectif est de calculer des recommandations top-N à partir des données implicites de diverses plateformes du monde réel. À cet effet, nous effectuons des expérimentations sur des jeux de données extraits des marque-pages sociales sur internet CiteUlike et Delicious qui ont été utilisés par Xiang et al. [142]. Ceci permet d'avoir une bonne base de comparaison entre STG et TCSTG que nous proposons. Nous utilisons également les données de Last.fm, un site web où les utilisateurs peuvent écouter de la musique car, dans ce domaine, l'effet de mode joue un rôle important, de sorte que l'impact des goûts passés sur les goûts futurs diminue considérablement avec le temps.

Nous modélisons nos données sous forme de flot de liens (t, u, i, c), où chaque quadruplet a une interprétation différente selon les domaines. Dans le cas de CiteUlike et de Delicious, chaque quadruplet signifie que l'utilisateur u a marqué la page i à l'instant t avec l'étiquette c. Et pour les données de Last.fm, cela signifie que l'utilisateur u a écouté la chanson i à l'instant t et c est l'auteur de i.

Avant de modéliser nos données sous forme de flot de liens, nous avons ignoré les produits et les utilisateurs qui n'apparaissaient pas un nombre de fois supérieur à un seuil σ fixé. Le tableau 6.1 fournit des détails sur les données utilisées : date du premier lien, date du dernier lien, seuil σ utilisé, nombre d'utilisateurs, nombre de produits, nombre d'attributs de description des produits et enfin le nombre de liens $|L|$.

Table 6.1 – Statistiques sur les jeux de données

	Date début	**Date de fin**	σ	**|U|**	**|I|**	**|C|**	**|L|**
CiteUlike	2010-01-01	2010-07-02	10	1 318	424	4 216	16 885
Delicious	2010-05-11	2010-11-09	7	894	298	2 789	13 825
Last.fm	2005-02-14	2005-08-16	8	135	1 054	225	41 604

Protocole expérimental et évaluation des recommandations

Nous utilisons le protocole de validation croisée à fenêtres de temps de taille croissante et la séparation des jeux d'apprentissage et de test par une ligne temporelle. Le flot de liens

est divisé en tranches de temps de taille 15 jours. Nous avons choisi 15 jours car cela est proche de certains caractères du comportement humain. Par exemple, les comportements des 15 premiers jours après l'obtention du salaire sont différents de ceux observés au cours des 15 derniers jours du mois. Pour simplifier le processus d'expérimentation, nous fixons également la taille du court terme des sessions du STG à 15 jours ($\Delta = 15 jours$).

Pour chaque fenêtre de temps W_k, pour $k = 1, .., Z-1$, avec Z le nombre de tranches, nous procédons comme suit :

- Construire les graphes correspondant aux données de W_1, W_2, .., W_k
- Calculez les recommandations Top-N pour les utilisateurs ayant sélectionné au moins un "nouveau produit" dans la tranche de temps suivante W_{k+1}.
- Évaluez les graphes de recommandation en calculant le Hit ratio [72], le ratio entre le nombre d'utilisateurs qui ont sélectionné au moins un produit recommandé dans la tranche de temps W_{k+1}. C'est le Hit ratio de la tranche de temps de rang k que nous notons HR_k.

Après avoir déterminé les valeurs des Hit ratio HR_k pour toutes les tranches de temps, nous calculons la moyenne temporelle de toute ces valeurs nommée *Time Averaged Hit Ratio* (TAHR) qui est une combinaison linéaire des Hit ratio calculés dans les (Z-1) premières tranches de temps. Dans la combinaison linéaire, le poids de chaque HR_k est le nombre d'utilisateurs $|U_k|$ concernés par le processus d'évaluation de la tranche de temps de rang k. On a l'équation 6.7 ci-après.

$$TAHR = \frac{\Sigma_k HR_k \cdot |U_k|}{\Sigma_k |U_k|} \tag{6.7}$$

Exploration des valeurs des paramètres

Pour comparer les graphes BIP, STG, CSTG et TCSTG, nous devons nous assurer d'avoir de bonnes valeurs des paramètres utilisés dans ces graphes. A cet effet, nous avons procédé comme Xiang et al. [142]. On considère le vecteur de l'ensemble des paramètres $[\tau_0, \beta, \eta, \eta_c, \rho, \alpha]$, on fixe une valeur initiale à chaque paramètre de ce vecteur [0, 0.5, 0.5, 0.5, 0.5, 0.5]. Ensuite, on recherche la meilleure valeur de chaque paramètre suivant l'ordre du vecteur des paramètres, et en parcourant uniquement les valeurs définies dans le tableau 6.2. Ainsi, le premier paramètre à optimiser est le paramètre τ_0, qui aura comme valeur 0, puis 1, puis 7 ainsi de suite jusqu'à 365, ceci sans que les valeurs des autres paramètres ne changent. La valeur qui aboutit à la meilleure performance est attribuée à τ_0 et on recommence le processus avec les paramètres β, η, η_c, ρ et α.

La figure 6.2.3 montre les variations de TAHR en fonction des valeurs de tous les paramètres pour le cas du jeu de données de CiteUlike. Sur cette figure, chaque système de recommandation est une combinaison algorithme-graphe. Par exemple, IPF-BIP correspond à l'application de l'algorithme IPF sur le graphe biparti classique BIP.

TABLE 6.2 – Valeurs prédéfinies des paramètres. La colonne $\boldsymbol{v_0}$ représente la valeur initiale de chaque paramètre qui est sur la ligne

	Description du paramètre	$\boldsymbol{v_0}$	**Ensemble des valeurs**
$\boldsymbol{\tau_0}$	Demi-vie (en nombre de jours)	0	0, 1, 7, 15, 30, 45, 60, 90, 180, 365
$\boldsymbol{\beta}$	Préférence à long terme	0.5	$0.1 \times i$ for $i = 0,..,10$
$\boldsymbol{\eta}$	Poids des arcs (i, u)	0.5	$0.1 \times i$ for $i = 0,..,10$ 1.5, 3, 5, 10, 15, 20, 30, 50, 100
$\boldsymbol{\eta_c}$	Poids des arcs (i, c) et (c, i)	0.5	$0.1 \times i$ for $i = 0,..,10$ 1.5, 3, 5, 10, 15, 20, 30, 50, 100
$\boldsymbol{\rho}$	Ajuste la propagation des préférences	0.5	$0.1 \times i$ for $i = 0,..,10$
$\boldsymbol{\alpha}$	Facteur d'amortissement du PageRank	0.5	$0.1 \times i$ for $i = 0,..,10$

Les meilleures valeurs obtenues pour tous les paramètres, pour toutes les combinaisons algorithme-graphe et pour tous les jeux de données, sont présentées dans le tableau 6.3. Par exemple, pour le paramètre τ_0 la meilleure valeur est dans l'intervalle [7, 30] pour la combinaison TPRW-TCSTG dans le jeu de données CiteUlike. Après avoir fixé cette valeur optimale de τ_0, celle de β est dans l'intervalle [0.4, 1], puis celle de η est dans l'intervalle [0.3, 1.5].

Résultats des expérimentations

Les performances des algorithmes TPRW et IPF appliqués aux graphes BIP, STG, CSTG et TCSTG pour la recommandation top-10 dans les trois jeux de données CiteUlike, Delicious et Last.fm sont présentées dans le tableau 6.4. On constate que TPRW-TCSTG que nous proposons, a les meilleurs résultats dans tous les jeux de données et est suivi de TPRW-CSTG. Cette remarque confirme la pertinence de l'insertion des fonctions de décroissance temporelle dans les graphes de recommandation. D'autre part, les performances du STG sont toujours meilleures que celle du graphe biparti classique BIP, ce qui confirme d'avantage l'importance de la prise en compte de la dynamique temporelle dans les graphes de recommandation.

En outre, lorsqu'on considère les meilleures valeurs du paramètre τ_0 de la combinaison TPRW-TCSTG, on constate que dans le cas des jeux de données des marque-pages sociales CiteUlike et Delicious, ces valeurs sont inférieures à un mois. Par contre dans le jeu de données Last.fm, la valeur optimale de τ_0 est supérieure à un mois. Cette observation signifie que l'impact des pages web marquées dans le passé, sur les pages web à recommander diminue rapidement dans le temps comparé à l'impact des anciennes chansons écoutées dans le passé, sur les chansons à recommander. Ceci peut s'expliquer par le fait que les goûts musicaux sont plus stables dans le temps.

Une autre observation que nous faisons est le fait que TPRW est de loin meilleur que l'algorithme IPF. Ainsi, nous recommandons fortement l'usage du PageRank personnalisé pour avoir les meilleures qualités de recommandation. On peut justifier ces performances

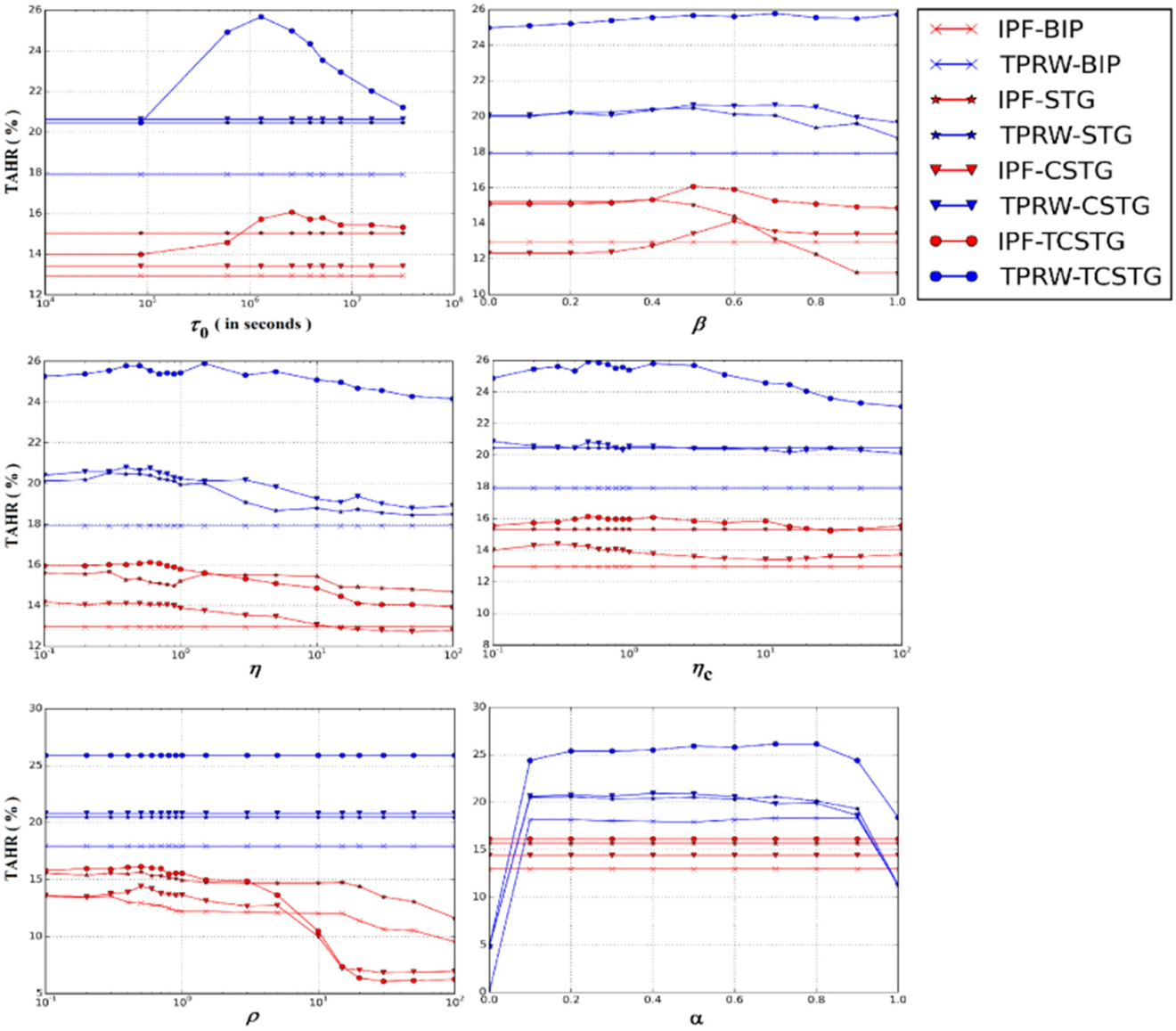


FIGURE 6.3 – Variation de la métrique d'évaluation TAHR en fonction des valeurs des paramètres τ_0, β, η, η_c, ρ et α dans le cas du jeu de données CiteUlike.

par le fait que le processus de propagation de TPRW ne se limite pas uniquement à la proximité au nœud source comme IPF, mais favorise également la recommandation des produits populaires du graphe. En effet, lorsqu'on s'éloigne du nœud source, on a plus de chance d'atteindre les produits populaires grâce à leur degré entrant élevé.

6.3 Link Stream Graph

Les graphes de recommandation de base des sections précédentes à savoir le graphe biparti classique (BIP) et le *Session−based Temporal Graph* (STG), ne tiennent pas compte du temps de manière continue dans leur structure. En effet, BIP ignore la dy-

TABLE 6.3 – Meilleures valeurs des paramètres τ_0, β, η, η_c, ρ et α dans les jeux de données CiteUlike, Delicious et Last.fm.

CiteUlike	$\tau 0$	β	η	η_c	ρ	α
IPF-BIP	-	-	-	-	0.1-0.3	-
IPF-STG	-	0.0-0.5	0.1-0.3	-	0.1-0.7	-
IPF-CSTG	-	0.5-1	0.0-0.9	0.2-0.5	0.5-0.6	-
IPF-TCSTG	15-60	0.5-0.6	0.1-0.9	0.4-1.5	0.1-1	-
TPRW-BIP	-	-	-	-	-	0.1-0.9
TPRW-STG	-	0.0-0.7	0.3-0.6	-	-	0.1-0.7
TPRW-CSTG	-	0.5-0.8	0.3-0.6	0.1-0.7	-	0.1-0.6
TPRW-TCSTG	7-30	0.4-1	0.3-1.5	0.5-3	-	0.5-0.8

Delicious	$\tau 0$	β	η	η_c	ρ	α
IPF-BIP	-	-	-	-	0.1-10	-
IPF-STG	-	0.0-0.4	0-0.1	-	0.1-0.6	-
IPF-CSTG	-	0.5	15-50	0.3-0.9	0.4-1.5	-
IPF-TCSTG	1-7	0.5-0.6	0.5-0.8	50-100	0.5-1.5	-
TPRW-BIP	-	-	-	-	-	0.1-0.9
TPRW-STG	-	0.0-0.4	0.1-0.2	-	-	0.2-0.5
TPRW-CSTG	-	0.0-0.6	15-100	0.2-0.8	-	0.5-0.7
TPRW-TCSTG	7	0-1	0.5-0.8	0-0.1	-	0.1-04

Last.fm	$\tau 0$	β	η	η_c	ρ	α
IPF-BIP	-	-	-	-	0.1-0.8	-
IPF-STG	-	0.5-1	0.9-1.5	-	1.5-10	-
IPF-CSTG	-	0-0.4	0-0.3	30-100	0.4-0.6	-
IPF-TCSTG	1-15	0-0.4	0.1-0.3	1-100	10-50	-
TPRW-BIP	-	-	-	-	-	0.1-0.5
TPRW-STG	-	0.5-0.7	0.1-1.5	-	-	0.2-0.5
TPRW-CSTG	-	0.2-0.4	0.2-0.5	5-30	-	0.4-0.6
TPRW-TCSTG	30-90	0.5-0.7	0.4-0.6	1-5	-	0.4-0.7

TABLE 6.4 – Meilleurs résultats obtenus avec les algorithmes IPF et TPRW appliqués aux graphes BIP, STG, CSTG et TCSTG sur les données de CiteUlike, Delicious et Last.fm.

	CiteUlike	**Delicious**	**Last.fm**
	TAHR (%)	**TAHR (%)**	**TAHR (%)**
IPF-BIP	13.5	7.3	16.3
IPF-STG	15.7	8.6	18.2
IPF-CSTG	14.4	6.4	28.9
IPF-TCSTG	16.1	9.7	26.6
TPRW-BIP	18.3	8.8	27.9
TPRW-STG	20.6	9.2	30.2
TPRW-CSTG	20.9	10.7	37.7
TPRW-TCSTG	**26.1**	**13.2**	**38.9**

namique temporelle et STG tient compte du temps de manière discontinue car divise le temps en tranches pour représenter les sessions des préférences à court terme. Ces considérations du temps font perdre des informations sur la dynamique temporelle des actions des utilisateurs.

Par exemple, si un utilisateur u sélectionne plusieurs fois un produit i, ces multiples sélections de l'utilisateur u sont agrégés en une seule arête (u, i) dans BIP. Néanmoins, dans STG, ceci peut être représenté séparément si les multiples sélections de u se déroulent dans des tranches de temps différentes. Dans le cas contraire, les multiples sélections de i par u sont agrégés en deux arêtes $(u, i), ((u, T_k), i)$. En d'autres termes, si les tranches de temps T_k sont grandes, on perd d'avantage d'informations sur la dynamique des actions des utilisateurs.

Pour dépasser la limite de ces graphes à représenter la dynamique des actions des utilisateurs, tout en considérant le temps de manière continue, nous proposons le *Link Stream Graph* (LSG) qui est inspiré d'un formalisme de représentation des flots de liens décrit dans [80]. Ce graphe permet de conserver les dynamiques temporelle et structurelle des liens utilisateur-produit. Dans la suite de cette section, nous détaillons ce travail qui a été présenté à la première édition du colloque de Mathématiques et d'Informatique (CMI/CMC 2019) de l'Université de Dschang au Cameroun [102].

6.3.1 Construction du graphe

Dans le graphe LSG, chaque utilisateur est représenté par un nombre de nœuds qui est égal au nombre de ses actions sur les produits et chaque produit est représenté par un nombre de noeuds égal au nombre d'actions effectuées sur ce produit. Ainsi, l'ensemble des nœuds de ce graphe est donné par : $\{(t, u) : \exists i, (t, u, i) \in L\} \cup \{(t, i) : \exists u, (t, u, i) \in L\}$, où u est un utilisateur, i un produit et L est le flot de liens de la période d'observation. En d'autres termes, chaque utilisateur u est représenté par un nœud (t, u) s'il a sélectionné un produit à l'instant t tel que $(t, u, i) \in L$. De même, chaque produit i est représenté par un nœud (t, i) si ce produit a été sélectionné à l'instant t tel que $(t, u, i) \in L$.

Nous définissons les arcs du LSG de telles sortes que : tous les nœuds du même utilisateur (resp. produit), sont liés par un chemin dans l'ordre chronologique ; et pour chaque lien $(t, u, i) \in L$, le nœud (t, u) de l'utilisateur et le nœud (t, i) du produit sont reliés par un arc bidirectionnel. L'ensemble E des arcs est donc défini par $\{((t, u), (t, i)) : (t, u, i) \in L\} \cup \{((t, u), (t', u)) : \exists i, (t, u, i) \in L, t' = \min\{x : x > t \text{ and } \exists i', (x, u, i') \in L\} \cup \{((t, i), (t', i)) : \exists u, (t, u, i) \in L, t' = \min\{x : x > t \text{ and } \exists u', (x, u', i) \in L\}$.

En ce qui concerne la pondération des arcs, nous attribuons un poids η aux arcs qui mènent à un nœud plus ancien et un poids de 1 est attribué à tous les autres arcs. De manière formelle, pour tout $((t, x), (t', y)) \in E$, $w((t, x), (t', y)) = \eta$ si $x = y$ et $t > t'$; et $w((t, x), (t', y)) = 1$ pour tous les autres, avec w la fonction de pondération des arcs. La figure 6.4 présente un exemple de graphe LSG.

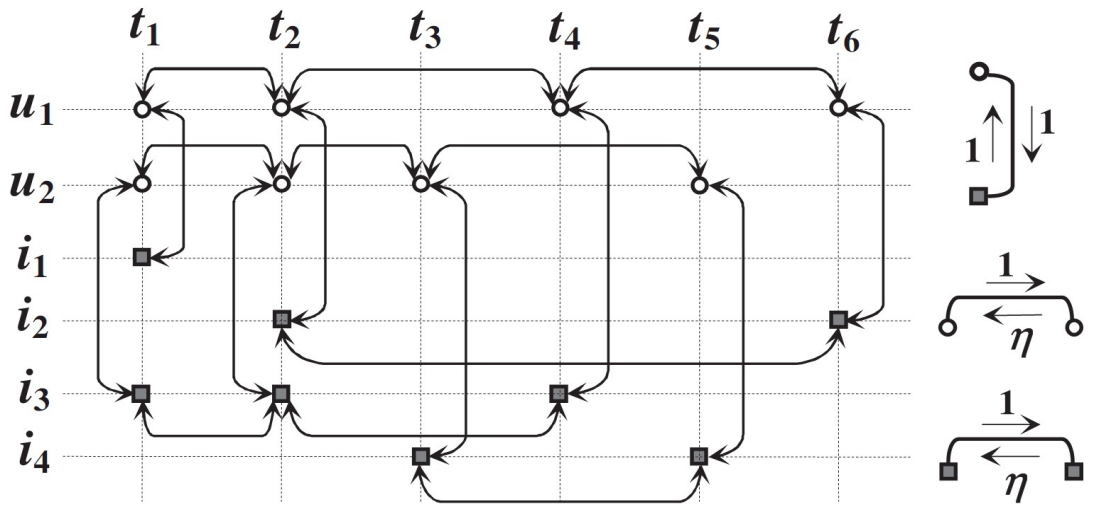


FIGURE 6.4 – Graphe LSG (*Link Stream Graph*) construit à partir du flot de liens $L = \{(t_1, u_1, i_1), (t_1, u_2, i_3), (t_2, u_1, i_2), (t_2, u_2, i_3), (t_3, u_2, i_4), (t_4, u_1, i_3), (t_5, u_2, i_4), (t_6, u_1, i_2)\}$.

6.3.2 Calcul des recommandations

Dans cette section, nous réutilisons l'algorithme *Temporal Personalized Random Walk* (TPRW) présenté dans la section 6.2.2 précédente. Cependant, la configuration du vecteur d de personnalisation du PageRank est légèrement différente. En effet, pour recommander des produits à un utilisateur cible u, la source de la propagation du PageRank est le nœud (t, u) le plus récent de cet utilisateur. La configuration du vecteur d est effectuée de telle sorte que le marcheur aléatoire redémarre uniquement sur le nœud (t, u) le plus récent. On a donc la configuration suivante :

$$d_{LSG}(x) = \begin{cases} 1 & si\ x = (t, u)\ \ est\ la\ plus\ récente\ occurrence\ de\ u \\ 0 & sinon \end{cases} \tag{6.8}$$

Une fois le processus de propagation achevé, la préférence de u pour un produit i est donnée par la somme des préférences reçues par les nœuds (t, i) de ce produit. Ceci est différent de ce qui est fait dans les graphs BIP et STG dans lesquels chaque produit est représenté par un unique nœud.

6.3.3 Expérimentations et résultats

Nous avons mené plusieurs expérimentations pour évaluer les performances de LSG comparé aux graphes BIP et STG. Dans la suite, nous décrivons d'abord les données utilisées, ensuite nous présentons le protocole expérimental utilisé et les résultats obtenus.

Description des données

En plus des données de CiteUlike, Delicious et Last.fm, nous utilisons également des jeux de données de Ponpare [1], un site de vente en ligne des coupons d'achat, et de deux sites de commentaires des produits de catégories variées (audiovisuel, santé, électroménager) Epinions et Ciao [2] [130]. Les données sont modélisées sous forme de flot de liens (t, u, i) et l'interprétation de chaque triplet dépend du domaine. Dans le site de vente en ligne, un triplet signifie que l'utilisateur u a acheté le produit i à l'instant t, dans les sites de marque-pages CiteUlike et Delicious, ceci veut dire que u a marqué la page i à l'instant t et pour le cas de Last.fm, u a écouté la chanson i à l'instant t.

En ce qui concerne Epinions et Ciao, les données disponibles sont des notes que les utilisateurs attribuent aux produits. On a donc le quadruplet (t, u, i, r), où $r \in 1, 2, 3, 4, 5$ est la note que l'utilisateur u attribue au produit i. Cependant, nous travaillons uniquement avec des données implicites qui traduisent généralement une relation positive entre l'utilisateur et le produit. Pour ce faire, nous filtrons ces données de manière à extraire uniquement les triplets (t, u, i) tel que l'utilisateur u a attribué une note supérieure à 2.5 et à la moyenne de ses notes.

Tous les jeux de données sont filtrés de sorte que les utilisateurs et les produits considérés sont ceux qui apparaissent dans au moins σ triplets. Le tableau 6.5 fournit des détails sur les données utilisées : date de début, date de fin, seuil σ, nombre d'utilisateurs, nombre de produits, nombre de couple utilisateur-produit distincts et enfin le nombre de liens du flot de liens.

TABLE 6.5 – Statistiques sur les données

	CiteUlike	**Delicious**	**Last.fm**	**Ponpare**	**Epinions**	**Ciao**
D. début	2010-01-01	2010-05-10	2005-02-14	2011-07-01	2010-01-01	2007-01-01
D. fin	2010-09-01	2010-11-09	2005-06-14	2011-11-01	2010-12-31	2010-12-31
$\boldsymbol{\sigma}$	10	4	6	10	1	1
|U|	1 726	1 175	104	1 322	1 843	879
|I|	619	1 352	1 035	1 115	15 899	6 005
Nb. (u-i)	9 360	7 652	8 113	12 863	17 722	8 109
|L|	24 772	35 558	25 567	177 005	17 722	8 109

Protocole expérimental et évaluation des recommandations

Pour les expérimentations, nous utilisons le même protocole que celui défini dans la section 6.2.3. La première différence ici est que la taille d'une tranche de temps varie d'un jeu de données à l'autre, car nous avons fixé le nombre de tranche de temps à $Z = 8$. La seconde différence vient du fait que nous utilisons plusieurs métriques, à savoir F1-score,

1. https ://www.kaggle.com/c/coupon-purchase-prediction
2. `https://www.cse.msu.edu/~tangjili/trust.html`

Hit ratio et *Mean Average Precision* (MAP), pour évaluer les performances des graphes BIP, STG et LSG.

Lorsqu'on se sert de plusieurs métriques d'évaluation, les comparaisons sont plus crédibles car un système peut avoir de bons résultats pour une métrique et être moins intéressant pour une autre comme l'explique Gunawardana et Shani dans [50]. Dans cette logique, l'utilisation de la métrique F1-score permet d'évaluer la qualité de prédiction, la métrique Hit ratio permet d'évaluer le taux d'utilisateurs satisfaits et la métrique MAP permet d'évaluer la facilité d'accès aux bonnes recommandations.

Pour chaque fenêtre de temps W_k, pour $k = 1, .., Z-1$, avec Z le nombre de tranches, le protocole d'évaluation utilise les mêmes étapes que dans la section 6.2.3 :

- Construire les graphes correspondant aux données de W_1, W_2, .., W_k
- Calculez les recommandations Top-N pour les utilisateurs ayant sélectionné au moins un "nouveau produit" dans la tranche de temps suivante W_{k+1}.
- Évaluez les graphes de recommandation en calculant les métriques d'évaluation $F1_k$, HR_k et MAP_k de la tranche de temps de rang k, en utilisant la fenêtre W_{k+1} comme jeu de test. En plus des valeurs de ces métriques, nous conservons également leur dénominateur respectif $deno_{F1_k}$, $deno_{HR_k}$ et $deno_{MAP_k}$.

Soit $M \in F1, HR, MAP$, après avoir déterminé toutes les valeurs M_k de la métrique d'évaluation M pour toutes les tranches de temps, nous calculons la moyenne temporelle de toutes ces valeurs nommée $Time\ Averaged(M)$. C'est une combinaison linéaire des valeurs M_k calculées dans les (Z-1) premières tranches de temps. Dans la combinaison linéaire, le poids de chaque M_k est la valeur $deno_{M_k}$ du dénominateur de cette métrique dans la tranche de concernée. On a l'équation 6.9 ci-après.

$$TA(M) = \frac{\Sigma_k M_k \cdot deno_{M_k}}{\Sigma_k deno_{M_k}} \tag{6.9}$$

Performances optimales des graphes

Pour comparer les graphes BIP, STG et LSG, nous devons nous rassurer d'avoir les meilleures performances de ces graphes et pour chacune des métriques F1-score, Hit Ratio et MAP. L'espace des valeurs de chaque paramètre étant infini, nous avons fixé un ensemble de valeurs qui peuvent être attribuées à chacun des paramètres. Le tableau 6.6 présente la liste de ces valeurs.

D'autre part, nous souhaitons optimiser les performances des graphes pour plusieurs métriques d'évaluation. A cet effet, nous avons utilisé la technique d'optimisation *Randomized Search Cross−Validation* [18]. Cette méthode construit aléatoirement k paramétrages pour chaque système de recommandation, avec k fixé au départ. Les valeurs des paramètres de chaque paramétrage sont choisies aléatoirement dans les ensembles prédéfinis du tableau 6.6. Dans ce travail, nous avons fixé $k = 50$.

TABLE 6.6 – Valeurs prédéfinies des paramètres

	Description du paramètre	**Valeurs prédéfinies**
$\boldsymbol{\Delta}$	Durée d'une session court terme du STG	7, 30, 60, 90, 180, 365, 540, 730 jours
$\boldsymbol{\beta}$	Préférence à long terme dans STG	0.1, 0.3, 0.5, 0.7, 0.9
$\boldsymbol{\eta}$	Poids de connexion au passé	0, 0.1, 0.2, 0.5, 1, 2, 5, 10
$\boldsymbol{\alpha}$	Facteur d'amortissement du PageRank	0.05, 0.1, 0.15, 0.3, 0.5, 0.7, 0.9

Résultats des expérimentations

Les performances du PageRank personnalisé appliqué aux graphes BIP, STG et LSG pour la recommandation top-10 dans les six jeux de données sont présentées dans le tableau 6.7. On constate que le graphe LSG est le meilleur 12 fois parmi les 18 cas possibles. Ceci montre que l'usage du graphe LSG dans les systèmes de recommandation est pertinent. Cependant, LSG n'est pas toujours meilleur que BIP et STG. On déduit donc qu'il y a des contextes précis dans lesquels on peut recommander le graphe LSG et d'autres dans lesquels il vaut mieux utiliser les autres graphes.

TABLE 6.7 – Meilleures performances des graphes de recommandation

	F1-score (%)			**Hit ratio (%)**			**MAP (%)**		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
CiteUlike	**7.65**	7.39	7.48	28.7	**29.5**	28.5	10.7	**10.9**	**10.9**
Delicious	5.08	5.13	**6.56**	10.7	11.7	**13.7**	5.15	5.49	**5.96**
Last.fm	3.93	**4.88**	3.77	26.2	**28.5**	24.9	9.52	**11.8**	8.50
Ponpare	5.52	6.32	**6.67**	11.8	12.4	**29.5**	4.03	4.29	**11.7**
Epinions	1.21	1.23	**1.55**	5.14	5.68	**6.89**	2.02	2.05	**2.59**
Ciao	1.78	1.73	**3.0**	5.63	**6.35**	**6.35**	2.18	**2.23**	2.11

Nous pensons que LSG est adapté aux contextes dans lesquels la connaissance des vieilles préférences d'un utilisateur n'aide pas significativement dans la prédiction des préférences actuelles de ce dernier ou encore dans des contextes où les produits deviennent rapidement obsolètes ou démodés. Cette hypothèse peut justifier le fait que LSG ait de très mauvais résultats dans le jeu de données Last.fm, où les goûts musicaux des utilisateurs sont stables dans le temps, et de très bons résultats dans Ponpare où les coupons vendus ont des dates d'expiration.

En ce qui concerne les valeurs des paramètres, nous nous sommes particulièrement attardé sur le paramètre η du graphe LSG qui permet de pondérer les arcs qui mènent à des nœuds plus vieux. Les meilleures valeurs de η dans chaque jeu de données et selon chacune des métriques d'évaluation sont présentées dans le tableau 6.8. On constate que $\eta = 0$ dans le jeu de données Ponpare et que sa valeur est inférieurs à 1 dans les autres cas à l'exception des cas du jeu de données Last.fm et un cas du jeu de données Ciao. Cette dernière remarque montre une fois de plus que les vieilles données devraient avoir

moins de poids dans le processus de prédiction des préférences actuelles des utilisateurs.

TABLE 6.8 – Meilleures valeurs du paramètre η du Link Stream Graph dans chacun des jeux de données et suivant les métriques d'évaluation top-10

	CiteUlike	Delicious	Last.fm	Ponpare	Epinions	Ciao
F1@10	0.2	0.1	10	0	0	0
HR@10	0.5	0.5	1	0	0.2	0.2
MAP@10	0.2	0.5	1	0	0	1

6.4 Résumé

Dans ce chapitre, nous avons présenté les principaux graphes de recommandation de la littérature en mettant l'accent sur leur gestion de la dynamique temporelle des actions des utilisateurs et sur la prise en compte des informations sur le contenu et sur la confiance. Pour intégrer les informations secondaires, plusieurs de ces graphes étendent le graphe biparti classique en y insérant de nouveaux types de nœuds. On a par exemple, un type de nœuds pour les attributs descriptifs des produits en ce qui concerne les informations sur le contenu, et un type de nœud pour représenter le comportement à court terme des utilisateurs en ce qui concerne la dynamique temporelle.

Cependant, tous les précédents graphes de recommandation accordent le même poids aux vieilles arêtes et aux arêtes les plus récentes alors que les actions les plus récentes d'un utilisateur reflètent mieux ses préférences actuelles. Pour pallier à cette limite, nous avons proposé d'intégrer des fonctions de décroissance temporelles dans les graphes de recommandation. Ceci a permis d'améliorer la qualité des recommandations des graphes de recommandation.

D'autre part, les graphes de recommandation de la littérature ne considèrent pas le temps de manière continue dans leur structure ; la dimension temporelle est soit ignorée, soit divisée en tranches. Nous avons dépassé cette limite en proposant le *Link Stream Graph* dont la pertinence à été montrée suite à une série d'expérimentations sur 6 jeux de données et avec 3 métriques d'évaluation.

Tous les systèmes de recommandation de la littérature qui sont basés sur les graphes, à l'exception d'un seul, ne considèrent pas les informations sur la confiance. De plus, aucun de ces systèmes ne combine simultanément la confiance à une autre information secondaire. Par conséquent, aucun de ces systèmes ne tire avantage simultanément des informations sur le contenu, la confiance et temps. Nous abordons cette question dans le chapitre suivant.

Chapitre 7

GraFC2T2 : cadre général pour graphes de recommandation enrichis

Sommaire

La plupart des systèmes de recommandation basés sur les graphes ignorent les informations sur la confiance, et lorsque ces systèmes considèrent des informations secondaires ce sont soit des informations sur le contenu, soit la dimension temporelle et rarement les deux simultanément. Ces systèmes sont limités car l'intérêt qu'un utilisateur a pour un produit peut être dû aux caractéristiques de ce produit, ou peut changer avec le temps et peut également être fortement lié aux opinions de ceux à qui il fait confiance. Ainsi, ignorer une de ces informations peut être un frein pour l'obtention de meilleures performances.

Dans ce chapitre, nous proposons un cadre général pour les systèmes de recommandation basés sur les graphes qui permet de prendre en compte simultanément les attributs de description des produits, la dynamique temporelle des actions des utilisateurs et les données sur les relations de confiance entre les utilisateurs. Nous appliquons les systèmes

modélisables dans notre framework aux recommandations top-N et évaluons l'impact de la prise en compte des combinaisons variées des trois types d'informations secondaires considérées.

Plan du chapitre. Dans la section 7.1 nous présentons les différents modules du framework, à savoir les graphes de base, les méthodes d'extension par des informations sur le contenu et des fonctions de décroissance temporelle, et enfin le mécanisme de calcul des recommandations top-N qui tient compte des relations de confiance. La section 7.2 est dédiée à la mise en œuvre des systèmes de recommandation du framework proposé et à la présentation des résultats. Nous clôturons ce chapitre dans la section 7.3, en présentant l'impact de l'insertion des informations secondaires dans les graphes de recommandation et nous comparons les résultats obtenus à ceux de quelques systèmes de recommandation conçus pour des recommandations top-N à partir de données implicites.

7.1 Description de GraFC2T2

Dans cette section, nous considérons que les données contiennent un ensemble U d'utilisateurs, un ensemble I de produits, un ensemble C d'attributs de description des produits et que les actions des utilisateurs sur les produits sont observées durant un intervalle de temps T. De plus, nous avons une fonction qui exprime les relations de confiance entre les utilisateurs de telle sorte que $trust(u, v) \in [0, 1]$ représente le niveau de confiance que l'utilisateur u accorde à l'utilisateur v.

Nous modélisons les actions des utilisateurs sur les produits par un flot de liens L inclus dans $T \times U \times I$ où chaque lien $(t, u, i) \in L$ peut représenter un achat (u a acheté le produit i à l'instant t), un intérêt pour un produit audiovisuel (comme regarder un film ou écouter une chanson), ou une autre action de sélection d'un produit par un utilisateur suivant le contexte d'application.

Dans la suite, nous présentons d'abord les graphes de base considérés dans le framework, puis nous décrivons des méthodes d'extension de ces graphes par des informations secondaires liées aux caractéristiques de description des produits et à la dynamique temporelle. Nous clôturons cette section en présentant un processus de calcul des recommandations top-N qui tient compte des informations sur la confiance.

Ce travail est publié au numéro spécial "Analysis of networks and graphs" du *Journal of Interdisciplinary Methodologies and Issues in Science* (JIMIS 2019) du groupe *Episcience* [1] [101]. GraFC2T2 est accessible via le lien `https://github.com/nzekonarmel/GraFC2T2` afin d'aider d'autres chercheurs et praticiens à mener des expérimentations sur leurs propres jeux de données et à tester la pertinence de nouvelles informations secondaires.

1. https ://jimis.episciences.org/

7.1.1 Graphes de base

Pour illustrer les graphes de base dans la suite, nous considérons l'ensemble des utilisateurs $U = \{u_1, u_2\}$, l'ensemble des produits $I = \{i_1, i_2, i_3, i_4\}$, l'intervalle de temps d'observation $T = [t_1, t_6]$ et le flot de liens $L = \{(t_1, u_1, i_1), (t_1, u_2, i_3), (t_2, u_1, i_2), (t_2, u_2, i_3), (t_3, u_2, i_4), (t_4, u_1, i_3), (t_5, u_2, i_4), (t_6, u_1, i_2)\}$. La figure 7.1 montre une représentation du flot de liens de l'exemple en utilisant un formalisme décrit dans [136, 80].

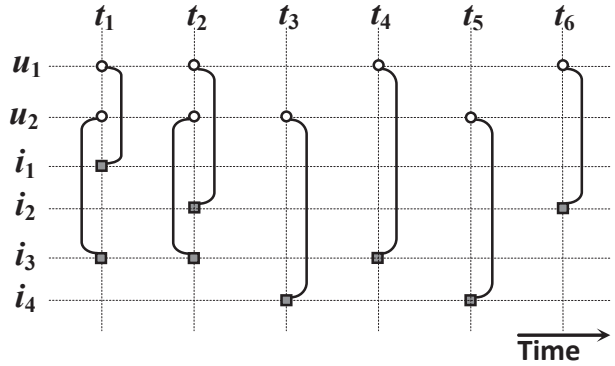


FIGURE 7.1 – Représentation du flot de liens exemple utilisé pour illustrer les graphes de base considérés dans le framework proposé.

Graphe biparti classique

Le premier graphe de base considéré est le graphe biparti classique (BIP) [14], c'est un graphe biparti (U, I, E) où U et I sont respectivement l'ensemble des utilisateurs et des produits définis ci-dessus. $E \subseteq U \times I$ est l'ensemble des arcs définis par $E = \{(u, i), (i, u) : \exists t \in T, (t, u, i) \in L\}$. En d'autres termes, u est lié à i dans BIP si l'utilisateur u a sélectionné le produit i pendant la période d'observation T. La figure 7.2(a) illustre le graphe BIP du flot de liens exemple.

Session-based Temporal Graph

Le graphe biparti classique ignore la dimension temporelle. Dans une première tentative de capturer des informations temporelles, nous considérons le *Session−based Temporal Graph* (STG) [142]. Ce graphe code les informations temporelles en utilisant un ensemble S de nœuds sessions définis en découpant le temps en tranches T_k. Chaque utilisateur a un nœud session $(u, T_k) \in S$ pour chaque intervalle de temps T_k au cours duquel cet utilisateur est actif. Le STG est décrit avec plus détails dans la section 6.1.2 du chapitre 6.

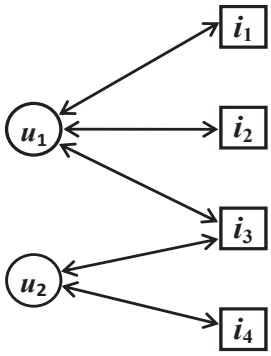
Par souci de simplicité, les valeurs des paramètres de pondération des arcs du STG sont fixées à 1. Néanmoins ces paramètres peuvent être ajoutés si nécessaire par des

utilisateurs de GraFC2T2. La figure 7.2(b) illustre la représentation du STG résultat pour le flot de liens exemple.

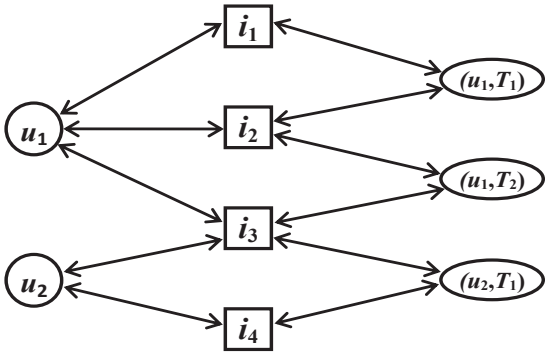
Link Stream Graph

Afin de capturer des informations temporelles tout en évitant les inconvénients du choix d'une taille de fenêtre de temps comme dans le STG, nous considérons également le *Link Stream Graph* (LSG) que nous proposons dans [102]. Dans ce graphe, chaque utilisateur u est représenté par les nœuds (t, u) lorsqu'un lien l'implique à l'instant t, et chaque produit est représenté de la même manière. Pour plus de détails, voir la section 6.3 du chapitre 6.

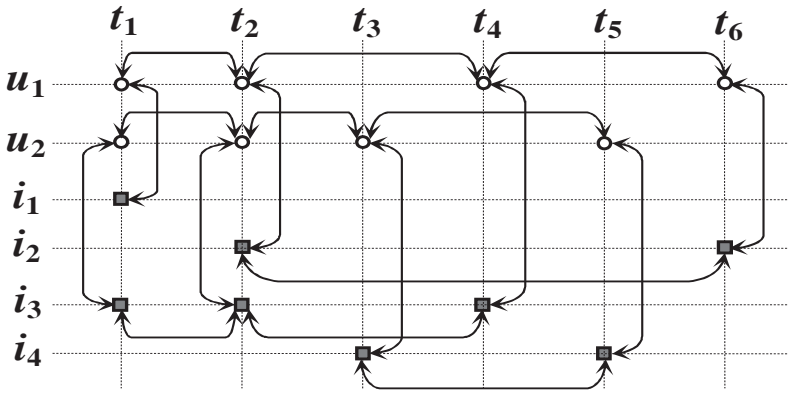
Une fois de plus, les valeurs des paramètres de pondération des arcs sont fixées à 1 pour des raisons de simplicité. La figure 7.2(c) illustre le LSG résultat du flot de liens exemple.



(a) Graphe biparti classique (BIP)



(b) Graphe STG (*Session–based Temporal Graph*)



(c) Graphe LSG (*Link Stream Graph*)

FIGURE 7.2 – Graphe biparti classique, graphe STG et graphe LSG construits à partir du flot de liens exemple $L = \{(t_1, u_1, i_1), (t_1, u_2, i_3), (t_2, u_1, i_2), (t_2, u_2, i_3), (t_3, u_2, i_4), (t_4, u_1, i_3), (t_5, u_2, i_4), (t_6, u_1, i_2)\}$.

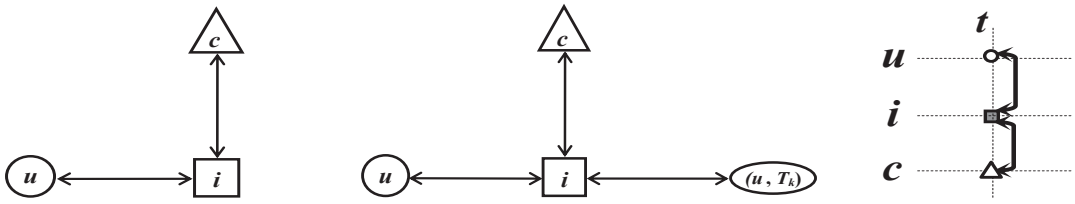
7.1.2 Extension des graphes de base

Une fois qu'un graphe de recommandation de base est construit comme décrit dans la section précédente, on peut y intégrer des informations secondaires comme les attributs descriptifs des produits et des fonctions de décroissance temporelle pour affiner la prise en compte de la dynamique temporelle. A ce niveau, l'utilisateur de notre framework a plusieurs choix que nous présentons dans la suite.

Intégration des informations sur le contenu

Ici on considère C l'ensemble des attributs de description des produits et la fonction $g(i) \subseteq C$ qui à chaque produit i associe la liste des attributs descriptifs de ce produit. Un attribut peut être le genre d'un film, l'auteur d'une chanson ou une étiquette associée à un produit. En suivant la méthode proposée dans [114, 148, 100], un nouveau type de nœuds (*content node* ou nœud contenu) est intégré dans chacun des graphes de base. Chaque nœud de ce type représente un attribut descriptif des produits et est lié à tous les nœuds des produits qui ont cet attribut en commun. Nous utilisons deux méthodes pour l'insertion des informations sur le contenu : la méthode CI (*content-item*) et la méthode CIU (*content-item-user*).

Méthode CI. Dans les graphes BIP et STG, nous ajoutons un nœud contenu $c \in C$ pour chaque attribut basée sur le contenu, et nous lions chaque nœud produit i aux nœuds contenus $c \in g(i)$. Pour le cas du graphe LSG, nous ajoutons un nœud contenu (t, c) qui a un lien bidirectionnel avec le nœud produit (t, i) du graphe de base, lorsque $c \in g(i)$. Nous appelons cette méthode *content-item* (CI) car on ajoute uniquement des liens entre les nœuds produit et les nœuds contenu. Voir la figure 7.3 pour une illustration.

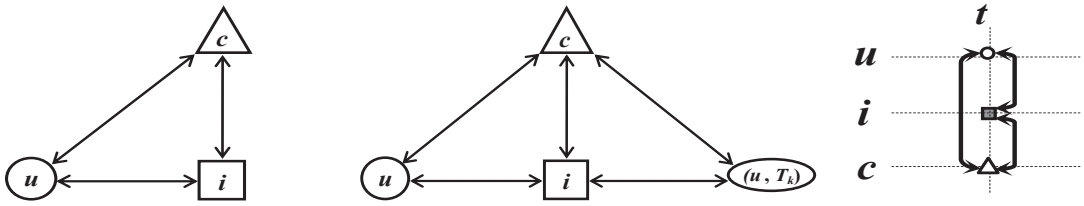


(a) BIP + CI (b) STG + CI (c) LSG + CI

FIGURE 7.3 – Ajout des nœuds des attributs de description des produits en utilisant la méthode CI dans chacun des graphes de base.

Méthode CIU. La seconde stratégie proposée permet de relier les nœuds contenus aux nœuds produits mais également aux nœuds utilisateurs, raison pour laquelle nous l'appelons $content - item - user$ (CIU). L'idée est de relier le nœud d'un utilisateur aux

nœuds des attributs qui décrivent les produits qui l'intéressent. Par conséquent, outre les ajouts de la méthode CI, la méthode CIU ajoute à BIP des liens bidirectionnels (u, c) entre chaque nœud utilisateur u et un nœud de contenu c chaque fois qu'il existe un nœud produit lié à la fois à u et à c. Dans le STG un lien bidirectionnel est ajouté entre chaque nœud session (u, T_k) et le nœud contenu c chaque fois qu'un nœud produit est lié à la fois à (u, T_k) et c. En ce qui concerne le graphe LSG un lien bidirectionnel est créé entre chaque nœud utilisateur (t, u) et le nœud contenu (t, c) chaque fois qu'un nœud produit est lié aux deux. Voir la figure 7.4 pour une illustration.



(a) BIP + CIU (b) STG + CIU (c) LSG + CIU

FIGURE 7.4 – Ajout des nœuds des attributs de description des produits en utilisant la méthode CIU dans chacun des graphes de base.

Comparé à CI, la méthode CIU augmente l'influence des attributs basés sur le contenu. En effet, la méthode CIU permet une meilleure promotion des produits qui ont les mêmes caractéristiques que ceux que l'utilisateur cible a sélectionnés dans le passé.

Intégration des fonctions de décroissance temporelle

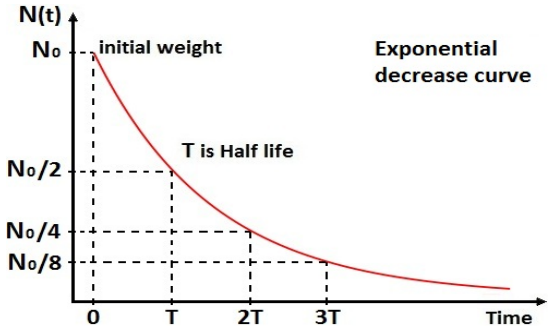
Dans les sections précédentes, la dynamique temporelle est modélisée directement dans les graphes STG et LSG, mais leurs pondérations des arcs donnent une vue statique des préférences des utilisateurs. Comme ces préférences évoluent avec le temps, comme indiqué par Ding et Li [41], il est important de différencier les poids des vieux arcs de ceux des arcs les plus récents. A cet effet, nous utilisons des fonctions de décroissance temporelle.

L'idée est de donner un poids élevé aux arcs récents et de diminuer le poids des arcs en fonction de leur âge. Supposons que t_e est l'instant d'apparition le plus récent de l'arc (a, b), et qu'on doit effectuer une recommandation à l'instant t, le poids de l'arc (a, b) au moment de la recommandation est donné par : $w_t(a, b) = f(t - t_e) \cdot w(a, b)$ où $f()$ est une fonction décroissante. De nombreuses fonctions de décroissance temporelle peuvent être utilisées, et nous avons conçu GraFC2T2 pour faciliter l'intégration de ces fonctions. Dans ce travail, nous considérons deux fonctions.

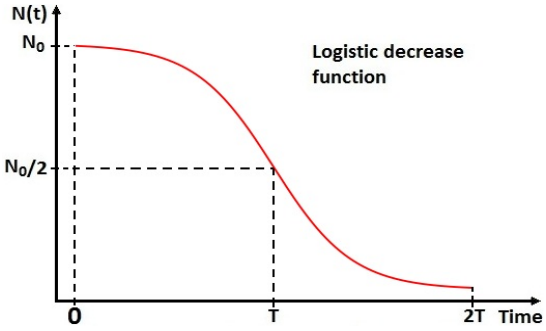
– La première est une fonction de décroissance exponentielle illustrée sur la figure 7.5(a) : $f(x) = e^{-x \cdot \ln(2)/\tau_0}$, où τ_0 est la demi-vie de la radioactivité ; après un délai

de τ_0, le poids de l'arc diminue de moitié.

– La seconde est la fonction de décroissance logistique illustrée dans la figure 7.5(b) : $f(x) = 1 - 1/(e^{-K(x-\tau_0)} + 1)$ où K est la pente de la courbe et τ_0 est le milieu du sigmoïde ; si $x = \tau_0$ alors $f(x) = 0.5$.



(a) Fonction de décroissance exponentielle - *Exponential decay function* (EDF)



(b) Fonction de décroissance logistique - *Logistic decay function* (LDF)

FIGURE 7.5 – Fonctions de pénalisation temporelle des poids des arcs.

7.1.3 Calcul des recommandations top-N

Après avoir construit un graphe de recommandation en utilisant un graphe de base et en y insérant des informations secondaires liées au contenu et au temps, il faut procéder au calcul des recommandations top-N à partir de ce graphe. Pour ce faire, nous utilisons l'algorithme du PageRank personnalisé comme dans le chapitre 6. Ci-dessous, nous faisons un rappel du PageRank personnalisé et présentons notre extension du processus de personnalisation du PageRank par le biais de la confiance entre utilisateurs.

PageRank personnalisé

Nous utilisons l'algorithme *Temporal Personalized Random Walk* (TPRW) proposé par Xiang et al. [142] pour le calcul des recommandations temporelles. Cet algorithme est défini en utilisant l'idée de personnalisation proposée dans [53], et qui correspond à l'équation suivante :

$$PR = \alpha \cdot M \cdot PR + (1 - \alpha) \cdot d \tag{7.1}$$

où PR est le vecteur de PageRank ; M est la matrice de transition du graphe ; α est le facteur d'amortissement de la personnalisation ; et d est le vecteur de personnalisation.

Pour recommander des produits à un utilisateur u à l'instant t, le vecteur de personnalisation d est configuré en fonction du graphe de base comme présenté dans la section 6.2.2 du chapitre 6 pour les graphes BIP et STG, et la section 6.3.2 pour le LSG.

Une fois que le vecteur d est configuré, le PageRank est exécuté sur le graphe de recommandation construit afin de calculer des recommandations top-N à proposer à u à l'instant t. A la fin de l'exécution du PageRank, l'intérêt que u accorde à un produit i est estimé par la valeur du PageRank du nœud i dans les graphes BIP et STG et par la somme des valeurs du PageRank des nœuds produits (t', i) associés à i. Ainsi, le système recommande à u les N produits pour lesquels l'intérêt estimé de u est le plus grand.

Personnalisation du PageRank avec des informations sur la confiance

Les informations sur la confiance sont intéressantes pour améliorer les recommandations, en particulier pour le démarrage à froid des utilisateurs (utilisateurs pour lesquels l'historique des actions passées est très limité). Comme mentionné dans le chapitre 4, certains systèmes de recommandation incorporent des informations explicites sur la confiance entre les utilisateurs [68, 52, 108]. Cependant, ces informations explicites étant rarement disponibles, plusieurs systèmes de recommandation utilisent la confiance implicite [110, 81]. Dans cette section, nous décrivons comment inclure ces deux types de confiance dans notre cadre général des systèmes de recommandation basés sur les graphes.

Nous considérons uniquement la confiance locale, et donc plusieurs utilisateurs peuvent avoir confiance différemment à un autre utilisateur donné. Pour un utilisateur u, nous considérons que TR_u est l'ensemble des utilisateurs à qui u fait confiance et que $trust(u, v) \in [0, 1]$ donne le niveau de confiance que u accorde à v, pout tout $v \in TR_u$.

Lorsque nous considérons les informations explicites, TR_u correspond à l'ensemble des utilisateurs pour lesquels u a exprimé explicitement dans le système qu'il leur fait confiance et pour tous ces utilisateurs v, $trust(u, v) = 1$. Nous avons désigné cette méthode ET - *Explicit Trust* ou confiance explicite.

Il est également possible d'utiliser la confiance implicite déduite des mesures de similarité comme dans [110]. Dans ce cas, TR_u correspond à tous les utilisateurs du système et $trust(u, v) = |I_u \cap I_v|/|I_u \cup I_v|$ qui est la mesure de similarité de Jaccard ; d'autres mesures de similarité peuvent être utilisées, comme la mesure du cosinus. Nous avons désigné cette méthode par IT -*Implicit Trust* ou confiance implicite.

Une fois que nous avons les informations sur la confiance, le vecteur d de personnalisation du PageRank peut être configuré comme défini ci-après.

- pour BIP, $d(u) = 1-\gamma$, $d(v) = (\gamma \cdot trust(u, v))/|TR_u|$ si $v \in TR_u$ et $d(v) = 0$ sinon ;
- pour STG, la préférence à long terme β reçue par le nœud utilisateur u est partagée entre u et les autres utilisateurs v à qui il fait confiance. On a $d(u) = \beta \cdot (1 - \gamma)$, $d(v) = (\beta \cdot \gamma \cdot trust(u, v))/|TR_u|$ pour tout $v \in TR_u$;
  et la préférence à court terme $1 - \beta$ est partagée entre le nœud session (u, T_k) le plus récent de u et les nœuds session les plus récents des utilisateurs à qui il fait confiance. On a $d(u, T_k) = (1 - \beta) \cdot (1 - \gamma)$, $d(v, T_v) = (1 - \beta) \cdot \gamma \cdot trust(u, v)/|TR_u|$ où $v \in TR_u$ et (v, T_k) est le nœud session le plus récent de v. Toutes les autres

entrées de d sont fixées à 0.

– pour LSG, $d(t_k, u) = 1 - \gamma$, $d(t_k, v) = \gamma \cdot trust(u, v)/|TR_u|$ si $v \in TR_u$ et (t_k, v) est le nœud temporel le plus récent de v, et toutes les autres entrées de d sont fixées à 0.

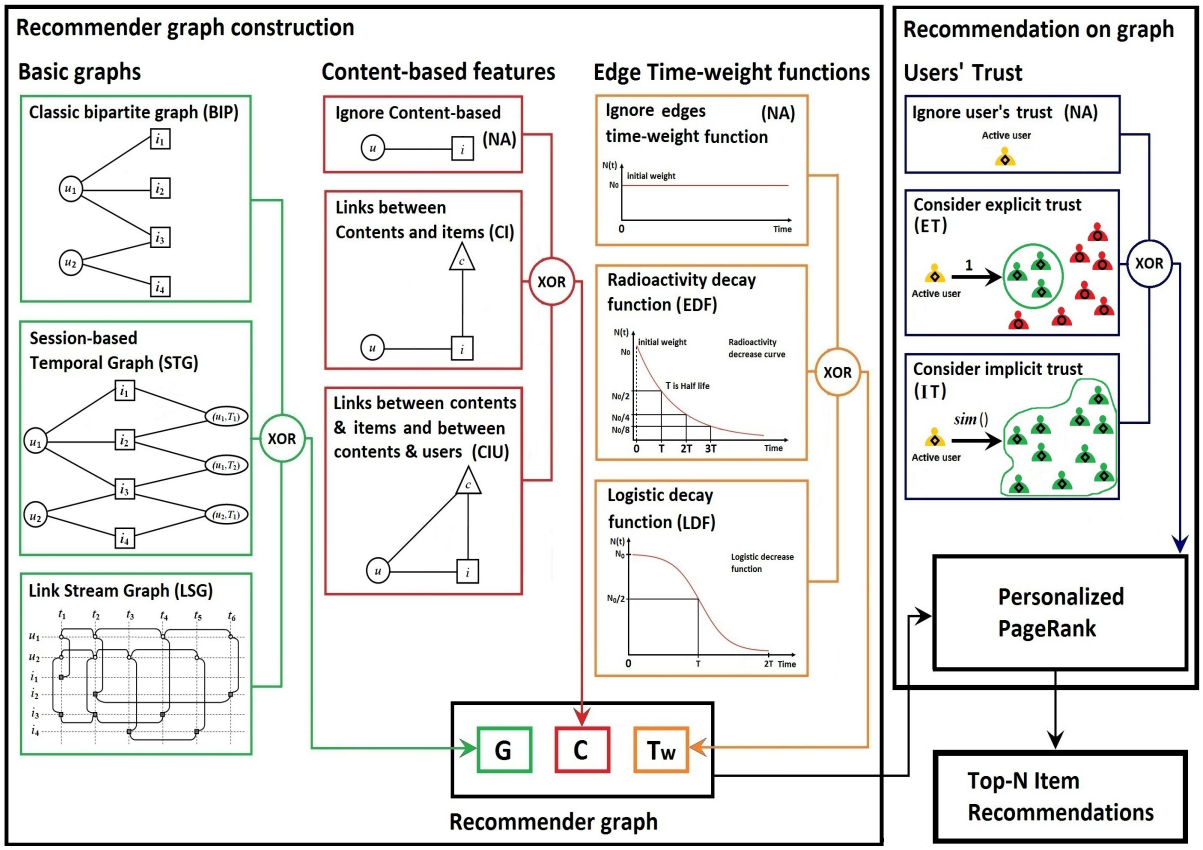


FIGURE 7.6 – **Architecture globale de GraFC2T2**, notre cadre de conception des systèmes de recommandation basé sur des graphes. Les graphes de recommandation sont construits à partir de trois composants : un graphe de base modélisant les relations utilisateur-produit, des attributs descriptifs des produits et une fonction de pénalisation temporelle des arcs les plus vieux, voir les sections 7.1.1 et 7.1.2. Ensuite, des recommandations top-N sont calculées à partir du graphe construit et en utilisant le PageRank personnalisé et les informations basées sur la confiance, voir la section 7.1.3.

La figure 7.6 résume l'architecture globale de GraFC2T2, composée de deux grandes parties : la construction du graphe de recommandation et l'utilisation de ce graphe pour calculer les recommandations top-N. Le graphe de recommandation intègre les informations disponibles en combinant un graphe de base, que nous détaillons dans la section 7.1.1, avec des méthodes permettant d'intégrer les caractéristiques descriptives des produits et des fonctions de décroissance temporelle des poids des arcs, que nous détaillons dans la section 7.1.2. Ensuite, nous utilisons le graphe de recommandation obtenu pour

calculer les recommandations top-N avec un PageRank personnalisé et des informations sur la confiance entre les utilisateurs, détaillé dans la section 7.1.3.

7.2 Expérimentations et résultats

Dans les sections précédentes, nous avons défini l'architecture globale de GraFC2T2, notre cadre général des systèmes de recommandation basés sur les graphes, ainsi que les différents modules impliqués. Dans cette section nous décrivons la mise en œuvre des graphes de recommandations proposés dans GraFC2T2. Pour les expérimentations, nous utilisons le même protocole que celui décrit dans la section 6.3.3 du chapitre 6. Dans la suite nous présentons le protocole expérimental et les résultats obtenus.

7.2.1 Description du protocole

Dans cette section, nous réutilisons les jeux de données Epinions et Ciao ainsi que les trois métriques d'évaluation F1-score, Hit ratio et *Mean Averaged Precision* (MAP) qui ont été définies dans la section 2.3 du chapitre 2. Pour avoir les meilleures performances des systèmes de recommandations, nous utilisons de nouveau la stratégie d'optimisation *Randomized Search Cross−Validation* [18]. Les détails sont apportés dans les sous-sections qui suivent.

Jeux de données

Nous utilisons les jeux de données Epinions et Ciao [2] [130], qui sont accessibles publiquement sur internet, et qui contiennent des critiques des utilisateurs sur des produits variés (santé, audiovisuel, électroménager). Ces deux jeux de données ont la particularité de contenir des informations sur les caractéristiques descriptives (catégories) des produits, les horodatages des actions des utilisateurs sur les produits ainsi qu'un ensemble de données sur les relations de confiance locale explicite entre les utilisateurs. Les autres jeux de données à savoir CiteUlike, Delicious, Last.fm et Ponpare, utilisés dans la section 6.3.3 du chapitre 6 ne contiennent pas des informations sur la confiance explicite entre les utilisateurs. C'est la raison pour laquelle nous ne présentons pas les résultats obtenus sur ces jeux de données dans cette section, mais plutôt dans la section A.2 des annexes.

Les données disponibles dans Epinions et Ciao sont un ensemble de quadruplets $\{(t, u, i, c, r)\}$ qui signifient que l'utilisateur u a attribué la note $r \in \{1, 2, 3, 4, 5\}$ au produit i à l'instant t et c est une catégorie du produit i. En plus de ces quadruplets, il y a les données explicites sur les relations de confiance entre les utilisateurs. Plus précisément, pour chaque utilisateur u, l'ensemble TR_u des utilisateurs à qui u fait confiance est donné ; c'est cet ensemble qui est utilisé dans la méthode ET.

2. `https://www.cse.msu.edu/~tangjili/trust.html`

GraFC2T2 n'utilise pas les notes explicites que les utilisateurs attribuent aux produits, mais utilise uniquement les liens implicites et positifs entre les utilisateurs et les produits qu'ils sélectionnent. A cet effet, nous avons considéré uniquement les quadruplets (t, u, i, c, r) dans lesquels la note r que l'utilisateur u a attribuée est supérieure ou égale à 2.5 et également supérieure à la moyenne des notes de u. La table 7.1 fournit les informations essentielles sur les jeux de données Epinions et Ciao après application du filtre : dates de début et de fin, nombres d'utilisateurs et de produits distincts, nombre de catégories des produits, nombre de liens utilisateur-produit (Nb. (u,i)), nombre de relations de confiance explicites (Nb. (u,u)), la densité des relations utilisateur-produit δ_r et enfin la densité des relations de confiance δ_t.

TABLE 7.1 – Statistiques sur les jeux de données Epinions et Ciao

	Date début	Date Fin	$\|U\|$	$\|I\|$	$\|C\|$	Nb. (u,i)	Nb. (u,u)	δ_r	δ_t
Epinions	2010-01-01	2010-12-31	1 843	15 899	24	17 722	4 867	0.06%	0.14%
Ciao	2007-01-01	2010-12-31	879	6 005	6	8 109	23 121	0.15%	3%

Protocole d'évaluation des performances

Le processus d'évaluation des systèmes de recommandation utilisé ici est le même que celui de la 6.3.3 du chapitre 6. C'est une application du protocole de validation croisée à fenêtres de temps de taille croissante et la séparation des jeux d'apprentissage et de test par une ligne temporelle. Pour l'évaluation de la qualité des recommandations top-N, nous utilisons les mêmes métriques : F1-score pour évaluer la qualité de prédiction du système de recommandation, Hit ratio pour évaluer la proportion d'utilisateurs satisfaits par le système de recommandation et enfin *Mean Average Precision* (MAP) pour évaluer la rapidité d'accès aux bonnes recommandations par les utilisateurs.

Performances optimales des systèmes de recommandation

Afin de comparer les systèmes de recommandation construits à partir de GraFC2T2, nous devons nous rassurer d'avoir les meilleures performances de ces systèmes et pour chacune des métriques F1-score, Hit ratio et MAP. La majorité des graphes de recommandation de GraFC2T2 ont plusieurs paramètres et chaque paramètre a un espace de valeurs infini, nous fixons un ensemble de valeurs qui peuvent être attribuées à chacun des paramètres. Le tableau 7.2 présente la liste de ces valeurs. Pour l'optimisation des performances, nous utilisons la stratégie *Randomized Search Cross−Validation* [18] exactement comme dans la section 6.3.3 du chapitre 6.

TABLE 7.2 – Valeurs prédéfinies des paramètres

	Description du paramètre	**Valeurs prédéfinies**
$\boldsymbol{\Delta}$	Durée des sessions court terme du STG	7, 30, 90, 180, 365, 540, 730 jours
$\boldsymbol{\beta}$	Préférence à long terme du STG	0.1, 0.3, 0.5, 0.7, 0.9
$\boldsymbol{\tau_0}$	Demi-vie des fonctions EDF et LDF	7, 30, 90, 180, 365, 540, 730 jours
$\boldsymbol{K}$	Pente de la courbe de la fonction LDF	0.1, 0.5, 1, 5, 10, 50, 100
$\boldsymbol{\gamma}$	Influence des utilisateurs de confiance	0.05, 0.1, 0.15, 0.3, 0.5, 0.7, 0.9
$\boldsymbol{\alpha}$	Facteur d'amortissement du PageRank	0.05, 0.1, 0.15, 0.3, 0.5, 0.7, 0.9

7.2.2 Résultats des recommandations top-10

Dans cette section, nous présentons les résultats des recommandations top-10 pour les jeux de données Epinions et Ciao. Nous avons choisi $N = 10$ comme par exemple dans [142, 19] et d'autres valeurs que nous avons testées ont donné des résultats similaires, voir la section A.1 des annexes. Le tableau 7.3 présente les résultats obtenus.

Dans ces tableaux, chaque colonne correspond à une métrique d'évaluation $M \in \{F1, HR, MAP\}$ et à un graphe de base $G \in \{BIP, STG, LSG\}$, et chaque ligne correspond à une combinaison des méthodes d'intégration des informations secondaires : la confiance (ET, IT), le temps (EDF, LDF) et les attributs de description des produits (CI, CIU). Chaque cellule contient la valeur de la métrique d'évaluation M pour le système de recommandation constitué du graphe de base G et d'une combinaison des informations secondaires en ligne.

Les cellules de couleur blanche correspondent aux meilleures performances et les cellules sombres correspondent aux pires performances suivant la métrique en colonne. En ce qui concerne la couleur des combinaisons des méthodes d'insertion des informations secondaires, la couleur rouge correspond à une seule information secondaire, la couleur bleu correspond à deux informations secondaires et la couleur verte correspond à trois informations secondaires.

Nous résumons les trois meilleurs résultats obtenus pour chaque graphe de base dans le tableau 7.4. Pour chaque graphe de recommandation de base en colonne et chaque mesure d'évaluation en ligne, nous avons sélectionné les trois graphes de recommandation qui offrent les meilleures performances. Chaque ligne contient la performance du graphe de base (*Basic*) - sans information secondaire, la meilleure performance (*Best*) - obtenue après usage des méthodes d'intégration des informations secondaires, le pourcentage d'amélioration (*Imp.*) effectué par *Best* comparé à *Basic* et enfin le nom du graphe de recommandation qui a la meilleure performance.

Lorsqu'on observe le contenu du tableau 7.4, on constate que les graphes étendus par des informations secondaires améliorent les performances des graphes de base d'au moins 46% dans Epinions et d'au moins 41% dans Ciao. On constate également que la meilleure combinaison des méthodes d'intégration des informations secondaires dans Epinions est CIU-EDF-IT pour les graphes de base BIP et STG, et CIU-LDF-IT pour le graphe de

TABLE 7.3 – Performances optimales des graphes construits à partir de GraFC2T2 pour les recommandations top-10 dans les jeux de données Epinions et Ciao. Chaque cellule contient la valeur de la métrique d'évaluation en colonne pour le système de recommandation constitué du graphe de base en colonne et de la combinaison des méthodes d'intégration des informations secondaires en ligne. Les cellules de couleur blanche correspondent aux meilleures performances et les cellules sombres correspondent aux performances moins intéressantes suivant la métrique en colonne.

EPINIONS	F1@10			HR@10			MAP@10		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	2.18	2.0	1.14	5.17	4.77	4.24	2.23	2.17	1.71
ET	1.81	1.86	1.14	4.64	4.64	4.11	2.04	2.07	1.66
IT	2.17	2.42	1.61	5.44	5.44	5.31	2.29	2.34	2.24
EDF	3.74	3.32	2.25	6.1	5.97	5.7	2.31	2.35	2.32
LDF	3.16	2.63	2.26	5.97	5.31	5.97	2.24	2.09	2.54
CI	2.53	3.0	0.86	5.97	6.23	3.32	2.48	2.73	1.74
CIU	3.29	3.51	0.66	6.37	6.63	2.79	2.66	2.88	1.66
EDF-ET	2.77	3.32	1.77	5.44	5.84	5.04	2.13	1.97	2.09
EDF-IT	3.88	4.43	2.74	7.03	7.16	6.37	2.64	2.57	2.42
LDF-ET	2.12	2.63	1.58	4.91	5.31	4.77	2.03	2.1	2.13
LDF-IT	3.1	2.77	3.29	6.1	6.5	7.16	2.98	2.98	3.01
CI-ET	2.44	3.03	0.92	5.84	6.23	3.45	2.28	2.67	1.74
CI-IT	3.12	3.14	1.77	6.37	6.37	5.44	2.49	2.66	2.15
CIU-ET	3.03	3.51	0.66	6.1	6.63	2.79	2.43	2.9	1.66
CIU-IT	3.89	3.72	1.99	7.03	7.03	5.44	2.79	2.88	2.25
CI-EDF	4.88	4.84	0.91	7.69	6.9	3.32	3.06	2.73	1.74
CI-LDF	4.91	3.56	1.1	7.03	6.5	3.98	2.61	2.67	1.75
CIU-EDF	4.51	6.48	0.7	7.69	7.69	2.92	3.23	3.03	1.66
CIU-LDF	5.29	4.49	0.9	7.16	6.76	3.58	2.89	2.85	1.72
CI-EDF-ET	3.84	4.73	0.98	6.63	6.76	3.45	2.44	2.46	1.74
CI-EDF-IT	4.85	4.47	1.69	7.43	6.76	5.31	2.8	2.7	2.17
CI-LDF-ET	3.02	3.39	0.92	5.97	6.37	3.45	2.28	2.66	1.74
CI-LDF-IT	3.81	4.02	3.41	6.76	6.63	7.29	3.0	3.0	2.66
CIU-EDF-ET	4.75	6.13	0.7	7.16	7.43	2.92	2.64	2.95	1.66
CIU-EDF-IT	6.34	7.66	1.99	7.82	7.96	5.44	3.32	3.18	2.27
CIU-LDF-ET	4.06	3.78	0.7	6.63	6.63	2.92	2.48	2.78	1.66
CIU-LDF-IT	5.69	4.49	3.68	7.82	7.03	7.03	3.18	3.07	3.17

CIAO	F1@10			HR@10			MAP@10		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.18	1.48	1.45	5.26	5.63	6.53	1.9	1.99	2.24
ET	1.08	1.44	1.5	5.08	5.44	6.72	1.74	1.9	2.17
IT	2.18	1.92	2.25	7.62	7.62	7.26	2.39	2.39	2.28
EDF	1.63	1.7	2.0	6.35	5.99	7.44	2.03	2.26	2.34
LDF	2.02	1.74	3.27	7.26	7.44	9.26	2.63	2.84	3.51
CI	1.25	2.14	1.25	6.53	6.35	5.26	2.04	2.14	1.7
CIU	2.38	4.56	1.24	7.08	8.53	4.9	2.39	3.01	1.48
EDF-ET	1.17	1.47	1.41	5.26	5.81	6.53	1.68	2.1	2.18
EDF-IT	2.13	3.08	2.37	9.98	9.44	7.26	3.04	2.84	2.32
LDF-ET	1.18	1.49	1.5	5.44	5.63	6.72	1.81	2.01	2.18
LDF-IT	2.66	2.76	2.76	8.53	8.53	8.71	2.91	2.96	2.66
CI-ET	1.39	1.66	1.51	5.99	6.17	5.63	1.97	2.06	1.72
CI-IT	2.41	2.25	2.84	7.8	7.8	8.53	2.47	2.42	2.43
CIU-ET	2.02	4.0	1.31	6.9	8.53	5.08	2.27	3.12	1.5
CIU-IT	2.76	4.21	2.96	8.53	8.53	8.35	2.82	2.96	2.44
CI-EDF	2.58	3.15	1.66	7.62	8.17	6.17	2.33	2.76	1.77
CI-LDF	3.38	2.91	2.37	8.35	8.89	7.62	2.87	2.86	2.85
CIU-EDF	3.46	4.46	1.44	8.71	9.26	5.44	3.24	3.29	1.65
CIU-LDF	7.74	5.79	2.11	9.98	9.8	6.9	3.46	3.31	2.68
CI-EDF-ET	1.86	2.11	1.63	7.26	6.72	5.99	2.27	2.37	1.81
CI-EDF-IT	3.42	3.54	2.9	9.8	9.8	7.99	3.09	3.14	2.46
CI-LDF-ET	1.66	2.67	1.74	6.35	7.08	5.81	1.98	2.12	1.77
CI-LDF-IT	4.56	4.89	2.64	10.7	11.3	8.53	3.19	3.16	3.16
CIU-EDF-ET	2.69	4.79	1.53	8.53	9.26	5.63	2.76	3.09	1.65
CIU-EDF-IT	4.62	5.1	2.88	10.3	10.5	8.35	3.32	3.37	2.38
CIU-LDF-ET	2.73	6.42	1.45	8.35	9.98	5.44	2.29	3.34	1.64
CIU-LDF-IT	5.07	6.11	2.51	11.1	11.1	9.07	3.34	3.35	3.18

TABLE 7.4 – Comparaison des trois meilleures performances des graphes de recommandation étendus par des informations secondaires avec celles des graphes de recommandation de base associés. Nous présentons les pourcentages d'amélioration obtenues. En ce qui concerne les noms des meilleurs graphes, la couleur rouge correspond à une seule information secondaire, la couleur bleu correspond à deux informations secondaires et la couleur verte correspond à trois informations secondaires.

Epinions Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@10	1	2.18	6.34	190%	CIU-EDF-IT	2.0	7.66	282%	CIU-EDF-IT	1.14	3.68	221%	CIU-LDF-IT
	2	2.18	5.69	160%	CIU-LDF-IT	2.0	6.48	223%	CIU-EDF	1.14	3.41	197%	CI-LDF-IT
	3	2.18	5.29	142%	CIU-LDF	2.0	6.13	206%	CIU-EDF-ET	1.14	3.29	187%	LDF-IT
H@10	1	5.17	7.82	51%	CIU-EDF-IT	4.77	7.96	66%	CIU-EDF-IT	4.24	7.29	71%	CI-LDF-IT
	2	5.17	7.82	51%	CIU-LDF-IT	4.77	7.69	61%	CIU-EDF	4.24	7.16	68%	LDF-IT
	3	5.17	7.69	48%	CI-EDF	4.77	7.43	55%	CIU-EDF-ET	4.24	7.03	65%	CIU-LDF-IT
M@10	1	2.23	3.32	48%	CIU-EDF-IT	2.17	3.18	46%	CIU-EDF-IT	1.71	3.17	85%	CIU-LDF-IT
	2	2.23	3.23	45%	CIU-EDF	2.17	3.07	41%	CIU-LDF-IT	1.71	3.01	76%	LDF-IT
	3	2.23	3.18	42%	CIU-LDF-IT	2.17	3.03	39%	CIU-EDF	1.71	2.66	55%	CI-LDF-IT

Ciao Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@10	1	1.18	7.74	556%	CIU-LDF	1.48	6.42	332%	CIU-LDF-ET	1.45	3.27	125%	LDF
	2	1.18	5.07	330%	CIU-LDF-IT	1.48	6.11	311%	CIU-LDF-IT	1.45	2.96	104%	CIU-IT
	3	1.18	4.62	291%	CIU-EDF-IT	1.48	5.79	290%	CIU-LDF	1.45	2.9	99%	CI-EDF-IT
H@10	1	5.26	11.1	110%	CIU-LDF-IT	5.63	11.3	100%	CI-LDF-IT	6.53	9.26	41%	LDF
	2	5.26	10.7	103%	CI-LDF-IT	5.63	11.1	96%	CIU-LDF-IT	6.53	9.07	38%	CIU-LDF-IT
	3	5.26	10.3	96%	CIU-EDF-IT	5.63	10.5	87%	CIU-EDF-IT	6.53	8.71	33%	LDF-IT
M@10	1	1.9	3.46	82%	CIU-LDF	1.99	3.37	69%	CIU-EDF-IT	2.24	3.51	57%	LDF
	2	1.9	3.34	76%	CIU-LDF-IT	1.99	3.35	68%	CIU-LDF-IT	2.24	3.18	42%	CIU-LDF-IT
	3	1.9	3.32	74%	CIU-EDF-IT	1.99	3.34	67%	CIU-LDF-ET	2.24	3.16	41%	CI-LDF-IT

base LSG. Dans le cas de Ciao, les meilleurs résultats sont obtenus par la combinaison CIU-LDF-IT pour tous les graphes de base. Ces observations confirment clairement la pertinence des graphes étendus simultanément par des informations sur le contenu, le temps et la confiance.

7.3 Analyse des meilleurs résultats

Dans cette section nous présentons l'impact de l'ajout des informations secondaires dans les graphes de base, puis des commentaires sur les meilleures valeurs des paramètres impliqués dans les systèmes de recommandation construits à partir de GraFC2T2. La section se termine par une comparaison des meilleures performances de CraFC2T2 avec d'autres systèmes de recommandation top-N dédiés aux données implicites.

7.3.1 Effets des informations secondaires

Nous donnons maintenant des détails sur l'impact des informations secondaires et leur combinaison dans GraFC2T2. Cet impact dépend du contexte, car les comportements observés varient en fonction des jeux de données ; on peut cependant facilement tester le framework GraFC2T2 avec ses propres données et découvrir les meilleurs choix de combinaison des informations secondaires à faire pour ces cas. Les détails fournis dans la suite sont apportés à titre d'illustration pour les jeux de données Epinions et Ciao.

Aucune information secondaire

Lorsque nous examinons les performances des graphes de recommandation de base sans aucune information secondaire, dans le cas d'Epinions, BIP donne les meilleurs résultats pour toutes les métriques d'évaluation. Par contre dans Ciao, LSG a les meilleures performances en HR et MAP, tandis que STG donne la meilleure performance de la métrique F1.

Une seule information secondaire

Si nous n'incluons qu'un seul type d'information secondaire, nous observons que :

- la confiance explicite (ET) n'améliore pas les résultats, contrairement à la confiance implicite (IT) pour tous les graphes de recommandation de base.
- L'insertion des fonctions de décroissance temporelle (EDF ou LDF) produit toujours des améliorations.
- Enfin, l'insertion des attributs de description des produits augmentent les performances pour BIP et STG mais pas pour LSG.

Dans le cas du jeu de données Epinions, le meilleur graphe de recommandation avec un seul type d'information secondaire est BIP-EDF pour la métrique F1 et STG-CIU pour les métriques HR et MAP. En ce qui concerne le jeu de données Ciao, le meilleur est LSG-LDF pour les métriques HR et MAP, et STG-CIU est le meilleur pour la métrique F1. Cela montre que l'impact d'un type unique d'informations secondaires dépend fortement du graphe de base considéré et des données utilisées.

Deux informations secondaires

Les systèmes de recommandations qui utilisent deux types d'informations secondaires simultanément ont des résultats bien meilleurs que ceux qui intègrent un seul type d'information secondaire. Par exemple, dans le cas du jeu de données Epinions, les performances augmentent de 3,74 à 6,48% pour la métrique F1, de 6,63 à 7,69% pour la métrique HR et de 2,88 à 3,23% pour la métrique MAP.

Lorsqu'on s'attarde sur les systèmes de recommandation de GraFC2T2 qui intègrent simultanément deux informations secondaires, on fait les constats suivants :

- Les combinaisons d'une fonction de décroissance temporelle et de la confiance implicite donnent de meilleurs résultats que l'ajout uniquement d'une fonction de décroissance temporelle ou de la confiance implicite pris séparément.
- De même, les combinaisons des méthodes d'ajout des attributs basés sur le contenu avec l'insertion de la confiance implicite sont préférables à chacune de ces informations secondaires prises séparément, mais généralement moins intéressantes que la combinaison fonction de décroissance temporelle + confiance implicite.
- Les combinaisons des méthodes d'insertion des attributs basés sur le contenu avec une fonction de décroissance temporelle produisent généralement de meilleures améliorations pour BIP et STG, mais aucune amélioration pour LSG. Par exemple, dans Epinions, BIP-CI-EDF et BIP-CIU-EDF sont les meilleurs graphes de recommandation qui intègrent deux informations secondaires. Dans Ciao, BIP-CIU-LDF est toujours le meilleur de cette catégorie.

La dernière observation confirme la pertinence des graphes de recommandation qui intègrent simultanément les attributs de description des produits et les fonctions de décroissance temporelle comme le *Time−weight Content−based Session−based Temporal Graph* que nous avons proposé dans [100] et présenté dans la section 6.2 du chapitre 6.

Trois informations secondaires

L'utilisation de trois types d'informations secondaires n'améliore pas beaucoup les meilleures performances obtenues avec deux types d'informations secondaires. Par exemple, dans Epinions, les performances augmentent de 6,48 à 7,66% pour la métrique F1, de 7,69 à 7,96% pour la métrique HR, et de 3,23 à 3,32% pour la métrique MAP.

Néanmoins, le tableau 7.4 montre que les graphes de recommandation avec trois types d'informations secondaires sont de loin les plus fréquents parmi les meilleurs. Pour cette raison, nous recommandons d'utiliser simultanément les attributs de description des produits, les fonctions de décroissance temporelle et les informations sur la confiance dans le but d'obtenir de bons résultats.

7.3.2 Meilleures valeurs des paramètres

Dans cette section, nous nous concentrons uniquement sur les graphes de recommandation associés aux combinaisons CIU-EDF-IT et CIU-LDF-IT des méthodes d'intégration des informations secondaires. Ces deux combinaisons sont les plus courantes pour l'obtention des meilleures performances dans le tableau 7.4. Nous avons fait les observations suivantes :

Cas du jeu de données Epinions.

- pour la combinaison CIU-EDF-IT, la durée d'une session à court terme $\Delta = 7$ et la proportion des préférences à long terme injectées $\beta = 0.5$ pour tous les graphes de base. La demi-vie des fonctions de décroissance temporelle $\tau_0 = 90$ pour BIP et STG, et 180 pour LSG ; Le poids de l'influence des utilisateurs de confiance $\gamma \in \{0.15, 0.3\}$ pour BIP et STG, et $\gamma = 0.9$ pour LSG, et la meilleure valeur du facteur d'amortissement du PageRank $\alpha = 0.9$ pour tous les graphes de base.
- Pour la combinaison CIU-LDF-IT, $\Delta = 365$ et $\beta = 0.7$ pour tous les graphes de base. $\tau_0 \in \{30, 90\}$ pour les graphes BIP et STG et $\tau_0 = 7\}$ pour LSG. La pente de la courbe de la fonction logistique décroissante LDF est $K = 0.5$ pour BIP, 100 pour STG et 5 pour LSG. Le poids de l'influence de la confiance $\gamma \in \{0.1, 0.15\}$ pour BIP et STG, et $\gamma = 0.9$ pour LSG et la meilleure valeur du facteur d'amortissement du PageRank $\alpha \in \{0.7, 0.9\}$ pour tous les graphes de base.

Cas du jeu de données Ciao.

- Pour la combinaison CIU-EDF-IT, on a $\Delta = 180$, $\beta = 0.3$, $\tau_0 = 180$, $\gamma = 0.9$ et $\alpha = 0.9$ pour tous les graphes de base.
- Pour la combinaison CIU-LDF-IT, on a $\Delta = 540$, $\beta = 0.1$ pour tous les graphes de base. $\tau_0 = 365$ pour BIP et STG, et 180 pour LSG. La pente de la fonction LDF $K = 10$ pour BIP et STG, et 100 pour LSG. Pour ce qui est du poids de la confiance $\gamma \in \{0.7, 0.9\}$, et pour le facteur d'amortissement $\alpha = 0.9$ pour tous les graphes de base.

Les valeurs de ces paramètres indiquent que, dans Epinions, les poids des données utilisées (poids des arcs) diminuent plus rapidement que dans Ciao ; la demi-vie τ_0 est petit dans Epinions $\{7, 30, 90\}$ et est plus grand dans Ciao $\{180, 365\}$. En ce qui concerne la confiance, γ est toujours élevé dans Ciao $\{0.7, 0.9\}$ et plus petit dans Epinions $\{0.1, 0.15, 0.3\}$, ce qui montre que l'influence de la confiance implicite est plus importante dans Ciao. Cependant, le poids de cette influence doit toujours être grand pour le graphe LSG ($\gamma = 0.9$) dans les deux jeux de données.

7.3.3 Comparaison avec d'autres systèmes de recommandation

Pour évaluer la pertinence de GraFC2T2, nous comparons ses meilleures performances à celles des systèmes de recommandation Top-N basés sur des données implicites. Les modèles considérés sont les suivants :

- le système de base de recommandation des produits les plus populaires (POP) qui calcule le score de classement d'un produit en fonction de sa popularité ;

- les techniques des K-plus proches voisins orienté utilisateur (UKNN) et orienté produit (IKNN) [72, 93] ;
- un algorithme standard des recommandations basées sur les réseaux bayésiens *Bayesian Personalized Ranking* (BPR) [118] ;
- l'algorithme de recommandation SLIM (*Sparse LInear Methods for top − N recommender systems*) [99] ;
- le système de recommandation CLiMF (Collaborative Less-Is-More Filtering) du meilleur papier de la conférence ACM RecSys 2012 [3] [125] ;
- et enfin le système de recommandation ALS (Alternating Least Squares) une technique de factorisation matricielle avec alternance des moindres carré [62].

Nous utilisons une fois de plus la techniques *Randomized Search Cross−Validation* [18] pour obtenir les meilleures performances des systèmes de recommandation considérés dans notre processus de comparaison. Pour les modèles UKNN et IKNN, 10 paramètres sont générés, de sorte que la taille du voisinage $k \in \{10, 20, 30, 40, 50, 80, 100, 150, 200, 500\}$. Pour les modèles BPR, SLIM, CLIMF et ALS, 50 paramètres sont générés de sorte que le nombre de facteurs latents $l \in \{10, 20, 30, 50, 100, 200, 500\}$, le taux d'apprentissage et tous les biais de régularisations sont pris dans $\{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05\}$. Le tableau 7.5 présente les meilleurs résultats obtenus pour ces systèmes de recommandation et une comparaison avec ceux obtenus avec GraFC2T2. Dans ce tableau, on constate que les meilleures performances atteintes avec GraFC2T2 sont toujours les meilleures.

Table 7.5 – Résultats des expérimentations avec les jeux de données Epinions et Ciao pour le calcul des recommandations top-10. Les meilleures performances sont en gras.

		POP	UKNN	IKNN	BPR	SLIM	CLIMF	ALS	GraFC2T2
Epinions	F@10	1.79	0.30	0.70	0.15	0.82	1.97	2.27	**7.66**
	H@10	4.91	1.46	2.79	0.80	2.92	5.17	4.91	**7.96**
	M@10	2.07	0.61	1.29	0.45	1.16	2.15	2.26	**3.32**
Ciao	F@10	2.26	0.31	0.94	0.22	1.49	3.38	2.10	**7.74**
	H@10	7.62	1.63	4.17	1.27	5.08	8.71	6.90	**11.3**
	M@10	2.62	0.59	1.65	0.56	2.09	3.06	2.46	**3.51**

De plus, en comparant les performances de GraFC2T2 avec les résultats obtenus pour Epinions (MAP@10 = 1.32%) et Ciao (MAP@10 = 3.07%) par la technique TDAE *Trust−aware Denoising Auto Encoder* basée sur les réseaux de neurones profonds [108] confirme la pertinence de GraFC2T2. Ceci montre que les graphes de recommandation peuvent atteindre des performances comparables, voir meilleures à la factorisation matricielle et aux approches d'apprentissage profond, lorsque ces graphes sont étendus simul-

3. Les conférences ACM RecSys sont les plus prestigieuses du domaine des systèmes de recommandation. Se sont des conférences annuelles internationales avec comité de lecture. https ://recsys.acm.org

tanément par des informations basées sur le contenu des produits, les horodatages des actions utilisateur et la confiance entre les utilisateurs.

Notons que la technique la plus élémentaire (POP) permet d'obtenir de meilleurs résultats que BPR, SLIM, UBCF et IBCF. Cela indique que les utilisateurs ont tendance à consommer des produits populaires. Ce travail n'est pas le premier dans lequel la technique POP est meilleure que BPR ou d'autres modèles de la factorisation matricielle ; c'est le cas dans [151, 52].

7.4 Résumé

Notre objectif principal dans ce chapitre était de montrer que l'intégration de plusieurs informations secondaires améliore la qualité des graphes construits pour la recommandation top-N à partir des données implicites. A cet effet, nous avons conçu et mis en œuvre GraFC2T2, un cadre général des graphes de recommandation enrichis par des informations sur le contenu des produits, sur les horodatages des actions utilisateur et sur les relations de confiance entre les utilisateurs. Plus précisément, GraFC2T2 étend le graphe biparti classique (BIP), *Session−based Temporal Graph* (STG) et *Link Stream Graph* (LSG) en y intégrant un nouveau type de nœud pour les caractéristiques des produits, une fonction de pénalisation des poids des vieux arcs et en utilisant des données sur la confiance pour personnaliser l'algorithme PageRank pour le calcul des recommandations top-N à partir des graphes construits.

Les expérimentations que nous avons menées sur les jeux de données Epinions et Ciao avec les métriques d'évaluation F1-score, Hit ratio et *Mean Average Precision* (MAP) montrent que les meilleures performances sont toujours atteintes par des graphes qui intègrent au moins deux informations secondaires, et ceux qui pénalisent les poids des arcs dans le temps sont généralement meilleurs que les autres. Les améliorations résultantes sont d'au moins 41%. De plus, la comparaison avec des techniques de factorisation matricielle, des K-plus proches voisins basées sur les utilisateurs et les produits confirme la pertinence de GraFC2T2 pour les recommandations Top-N.

Cependant, les taux d'amélioration obtenus avec l'extension des graphes de recommandation par l'intégration des informations secondaires ne garantissent pas qu'on puisse avoir de telles améliorations pour d'autres types de systèmes de recommandation tels que la factorisation matricielle et les réseaux de neurones. Par conséquent, l'intégration simultanée des informations basées sur le contenu, sur le temps et sur la confiance dans de telles techniques est une perspective de ce travail.

Chapitre 8

Conclusion

Sommaire

8.1 Bilan des travaux

Dans cette thèse, nous nous sommes intéressés à la conception de systèmes de recommandation top-N à partir des données implicites modélisées par des flots de liens. Les flots de liens conservent les traces des actions des utilisateurs sur les produits tout en conservant les instants d'apparition de ces actions. De plus, les flots de liens permettent aisément d'extraire des graphes qui sont des modèles simples et faciles à interpréter. D'où l'objectif de notre thèse d'exploiter des concepts temporels et topologiques extraits des flots de liens en vue d'une amélioration des systèmes de recommandation top-N existant et particulièrement des graphes de recommandation.

Pour atteindre cet objectif, nous avons fait le point sur le filtrage collaboratif qui est l'approche de recommandation la plus étudiée, la plus utilisée et la catégorie dans laquelle on classe les graphes de recommandation. Cependant, ces techniques ont des limites comme le manque de données ou le problème du démarrage à froid des utilisateurs et des produits. D'où la nécessité d'intégrer les informations secondaires comme les attributs descriptifs des produits et les informations sur les relations de confiance entre les utilisateurs. La prise en compte de ces données supplémentaires a permis dans le passé d'améliorer la qualité des recommandations, sans toutefois atteindre les meilleures performances car la dynamique temporelle des préférences des utilisateurs n'était pas prise

en compte.

Afin d'améliorer la qualité des recommandations, la dynamique temporelle à été intégrée dans certaines techniques de filtrage collaboratif, comme les K-plus proches voisins et les techniques basées sur un modèle d'apprentissage automatique comme SVM (Séparateurs à Vaste Marge) et la factorisation des tenseurs, qui intégraient déjà soit des informations sur le contenu des produits, soit des informations sur les relations de confiance entre les utilisateurs. Cependant ceci a rarement été fait dans les graphes de recommandation. De plus très peu de travaux sur les graphes de recommandation tiennent compte de la confiance et aucun de ces travaux n'intègre simultanément les informations sur le contenu des produits, sur la dynamique temporelle des préférences des utilisateurs et sur les relations de confiance entre les utilisateurs.

Dans le but de dépasser les limites présentées précédemment, notre première contribution est l'intégration des fonctions de décroissance temporelle dans les graphes de recommandation. Cet ajout pénalise les arcs les plus anciens et donne plus d'importance aux récentes actions des utilisateurs qui traduisent mieux leurs préférences actuelles. Les expérimentations sur les jeux de données de CiteUlike, Delicious et Last.fm ont démontré la pertinence de cet ajout dans le graphe *Session−based Temporal Graph* (STG) qui est l'un des graphes les plus populaires de la littérature et qui prend en compte la dynamique temporelle en combinant les préférences à long terme et à court terme, mais qui ne fait aucune différence entre les poids des arcs les plus anciens et les plus récents.

Notre seconde contribution s'inscrit toujours dans la dynamique temporelle des graphes de recommandation. En effet, les graphes de recommandation de la littérature ne considèrent pas le temps de manière continue dans leur structure ; le temps est soit ignoré (comme dans le graphe biparti classique (BIP)), soit divisé en tranches (comme dans le graphe STG). Nous avons dépassé cette limite en proposant le *Link Stream Graph* (LSG) dont la pertinence a été montrée suite à une série d'expérimentations sur les jeux de données de CiteUlike, Delicious, Last.fm, Ponpare, Epinions et Ciao en utilisant les métriques d'évaluation F1-score, Hit ratio (HR) et *Mean Average Precision* (MAP). Dans 12 cas sur les 18 possibles, le graphe LSG a la meilleure performance.

Notre dernière contribution est GraFC2T2, un cadre général qui permet d'enrichir les graphes de recommandation par des informations sur le contenu des produits, sur la dynamique temporelle des préférences des utilisateurs et sur les relations de confiance entre les utilisateurs. Plus précisément, GraFC2T2 permet d'étendre les graphes BIP, STG et LSG en y intégrant un nouveau type de nœud pour les caractéristiques des produits, une fonction de pénalisation des arcs anciens et en utilisant des données sur la confiance pour personnaliser l'algorithme PageRank pour le calcul des recommandations top-N à partir des graphes construits. Le code source du framework GraFC2T2 est accessible via le lien `https://github.com/nzekonarmel/GraFC2T2`.

Les expérimentations que nous avons menées sur les jeux de données Epinions et Ciao avec les métriques d'évaluation F1-score, HR et MAP montrent que les meilleures perfor-

mances sont toujours atteintes par des graphes qui intègrent au moins deux informations secondaires, et ceux qui pénalisent les poids des arcs dans le temps sont généralement meilleurs que les autres. Les améliorations résultantes sont d'au moins 41%, et la comparaison avec des techniques de factorisation matricielle et les K-plus proches voisins, confirme la pertinence de GraFC2T2 pour les recommandations Top-N.

8.2 Perspectives

Le travail proposé dans cette thèse permet d'étendre des modèles de graphes en utilisant plusieurs informations secondaires. Nous envisageons plusieurs perspectives qui portent principalement sur :

- Estimer de bonnes valeurs des paramètres
- Répliquer l'intégration de plusieurs informations secondaires dans d'autres modèles
- Prendre en compte plus d'informations secondaires
- Tenir compte des feedbacks négatifs

8.2.1 Estimer les bonnes valeurs des paramètres

Les graphes de recommandation considérés dans cette thèse imposent un certain nombre de paramètres comme la durée d'une tranche de temps (court terme) utilisée pour construire les nœuds session du graphe STG ; ou la durée de demi-vie τ_0 qui est utilisée de telle sorte que les poids des arcs diminuent de moitié tous les τ_0 unités de temps. D'autre part, des fonctions prédéfinies sont utilisées pour décrire la dynamique temporelle des préférences des utilisateurs. Toutes ces considérations sont fixées sans tenir compte des données.

A cet effet, des travaux futurs peuvent être consacrés à l'analyse des données de manière à avoir une meilleure estimation des paramètres et des fonctions pertinentes pour intégrer la dynamique temporelle. Cette démarche limitera le recours à la méthode *Randomized Search Cross−Validation* [18] pour l'optimisation des performances des graphes de recommandation, d'autant plus que cette stratégie n'est pas applicable dans certains contextes réels où les données sont plus volumineuses et dans lesquels on ne dispose pas d'assez de temps.

8.2.2 Intégrer plusieurs informations secondaires dans d'autres modèles

Dans cette thèse nous montrons comment étendre les graphes de recommandation en intégrant simultanément des informations sur le contenu des produits, sur la dynamique

temporelle et sur les relations de confiance entre les utilisateurs. L'usage de cette combinaison des informations secondaires n'est pas présent dans la littérature pour d'autres techniques de recommandation comme la factorisation matricielle, les réseaux de neurones ou les K-plus proches voisins. Intégrer à ces approches simultanément ces trois types d'informations secondaires et éventuellement des informations secondaires supplémentaires peut permettre de confirmer que l'usage de plusieurs informations secondaires améliore la qualité des recommandations de manière générale.

8.2.3 Prendre en compte plus d'informations secondaires

Les travaux présentés dans ce manuscrit ne tiennent compte que de trois types d'informations secondaires : informations basées sur le contenu des produits, informations sur la dynamique temporelle, et informations sur les relations de confiance entre les utilisateurs. Cependant, d'autres types d'informations secondaires existent et peuvent être utiles dans le processus de calcul des recommandations. On peut citer les attributs descriptifs des utilisateurs, la méfiance (*distrust*), les positions géographiques et les influenceurs.

- **Attributs descriptifs des produits** : la prise en compte des attributs des profils des utilisateurs seraient d'un apport positif car dans une plateforme de e-commerce, on peut avoir des produits pour les enfants, d'autres pour les jeunes et certains pour les vieillards. Ou encore avoir des produits pour des travailleurs aux salaires importants et d'autres pour des chômeurs ou des travailleurs avec de petits salaires.
- **Informations sur la méfiance** : les informations sur la méfiance (*distrust*) prennent en compte les désaccords entre les utilisateurs. De telles données peuvent être utiles pour éviter de recommander à l'utilisateur cible u, des produits qui sont aimés par des personnes avec qui u est méfiant et à qui il n'accorde aucun crédit. Les informations sur la méfiance aident également à identifier les produits qu'un utilisateur n'aime pas ; si un utilisateur ne fait pas confiance à un autre, alors il n'aime pas les produits que ce dernier lui recommande [2].
- **Positions géographiques** : les informations sur les positions géographiques des utilisateurs peuvent être importantes dans une plateforme de streaming des chansons où les utilisateurs sont dans des pays différents : chaque utilisateur peut avoir tendance à écouter les chansons des artistes du pays dans lequel il se trouve.
- **Influenceurs** : des récents travaux en marketing montrent que les personnes ont tendance à faire plus confiance aux influenceurs qui sont sur les réseaux sociaux comme Youtube, Twitter, Facebook et Instagram, plutôt qu'à leurs amis, pour l'achat de nouveaux produits [47]. La prise en compte des influenceurs et leur impact sur les actions des utilisateurs peuvent être intégrés comme information secondaire supplémentaire comme dans [97]. On pourrait l'intégrer dans le processus de personnalisation du PageRank comme les informations sur la confiance.

8.2.4 Tenir compte des feedbacks négatifs

Dans ce travail nous considérons uniquement les feedbacks positifs des utilisateurs sur les produits. Cependant dans certaines situations, les feedbacks négatifs sont également disponibles. Dans ces cas, prendre en compte ces données permet d'éviter de recommander des produits similaires à ceux que l'utilisateur cible n'aime pas. Pour ce faire, les graphes de recommandation construits doivent tenir compte à la fois des feedbacks négatifs et des feedbacks positifs notamment dans la manière de pondérer les arcs.

Bibliographie

[1] Mohammad Ali Abbasi, Jiliang Tang, and Huan Liu. Trust-aware recommender systems. *Machine Learning book on computational trust, Chapman & Hall/CRC Press*, 2014.

[2] Zeinab Abbassi, Christina Aperjis, and Bernardo A Huberman. Friends versus the crowd : tradeoffs and dynamics. *HP Report*, 2013.

[3] Sibel Adali, Robert Escriva, Mark K Goldberg, Mykola Hayvanovych, Malik Magdon-Ismail, Boleslaw K Szymanski, William A Wallace, and Gregory Williams. Measuring behavioral trust in social networks. In *2010 IEEE International Conference on Intelligence and Security Informatics*, pages 150–152. IEEE, 2010.

[4] PH Aditya, I Budi, and Q Munajat. A comparative analysis of memory-based and model-based collaborative filtering on the implementation of recommender system for e-commerce in indonesia : A case study pt x. In *2016 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pages 303–308. IEEE, 2016.

[5] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems : A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering*, (6) :734–749, 2005.

[6] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2011.

[7] Charu C Aggarwal. *Data mining : the textbook*. Springer, 2015.

[8] Asim Ansari, Skander Essegaier, and Rajeev Kohli. Internet recommendation systems, 2000.

[9] Sylvain Arlot, Alain Celisse, et al. A survey of cross-validation procedures for model selection. *Statistics surveys*, 4 :40–79, 2010.

[10] Thibaud Arnoux, Lionel Tabourier, and Matthieu Latapy. Combining structural and dynamic information to predict activity in link streams. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, pages 935–942. ACM, 2017.

[11] Ricardo Baeza-Yates, Berthier de Araújo Neto Ribeiro, et al. *Modern information retrieval*. 2011.

[12] Marko Balabanović and Yoav Shoham. Fab : content-based, collaborative recommendation. *Communications of the ACM*, 40(3) :66–72, 1997.

[13] Linas Baltrunas and Xavier Amatriain. Towards time-dependant recommendation based on implicit feedback. In *Workshop on context-aware recommender systems (CARS'09)*, pages 25–30. Citeseer, 2009.

[14] Shumeet Baluja, Rohan Seth, D Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. Video suggestion and discovery for youtube : taking random walks through the view graph. In *Proceedings of the 17th international conference on World Wide Web*, pages 895–904. ACM, 2008.

[15] Chumki Basu, Haym Hirsh, William Cohen, et al. Recommendation as classification : Using social and content-based information in recommendation. In *Aaai/iaai*, pages 714–720, 1998.

[16] Robert M Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *icdm*, pages 43–52. IEEE, 2007.

[17] James Bennett, Stan Lanning, et al. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, NY, USA., 2007.

[18] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb) :281–305, 2012.

[19] Daniel Bernardes, Mamadou Diaby, Raphaël Fournier, Françoise FogelmanSoulié, and Emmanuel Viennet. A social formalism and survey for recommender systems. *Acm Sigkdd Explorations Newsletter*, 16(2) :20–37, 2015.

[20] Daniel Billsus and Michael J Pazzani. User modeling for adaptive news access. *User modeling and user-adapted interaction*, 10(2-3) :147–180, 2000.

[21] Toine Bogers. Movie recommendation using random walks over the contextual graph. In *Proc. of the 2nd Intl. Workshop on Context-Aware Recommender Systems*, 2010.

[22] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

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

[24] Robin Burke. Hybrid web recommender systems. In *The adaptive web*, pages 377–408. Springer, 2007.

[25] Pedro G Campos, Alejandro Bellogín, Fernando Díez, and J Enrique Chavarriaga. Simple time-biased knn-based recommendations. In *Proceedings of the Workshop on Context-Aware Movie Recommendation*, pages 20–23. ACM, 2010.

[26] Pedro G Campos, Fernando Díez, and Iván Cantador. Time-aware recommender systems : a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, 24(1-2) :67–119, 2014.

[27] Pedro G Campos Soto. Recommender systems and time context : Characterization of a robust evaluation protocol to increase reliability of measured improvements. 2013.

[28] Oscar Celma. Music recommendation. In *Music recommendation and discovery*, pages 43–85. 2010.

[29] Soumen Chakrabarti. *Mining the Web : Discovering knowledge from hypertext data*. Elsevier, 2002.

[30] Kailong Chen, Tianqi Chen, Guoqing Zheng, Ou Jin, Enpeng Yao, and Yong Yu. Collaborative personalized tweet recommendation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 661–670. ACM, 2012.

[31] Evangelia Christakopoulou and George Karypis. Local item-item models for top-n recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 67–74. ACM, 2016.

[32] Mark Claypool, Anuja Gokhale, Tim Miranda, Paul Murnikov, Dmitry Netes, and Matthew Sartin. Combing content-based and collaborative filters in an online newspaper. 1999.

[33] Michelle Keim Condliff, David D Lewis, David Madigan, and Christian Posse. Bayesian mixed-effects models for recommender systems. In *ACM SIGIR*, volume 99, pages 23–30. Citeseer, 1999.

[34] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 39–46. ACM, 2010.

[35] Paolo Cremonesi and Roberto Turrin. Analysis of cold-start recommendations in iptv systems. In *Proceedings of the third ACM conference on Recommender systems*, pages 233–236. ACM, 2009.

[36] Paolo Cremonesi and Roberto Turrin. Time-evolution of iptv recommender systems. In *Proceedings of the 8th European Conference on Interactive TV and Video*, pages 105–114. ACM, 2010.

[37] Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. Google news personalization : scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, pages 271–280. ACM, 2007.

[38] Luis M De Campos, Juan M Fernández-Luna, Juan F Huete, and Miguel A Rueda-Morales. Combining content-based and collaborative recommendations : A hybrid approach based on bayesian networks. *International Journal of Approximate Reasoning*, 51(7) :785–799, 2010.

[39] Thomas G Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7) :1895–1923, 1998.

[40] Yi Ding and Xue Li. Time weight collaborative filtering. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 485–492. ACM, 2005.

[41] Yi Ding and Xue Li. Time weight collaborative filtering. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 485–492. ACM, 2005.

[42] Georges Dupret. Discounted cumulative gain and user decision models. In *International Symposium on String Processing and Information Retrieval*, pages 2–13. Springer, 2011.

[43] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. Pixie : A system for recommending 3+ billion items to 200+ million users in real-time. In *Proceedings of the 2018 World*

Wide Web Conference on World Wide Web, pages 1775–1784. International World Wide Web Conferences Steering Committee, 2018.

[44] Francois Fouss, Luh Yen, Alain Pirotte, and Marco Saerens. An experimental investigation of graph kernels on a collaborative recommendation task. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 863–868. IEEE, 2006.

[45] Noé Gaumont, Clémence Magnien, and Matthieu Latapy. Finding remarkably dense sequences of contacts in link streams. *Social Network Analysis and Mining*, 6(1) :87, 2016.

[46] Sergiu Gordea and Markus Zanker. Time filtering for better recommendations with small and sparse rating matrices. In *International Conference on Web Information Systems Engineering*, pages 171–183. Springer, 2007.

[47] Johan Grafström, Linnéa Jakobsson, and Philip Wiede. The impact of influencer marketing on consumers' attitudes, 2018.

[48] Benjamin Gras, Armelle Brun, and Anne Boyer. Identifying grey sheep users in collaborative filtering : a distribution-based technique. In *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*, pages 17–26. ACM, 2016.

[49] Benjamin Gras, Armelle Brun, and Anne Boyer. Can matrix factorization improve the accuracy of recommendations provided to grey sheep users ? In *13th International Conference on Web Information Systems and Technologies (WEBIST)*, pages 88–96, 2017.

[50] Asela Gunawardana and Guy Shani. A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research*, 10(Dec) :2935–2962, 2009.

[51] Guibing Guo, Jie Zhang, Daniel Thalmann, Anirban Basu, and Neil Yorke-Smith. From ratings to trust : an empirical study of implicit trust in recommender systems. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, pages 248–253. ACM, 2014.

[52] Guibing Guo, Jie Zhang, Feida Zhu, and Xingwei Wang. Factored similarity models with social trust for top-n item recommendation. *Knowledge-Based Systems*, 122 :17–25, 2017.

[53] Taher H Haveliwala. Topic-sensitive pagerank. In *Proceedings of the 11th international conference on World Wide Web*, pages 517–526. ACM, 2002.

[54] Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. Nais : Neural attentive item similarity model for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 30(12) :2354–2366, 2018.

[55] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pages 173–182. International World Wide Web Conferences Steering Committee, 2017.

[56] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 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*, pages 549–558. ACM, 2016.

[57] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237. ACM, 1999.

[58] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1) :5–53, 2004.

[59] Christoph Hermann. Time-based recommendations for lecture materials. In *Ed-Media+ Innovate Learning*, pages 1028–1033. Association for the Advancement of Computing in Education (AACE), 2010.

[60] Sébastien Heymann. Investigation visuelle d'événements dans un grand flot de liens. In *EGC*, pages 89–100, 2014.

[61] Balázs Hidasi and Domonkos Tikk. Fast als-based tensor factorization for context-aware recommendation from implicit feedback. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 67–82. Springer, 2012.

[62] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. Ieee, 2008.

[63] Zan Huang, Wingyan Chung, and Hsinchun Chen. A graph model for e-commerce recommender systems. *Journal of the American Society for information science and technology*, 55(3) :259–274, 2004.

[64] Zan Huang, Daniel Zeng, and Hsinchun Chen. A comparison of collaborative-filtering recommendation algorithms for e-commerce. *IEEE Intelligent Systems*, 22(5) :68–78, 2007.

[65] Sylvain Iloga, Olivier Romain, and Maurice Tchuenté. A sequential pattern mining approach to design taxonomies for hierarchical music genre recognition. *Pattern Analysis and Applications*, 21(2) :363–380, 2018.

[66] Michael Jahrer, Andreas Töscher, and Robert Legenstein. Combining predictions for accurate recommender systems. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 693–702. ACM, 2010.

[67] Mohsen Jamali and Martin Ester. Trustwalker : a random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 397–406. ACM, 2009.

[68] Mohsen Jamali and Martin Ester. Using a trust network to improve top-n recommendation. In *Proceedings of the third ACM conference on Recommender systems*, pages 181–188. ACM, 2009.

[69] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 135–142. ACM, 2010.

[70] Wenjun Jiang, Guojun Wang, Md Zakirul Alam Bhuiyan, and Jie Wu. Understanding graph-based trust evaluation in online social networks : Methodologies and challenges. *ACM Computing Surveys (CSUR)*, 49(1) :10, 2016.

[71] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. Multiverse recommendation : n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 79–86. ACM, 2010.

[72] George Karypis. Evaluation of item-based top-n recommendation algorithms. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 247–254. ACM, 2001.

[73] Peter Knees and Markus Schedl. A survey of music similarity and recommendation from music context data. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 10(1) :2, 2013.

[74] Bart P Knijnenburg, Martijn C Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction*, 22(4-5) :441–504, 2012.

[75] Joseph A Konstan, Bradley N Miller, David Maltz, Jonathan L Herlocker, Lee R Gordon, and John Riedl. Grouplens : applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3) :77–87, 1997.

[76] Yehuda Koren. Factorization meets the neighborhood : a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[77] Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 447–456. ACM, 2009.

[78] Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 447–456. ACM, 2009.

[79] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8) :30–37, 2009.

[80] Matthieu Latapy, Tiphaine Viard, and Clémence Magnien. Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining*, 8(1) :61, 2018.

[81] Neal Lathia, Stephen Hailes, and Licia Capra. Trust-based collaborative filtering. In *IFIP international conference on trust management*, pages 119–134. Springer, 2008.

[82] Neal Lathia, Stephen Hailes, and Licia Capra. Temporal collaborative filtering with adaptive neighbourhoods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 796–797. ACM, 2009.

[83] Neal Kiritkumar Lathia. *Evaluating collaborative filtering over time.* PhD thesis, UCL (University College London), 2010.

[84] Dongjoo Lee, Sung Eun Park, Minsuk Kahng, Sangkeun Lee, and Sang-goo Lee. Exploiting contextual information from event logs for personalized recommendation. In *Computer and Information Science 2010*, pages 121–139. Springer, 2010.

[85] Sangkeun Lee, Sang-il Song, Minsuk Kahng, Dongjoo Lee, and Sang-goo Lee. Random walk based entity ranking on graph for multidimensional recommendation. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 93–100. ACM, 2011.

[86] Huayu Li, Yong Ge, Richang Hong, and Hengshu Zhu. Point-of-interest recommendations : Learning potential check-ins from friends. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 975–984. ACM, 2016.

[87] Lei Li, Li Zheng, Fan Yang, and Tao Li. Modeling and broadening temporal user interest in personalized news recommendation. *Expert Systems with Applications*, 41(7) :3168–3177, 2014.

[88] Lin Li, Zhenglu Yang, Botao Wang, and Masaru Kitsuregawa. Dynamic adaptation strategies for long-term and short-term user profile to personalize search. In *Advances in Data and Web Management*, pages 228–240. 2007.

[89] Xin Li and Hsinchun Chen. Recommendation as link prediction in bipartite graphs : A graph kernel-based machine learning approach. *Decision Support Systems*, 54(2) :880–890, 2013.

[90] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations : Item-to-item collaborative filtering. *IEEE Internet computing*, (1) :76–80, 2003.

[91] Paolo Massa and Paolo Avesani. Controversial users demand local trust metrics : An experimental study on epinions. com community. In *AAAI*, pages 121–126, 2005.

[92] Paolo Massa and Paolo Avesani. Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*, pages 17–24. ACM, 2007.

[93] Matthew R McLaughlin and Jonathan L Herlocker. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 329–336. ACM, 2004.

[94] Prem Melville, Raymond J Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. *Aaai/iaai*, 23 :187–192, 2002.

[95] Morteza Ghorbani Moghaddam, Norwati Mustapha, Aida Mustapha, Nurfadhlina Mohd Sharef, and Anousheh Elahian. Agetrust : A new temporal trust-based collaborative filtering approach. In *2014 International Conference on Information Science & Applications (ICISA)*, pages 1–4. IEEE, 2014.

[96] Raymond J Mooney and Loriene Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204. ACM, 2000.

[97] Subhabrata Mukherjee and Stephan Günnemann. Ghostlink : Latent network inference for influence-aware recommendation. *Work*, 1145 :3308558–3313449, 2019.

[98] Makoto Nakatsuji, Yasuhiro Fujiwara, Toshio Uchiyama, and Hiroyuki Toda. Collaborative filtering by analyzing dynamic user interests modeled by taxonomy. In *International Semantic Web Conference*, pages 361–377. Springer, 2012.

[99] Xia Ning and George Karypis. Slim : Sparse linear methods for top-n recommender systems. In *2011 11th IEEE International Conference on Data Mining*, pages 497–506. IEEE, 2011.

[100] Armel Jacques Nzekon Nzeko'o, Maurice Tchuente, and Matthieu Latapy. Time weight content-based extensions of temporal graphs for personalized recommendation. In *WEBIST 2017-13th International Conference on Web Information Systems and Technologies*, 2017.

[101] Armel Jacques Nzekon Nzeko'o, Maurice Tchuente, and Matthieu Latapy. A general graph-based framework for top-n recommendation using content, temporal and trust information. *Journal of Interdisciplinary Methodologies and Issues in Science. DOI 10.18713/JIMIS-300519-5-2*, Analysis of networks and graphs, 2019.

[102] Armel Jacques Nzekon Nzeko'o, Maurice Tchuente, and Matthieu Latapy. Link stream graph for temporal recommendations. *Colloquium of Mathematics and Computer Science, University of Dschang, Cameroon. arXiv preprint arXiv :1904.12576*, 2019.

[103] John O'Donovan and Barry Smyth. Trust in recommender systems. In *Proceedings of the 10th international conference on Intelligent user interfaces*, pages 167–174. ACM, 2005.

[104] Kenta Oku, Shinsuke Nakajima, Jun Miyazaki, and Shunsuke Uemura. Context-aware svm for context-dependent information recommendation. In *Proceedings of the 7th international Conference on Mobile Data Management*, page 109. IEEE Computer Society, 2006.

[105] Vito Claudio Ostuni, Tommaso Di Noia, Eugenio Di Sciascio, and Roberto Mirizzi. Top-n recommendations from implicit feedback leveraging linked open data. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 85–92. ACM, 2013.

[106] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking : Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[107] Cosimo Palmisano, Alexander Tuzhilin, and Michele Gorgoglione. Using context to improve predictive modeling of customers in personalization applications. *IEEE transactions on knowledge and data engineering*, 20(11) :1535–1549, 2008.

[108] Yiteng Pan, Fazhi He, and Haiping Yu. Trust-aware collaborative denoising auto-encoder for top-n recommendation. *arXiv preprint arXiv :1703.01760*, 2017.

[109] Umberto Panniello, Alexander Tuzhilin, Michele Gorgoglione, Cosimo Palmisano, and Anto Pedone. Experimental comparison of pre-vs. post-filtering approaches in context-aware recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, pages 265–268. ACM, 2009.

[110] Manos Papagelis, Dimitris Plexousakis, and Themistoklis Kutsuras. Alleviating the sparsity problem of collaborative filtering using trust inferences. In *Trust management*, pages 224–239. 2005.

[111] Michael Pazzani and Daniel Billsus. Learning and revising user profiles : The identification of interesting web sites. *Machine learning*, 27(3) :313–331, 1997.

[112] Michael J Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial intelligence review*, 13(5-6) :393–408, 1999.

[113] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. 2007.

[114] Nguyen Duy Phuong, Tu Minh Phuong, et al. A graph-based method for combining collaborative and content-based filtering. In *Pacific Rim International Conference on Artificial Intelligence*, pages 859–869. Springer, 2008.

[115] Bruno Pradel, Savaneary Sean, Julien Delporte, Sébastien Guérif, Céline Rouveirol, Nicolas Usunier, Françoise Fogelman-Soulié, and Frédéric Dufau-Joel. A case study in a recommender system based on purchase data. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 377–385. ACM, 2011.

[116] Aghny Arisya Putra, Rahmad Mahendra, Indra Budi, and Qorib Munajat. Two-steps graph-based collaborative filtering using user and item similarities : case study of e-commerce recommender systems. In *2017 International Conference on Data and Software Engineering (ICoDSE)*, pages 1–6. IEEE, 2017.

[117] Lianyong Qi, Ruili Wang, Chunhua Hu, Shancang Li, Qiang He, and Xiaolong Xu. Time-aware distributed service recommendation with privacy-preservation. *Information Sciences*, 480 :354–364, 2019.

[118] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr : Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 452–461. AUAI Press, 2009.

[119] Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda. A new approach for item ranking based on review scores reflecting temporal trust factor. In *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*, pages 161–168. Springer, 2014.

[120] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[121] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Incremental singular value decomposition algorithms for highly scalable recommender systems. In *Fifth international conference on computer and information science*, volume 27, page 28. Citeseer, 2002.

[122] Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260. ACM, 2002.

[123] Shang Shang, Sanjeev R Kulkarni, Paul W Cuff, and Pan Hui. A randomwalk based model incorporating social information for recommendations. In *2012 ieee international workshop on machine learning for signal processing*, pages 1–6. IEEE, 2012.

[124] Guy Shani and Asela Gunawardana. Evaluating recommendation systems. In *Recommender systems handbook*, pages 257–297. Springer, 2011.

[125] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, Nuria Oliver, and Alan Hanjalic. Climf : learning to maximize reciprocal rank with collaborative less-is-more filtering. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 139–146. ACM, 2012.

[126] Rashmi R Sinha, Kirsten Swearingen, et al. Comparing recommendations made by online systems and friends. In *DELOS*, 2001.

[127] Ian Soboroff and Charles Nicholas. Combining content and collaboration in text filtering. In *Proceedings of the IJCAI*, volume 99, pages 86–91. sn, 1999.

[128] Yang Song, Ali Mamdouh Elkahky, and Xiaodong He. Multi-rate deep learning for temporal recommendation. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 909–912. ACM, 2016.

[129] Harald Steck. Evaluation of recommendations : rating-prediction and ranking. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 213–220. ACM, 2013.

[130] J. Tang, H. Gao, and H. Liu. mTrust : Discerning multi-faceted trust in a connected world. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 93–102. ACM, 2012.

[131] Jiliang Tang, Huiji Gao, Xia Hu, and Huan Liu. Exploiting homophily effect for trust prediction. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 53–62. ACM, 2013.

[132] Jiliang Tang, Huiji Gao, and Huan Liu. mtrust : discerning multi-faceted trust in a connected world. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 93–102. ACM, 2012.

[133] Jiliang Tang, Xia Hu, and Huan Liu. Social recommendation : a review. *Social Network Analysis and Mining*, 3(4) :1113–1133, 2013.

[134] Saúl Vargas and Pablo Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 109–116. ACM, 2011.

[135] Jordan Viard and Matthieu Latapy. Identifying roles in an ip network with temporal and structural density. In *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 801–806. IEEE, 2014.

[136] Tiphaine Viard, Matthieu Latapy, and Clémence Magnien. Computing maximal cliques in link streams. *Theoretical Computer Science*, 609 :245–252, 2016.

[137] Patricia Victor, Chris Cornelis, Martine De Cock, and Ankur M Teredesai. Trust- and distrust-based recommendations for controversial reviews. *IEEE Intelligent Systems*, (1) :48–55, 2011.

[138] João Vinagre and Alípio Mário Jorge. Forgetting mechanisms for scalable collaborative filtering. *Journal of the Brazilian Computer Society*, 18(4) :271, 2012.

[139] Chong Wang and David M Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 448–456. ACM, 2011.

[140] Jun Wang, Arjen P De Vries, and Marcel JT Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 501–508. ACM, 2006.

[141] Chen Wei, Richard Khoury, and Simon Fong. Web 2.0 recommendation service by multi-collaborative filtering trust network algorithm. *Information Systems Frontiers*, 15(4) :533–551, 2013.

[142] Liang Xiang, Quan Yuan, Shiwan Zhao, Li Chen, Xiatian Zhang, Qing Yang, and Jimeng Sun. Temporal recommendation on graphs via long-and short-term preference fusion. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 723–732. ACM, 2010.

[143] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 211–222. SIAM, 2010.

[144] Guandong Xu, Yanchun Zhang, and Xun Yi. Modelling user behaviour for web recommendation using lda model. In *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 3, pages 529–532. IEEE, 2008.

[145] Rui Yan, Mirella Lapata, and Xiaoming Li. Tweet recommendation with graph co-ranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics : Long Papers-Volume 1*, pages 516–525. Association for Computational Linguistics, 2012.

[146] Diyi Yang, Tianqi Chen, Weinan Zhang, Qiuxia Lu, and Yong Yu. Local implicit feedback mining for music recommendation. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 91–98. ACM, 2012.

[147] Dawei Yin, Liangjie Hong, Zhenzhen Xue, and Brian D Davison. Temporal dynamics of user interests in tagging systems. In *Twenty-Fifth AAAI conference on artificial intelligence*, 2011.

[148] Jianjun Yu, Yi Shen, and Zhenglu Yang. Topic-stg : Extending the session-based temporal graph approach for personalized tweet recommendation. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 413–414. ACM, 2014.

[149] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. Time-aware point-of-interest recommendation. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 363–372. ACM, 2013.

[150] Zhijun Zhang and Hong Liu. Social recommendation model combining trust propagation and sequential behaviors. *Applied Intelligence*, 43(3) :695–706, 2015.

[151] Tong Zhao, Julian McAuley, and Irwin King. Leveraging social connections to improve personalized ranking for collaborative filtering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 261–270. ACM, 2014.

[152] Nan Zheng and Qiudan Li. A recommender system based on tag and time information for social tagging systems. *Expert Systems with Applications*, 38(4) :4575–4587, 2011.

[153] Tao Zhou, Zoltán Kuscsik, Jian-Guo Liu, Matúš Medo, Joseph Rushton Wakeling, and Yi-Cheng Zhang. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences*, 107(10) :4511–4515, 2010.

[154] Tao Zhou, Jie Ren, Matúš Medo, and Yi-Cheng Zhang. Bipartite network projection and personal recommendation. *Physical Review E*, 76(4) :046115, 2007.

[155] Yanzan Zhou, Xin Jin, and Bamshad Mobasher. A recommendation model based on latent principal factors in web navigation data. In *3rd International Workshop on Web Dynamics in conjunction with the 13th International World Wide Web Conference New York City, New York, USA, May 18, 2004*, page 52, 2004.

[156] Cai-Nicolas Ziegler and Jennifer Golbeck. Investigating correlations of trust and interest similarity-do birds of a feather really flock together. *Decision Support Systems*, 43(2) :1–34, 2005.

[157] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, pages 22–32. ACM, 2005.

[158] Andrew Zimdars, David Maxwell Chickering, and Christopher Meek. Using temporal data for making recommendations. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 580–588. Morgan Kaufmann Publishers Inc., 2001.

Chapitre A

Autres résultats obtenus avec GraFC2T2

A.1 Résultats de Epinions et Ciao pour d'autres valeurs du Top-N

Dans cette section, nous présentons les résultats obtenus pour le top-5, le top-20, top-50 et top-100. La section est divisée en deux parties : la première présente les performances obtenues avec Epinions et la seconde présente les performances obtenues avec Ciao. Dans chacune des sous-parties, on présente d'abord les résultats généraux, ensuite on présente les meilleurs résultats obtenus.

Les deux parties de la section A.1 des annexes, confirment les résultats obtenus dans le chapitre 7. Par exemple, les graphes de recommandation qui intègrent simultanément les informations sur le contenu des produits, la dynamique temporelle des préférences des utilisateurs et la relation de confiance entre les utilisateurs sont généralement les meilleurs. Nous recommandons donc l'intégration simultanée de ces trois types d'informations secondaires dans les systèmes de recommandation afin d'augmenter les chances d'atteindre de bonnes performances.

A.1.1 Cas de Epinions

Résultats généraux

TABLE A.1 – Epinions - Résultats obtenus en Top-5.

EPINIONS	F1@5			HR@5			MAP@5		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.55	1.96	1.32	3.32	3.71	2.52	2.0	2.11	1.5
ET	2.2	1.8	1.25	3.32	3.45	2.39	1.89	1.93	1.45
IT	2.31	2.83	1.9	3.18	3.71	3.32	2.16	2.18	2.0
EDF	3.39	3.85	1.72	4.24	4.11	3.58	2.09	2.09	2.05
LDF	2.9	2.58	2.79	4.24	4.11	4.51	2.0	1.96	2.34
CI	3.35	3.67	0.81	3.71	4.11	1.86	2.2	2.56	1.55
CIU	4.78	5.01	0.75	4.24	4.11	1.72	2.46	2.65	1.53
EDF-ET	2.97	1.99	2.29	3.85	3.18	3.05	1.93	1.71	1.84
EDF-IT	4.8	4.1	2.46	4.77	4.51	3.98	2.36	2.32	2.12
LDF-ET	3.2	2.26	2.05	3.32	3.32	3.58	1.85	1.86	1.97
LDF-IT	5.27	5.14	4.61	4.64	4.51	5.17	2.74	2.74	2.89
CI-ET	3.05	3.53	0.83	3.45	3.71	1.86	1.99	2.36	1.55
CI-IT	3.38	4.14	1.48	3.85	3.85	3.58	2.17	2.36	1.95
CIU-ET	5.37	5.01	0.76	4.24	4.11	1.72	2.26	2.65	1.53
CIU-IT	4.25	5.12	1.63	3.98	4.11	3.58	2.52	2.65	2.02
CI-EDF	7.16	4.72	0.81	4.64	4.11	1.86	2.7	2.47	1.55
CI-LDF	6.67	5.89	1.12	4.51	3.98	2.25	2.3	2.36	1.56
CIU-EDF	20.0	7.96	0.8	5.04	4.91	1.86	2.9	2.75	1.53
CIU-LDF	11.3	7.51	1.02	5.04	4.91	2.65	2.7	2.67	1.59
CI-EDF-ET	5.08	3.17	0.83	4.11	3.58	1.86	2.19	2.11	1.55
CI-EDF-IT	7.96	5.83	1.41	4.91	4.38	3.45	2.6	2.47	1.98
CI-LDF-ET	3.9	3.47	0.83	3.85	3.71	1.86	1.97	2.31	1.55
CI-LDF-IT	8.29	5.72	3.6	5.17	4.64	4.24	2.78	2.78	2.22
CIU-EDF-ET	7.94	7.96	0.8	4.64	4.77	1.86	2.41	2.6	1.53
CIU-EDF-IT	19.1	7.96	1.84	5.31	4.91	3.58	3.04	2.85	2.04
CIU-LDF-ET	6.4	5.71	0.9	4.51	4.51	2.12	2.3	2.47	1.53
CIU-LDF-IT	8.09	6.34	3.73	4.91	4.77	4.64	2.82	2.83	2.84

TABLE A.2 – Epinions - Résultats obtenus en Top-20.

EPINIONS	F1@20			HR@20			MAP@20		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.56	1.59	1.11	8.22	8.22	7.16	2.38	2.4	1.84
ET	1.31	1.49	1.05	7.43	7.96	6.9	2.18	2.24	1.8
IT	1.73	1.77	1.39	8.75	9.02	8.89	2.51	2.49	2.41
EDF	2.7	2.15	2.55	10.6	9.55	10.6	2.62	2.56	2.61
LDF	1.88	1.68	1.53	8.89	8.49	8.49	2.38	2.23	2.67
CI	1.95	2.15	0.76	9.28	9.68	5.44	2.64	2.98	1.89
CIU	2.16	2.36	0.63	9.95	10.1	4.64	2.9	3.13	1.73
EDF-ET	1.8	1.68	2.46	8.75	8.62	10.3	2.36	2.15	2.41
EDF-IT	2.47	2.5	2.43	9.95	10.1	10.3	2.84	2.67	2.64
LDF-ET	1.39	1.57	1.21	7.69	8.22	7.56	2.17	2.24	2.32
LDF-IT	1.92	1.95	2.18	9.55	9.81	10.5	3.12	3.12	3.06
CI-ET	1.67	2.12	0.79	8.49	9.55	5.57	2.4	2.83	1.9
CI-IT	2.05	2.19	1.49	9.95	10.1	8.75	2.63	2.8	2.35
CIU-ET	2.07	2.12	0.64	9.68	9.95	4.64	2.67	3.13	1.73
CIU-IT	2.19	2.35	1.53	10.1	10.1	9.15	2.9	3.13	2.43
CI-EDF	3.25	2.88	0.81	11.7	11.1	5.7	3.33	2.97	1.9
CI-LDF	2.57	2.38	0.85	10.6	9.68	5.97	2.76	2.83	1.9
CIU-EDF	3.45	2.93	0.67	11.8	11.3	5.04	3.45	3.26	1.79
CIU-LDF	3.02	2.39	0.74	11.0	10.6	5.31	3.01	3.09	1.82
CI-EDF-ET	2.6	2.58	0.81	10.5	10.7	5.7	2.7	2.71	1.9
CI-EDF-IT	3.25	3.26	1.52	11.3	11.4	8.89	3.03	2.99	2.4
CI-LDF-ET	1.98	2.17	0.86	9.15	9.68	5.97	2.42	2.83	1.9
CI-LDF-IT	2.63	2.3	2.09	10.5	10.2	9.81	3.15	3.14	2.77
CIU-EDF-ET	2.92	2.85	0.67	11.0	10.9	5.04	2.87	3.11	1.79
CIU-EDF-IT	3.44	3.33	1.48	11.7	11.5	9.02	3.55	3.41	2.46
CIU-LDF-ET	2.43	2.34	0.67	10.1	10.5	5.04	2.73	3.01	1.74
CIU-LDF-IT	2.81	2.51	1.93	10.7	10.5	9.42	3.29	3.24	3.26

TABLE A.3 – Epinions - Résultats obtenus en Top-50.

EPINIONS	F1@50			HR@50			MAP@50		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.06	1.08	0.91	15.3	15.4	13.9	2.5	2.55	2.03
ET	0.95	1.05	0.87	14.2	15.1	13.5	2.3	2.4	1.99
IT	1.16	1.16	0.97	16.0	16.4	14.7	2.52	2.63	2.55
EDF	1.25	1.28	1.04	17.1	16.4	15.4	2.64	2.71	2.7
LDF	1.14	1.15	0.99	16.0	16.0	14.7	2.5	2.4	2.72
CI	1.55	1.48	0.56	19.0	18.4	9.68	2.87	3.15	1.93
CIU	1.59	1.53	0.48	19.0	18.6	8.62	3.01	3.26	1.83
EDF-ET	1.09	1.11	0.89	15.5	15.5	14.2	2.43	2.33	2.52
EDF-IT	1.36	1.34	0.97	17.5	16.8	15.0	2.83	2.74	2.73
LDF-ET	0.97	1.08	0.87	14.6	15.4	13.5	2.28	2.4	2.38
LDF-IT	1.18	1.16	0.98	16.6	16.4	14.9	3.09	3.09	3.07
CI-ET	1.43	1.47	0.56	18.2	18.3	9.68	2.63	3.05	1.93
CI-IT	1.49	1.47	0.89	18.4	18.4	14.2	2.77	3.03	2.41
CIU-ET	1.49	1.49	0.48	18.3	18.6	8.49	2.76	3.27	1.83
CIU-IT	1.51	1.55	0.91	18.6	18.6	14.6	3.07	3.29	2.5
CI-EDF	1.74	1.66	0.58	19.9	19.9	9.95	3.29	3.06	1.94
CI-LDF	1.67	1.51	0.6	19.6	18.7	10.1	2.89	3.05	1.95
CIU-EDF	1.63	1.77	0.55	19.4	19.9	9.81	3.5	3.36	1.84
CIU-LDF	1.59	1.55	0.54	19.0	19.0	9.95	3.2	3.33	1.92
CI-EDF-ET	1.72	1.63	0.59	19.8	19.8	9.95	2.71	2.93	1.94
CI-EDF-IT	1.52	1.66	0.88	18.8	19.9	14.2	3.1	2.99	2.46
CI-LDF-ET	1.56	1.52	0.56	19.0	18.8	9.68	2.65	3.06	1.93
CI-LDF-IT	1.49	1.49	0.93	18.6	18.6	14.5	3.14	3.14	2.82
CIU-EDF-ET	1.52	1.75	0.55	18.7	19.9	9.81	3.02	3.29	1.83
CIU-EDF-IT	1.62	1.73	0.91	19.6	19.8	14.6	3.65	3.48	2.54
CIU-LDF-ET	1.47	1.55	0.51	18.2	19.0	9.42	2.83	3.18	1.84
CIU-LDF-IT	1.52	1.53	0.97	18.7	18.8	15.1	3.36	3.45	3.27

TABLE A.4 – Epinions - Résultats obtenus en Top-100.

EPINIONS	F1@100			HR@100			MAP@100		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	0.7	0.72	0.67	22.5	22.4	21.9	2.54	2.59	2.13
ET	0.69	0.68	0.66	22.3	22.0	21.5	2.35	2.43	2.09
IT	0.74	0.77	0.66	23.3	23.6	21.9	2.6	2.68	2.61
EDF	0.77	0.77	0.67	22.9	23.3	21.9	2.66	2.66	2.72
LDF	0.71	0.72	0.67	22.5	22.3	21.9	2.54	2.42	2.65
CI	0.89	0.91	0.39	26.3	26.3	14.3	2.84	3.13	1.83
CIU	0.87	0.92	0.39	26.0	27.5	14.3	3.03	3.35	1.82
EDF-ET	0.71	0.7	0.65	22.1	22.0	21.2	2.44	2.33	2.54
EDF-IT	0.76	0.79	0.67	23.5	23.9	22.3	2.9	2.77	2.77
LDF-ET	0.71	0.69	0.66	22.4	21.9	21.5	2.33	2.42	2.36
LDF-IT	0.75	0.79	0.66	23.6	23.9	21.9	3.1	3.1	3.02
CI-ET	0.87	0.88	0.39	25.9	25.7	14.3	2.6	2.96	1.83
CI-IT	0.88	0.88	0.58	26.1	26.5	20.0	2.78	2.93	2.48
CIU-ET	0.86	0.92	0.39	25.9	27.5	14.3	2.85	3.36	1.82
CIU-IT	0.89	0.92	0.59	26.5	27.3	20.2	3.15	3.35	2.52
CI-EDF	0.92	0.98	0.42	26.8	27.6	15.1	3.29	3.15	1.85
CI-LDF	0.89	0.88	0.42	26.3	26.0	15.1	2.88	2.95	1.89
CIU-EDF	0.91	1.03	0.4	26.3	28.5	14.5	3.58	3.34	1.83
CIU-LDF	0.89	0.94	0.41	26.3	27.6	15.0	3.12	3.33	1.9
CI-EDF-ET	0.91	0.95	0.42	26.5	27.2	15.0	2.71	2.89	1.84
CI-EDF-IT	0.92	0.95	0.58	26.7	27.2	20.0	3.2	3.06	2.49
CI-LDF-ET	0.88	0.89	0.41	25.7	25.7	14.6	2.61	2.95	1.83
CI-LDF-IT	0.88	0.88	0.57	26.1	26.5	20.0	3.16	3.16	2.82
CIU-EDF-ET	0.91	1.02	0.39	26.1	28.5	14.3	2.94	3.21	1.82
CIU-EDF-IT	0.91	1.02	0.59	27.2	28.5	20.2	3.73	3.44	2.55
CIU-LDF-ET	0.88	0.94	0.39	26.0	27.6	14.3	2.79	3.17	1.82
CIU-LDF-IT	0.89	0.92	0.59	26.5	27.3	20.3	3.37	3.43	3.26

Meilleures performances

TABLE A.5 – Epinions - Meilleures performances en Top-5, -20, -50 et -100

Epinions Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@5	1	1.55	20.0	1192%	CIU-EDF	1.96	7.96	305%	CIU-EDF	1.32	4.61	247%	LDF-IT
	2	1.55	19.1	1136%	CIU-EDF-IT	1.96	7.96	305%	CIU-EDF-ET	1.32	3.73	181%	CIU-LDF-IT
	3	1.55	11.3	632%	CIU-LDF	1.96	7.96	305%	CIU-EDF-IT	1.32	3.6	172%	CI-LDF-IT
H@5	1	3.32	5.31	59%	CIU-EDF-IT	3.71	4.91	32%	CIU-EDF	2.52	5.17	105%	LDF-IT
	2	3.32	5.17	55%	CI-LDF-IT	3.71	4.91	32%	CIU-LDF	2.52	4.64	84%	CIU-LDF-IT
	3	3.32	5.04	51%	CIU-EDF	3.71	4.91	32%	CIU-EDF-IT	2.52	4.51	78%	LDF
M@5	1	2.0	3.04	52%	CIU-EDF-IT	2.11	2.85	34%	CIU-EDF-IT	1.5	2.89	92%	LDF-IT
	2	2.0	2.9	45%	CIU-EDF	2.11	2.83	33%	CIU-LDF-IT	1.5	2.84	89%	CIU-LDF-IT
	3	2.0	2.82	41%	CIU-LDF-IT	2.11	2.78	31%	CI-LDF-IT	1.5	2.34	55%	LDF

Epinions Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@20	1	1.56	3.45	121%	CIU-EDF	1.59	3.33	109%	CIU-EDF-IT	1.11	2.55	129%	EDF
	2	1.56	3.44	120%	CIU-EDF-IT	1.59	3.26	105%	CI-EDF-IT	1.11	2.46	121%	EDF-ET
	3	1.56	3.25	109%	CI-EDF-IT	1.59	2.93	84%	CIU-EDF	1.11	2.43	118%	EDF-IT
H@20	1	8.22	11.8	43%	CIU-EDF	8.22	11.5	40%	CIU-EDF-IT	7.16	10.6	48%	EDF
	2	8.22	11.7	41%	CI-EDF	8.22	11.4	38%	CI-EDF-IT	7.16	10.5	46%	LDF-IT
	3	8.22	11.7	41%	CIU-EDF-IT	8.22	11.3	37%	CIU-EDF	7.16	10.3	44%	EDF-ET
M@20	1	2.38	3.55	49%	CIU-EDF-IT	2.4	3.41	41%	CIU-EDF-IT	1.84	3.26	77%	CIU-LDF-IT
	2	2.38	3.45	45%	CIU-EDF	2.4	3.26	35%	CIU-EDF	1.84	3.06	66%	LDF-IT
	3	2.38	3.33	40%	CI-EDF	2.4	3.24	34%	CIU-LDF-IT	1.84	2.77	50%	CI-LDF-IT

Epinions Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@50	1	1.06	1.74	64%	CI-EDF	1.08	1.77	62%	CIU-EDF	0.91	1.04	14%	EDF
	2	1.06	1.72	61%	CI-EDF-ET	1.08	1.75	61%	CIU-EDF-ET	0.91	0.99	9%	LDF
	3	1.06	1.67	57%	CI-LDF	1.08	1.73	59%	CIU-EDF-IT	0.91	0.98	7%	LDF-IT
H@50	1	15.3	19.9	30%	CI-EDF	15.4	19.9	29%	CI-EDF	13.9	15.4	10%	EDF
	2	15.3	19.8	29%	CI-EDF-ET	15.4	19.9	29%	CIU-EDF	13.9	15.1	8%	CIU-LDF-IT
	3	15.3	19.6	28%	CI-LDF	15.4	19.9	29%	CI-EDF-IT	13.9	15.0	7%	EDF-IT
M@50	1	2.5	3.65	45%	CIU-EDF-IT	2.55	3.48	36%	CIU-EDF-IT	2.03	3.27	61%	CIU-LDF-IT
	2	2.5	3.5	40%	CIU-EDF	2.55	3.45	35%	CIU-LDF-IT	2.03	3.07	51%	LDF-IT
	3	2.5	3.36	34%	CIU-LDF-IT	2.55	3.36	31%	CIU-EDF	2.03	2.82	38%	CI-LDF-IT

Epinions Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@100	1	0.7	0.92	30%	CI-EDF-IT	0.72	1.03	44%	CIU-EDF	0.67	0.67	0%	-
	2	0.7	0.92	30%	CI-EDF	0.72	1.02	42%	CIU-EDF-ET	0.67	0.67	0%	LDF
	3	0.7	0.91	29%	CIU-EDF-IT	0.72	1.02	42%	CIU-EDF-IT	0.67	0.67	0%	EDF-IT
H@100	1	22.5	27.2	20%	CIU-EDF-IT	22.4	28.5	27%	CIU-EDF	21.9	22.3	1%	EDF-IT
	2	22.5	26.8	18%	CI-EDF	22.4	28.5	27%	CIU-EDF-ET	21.9	21.9	0%	-
	3	22.5	26.7	18%	CI-EDF-IT	22.4	28.5	27%	CIU-EDF-IT	21.9	21.9	0%	IT
M@100	1	2.54	3.73	46%	CIU-EDF-IT	2.59	3.44	32%	CIU-EDF-IT	2.13	3.26	53%	CIU-LDF-IT
	2	2.54	3.58	41%	CIU-EDF	2.59	3.43	32%	CIU-LDF-IT	2.13	3.02	41%	LDF-IT
	3	2.54	3.37	33%	CIU-LDF-IT	2.59	3.36	29%	CIU-ET	2.13	2.82	32%	CI-LDF-IT

A.1.2 Cas de Ciao

Résultats généraux

TABLE A.6 – Ciao - Résultats obtenus en Top-5.

CIAO	F1@5			HR@5			MAP@5		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.44	1.86	1.64	3.27	3.99	4.17	1.65	1.8	1.91
ET	1.09	1.86	1.18	2.72	3.81	3.63	1.42	1.69	1.74
IT	1.83	1.86	1.6	4.9	4.17	3.99	2.04	1.94	1.97
EDF	1.86	2.33	1.82	4.36	4.17	4.54	1.72	2.05	2.13
LDF	1.75	1.79	3.05	4.72	4.72	5.44	2.34	2.55	3.03
CI	1.52	1.8	1.05	3.27	4.17	2.72	1.63	1.93	1.38
CIU	1.85	2.91	1.23	3.81	4.9	3.09	1.96	2.61	1.28
EDF-ET	0.92	1.4	1.19	3.63	3.81	3.63	1.48	1.83	1.76
EDF-IT	3.47	2.31	1.6	5.44	4.9	3.99	2.52	2.36	1.99
LDF-ET	1.18	1.36	1.18	2.9	3.99	3.63	1.46	1.8	1.74
LDF-IT	2.83	2.46	2.41	5.81	5.44	4.9	2.72	2.66	2.4
CI-ET	0.91	1.7	1.07	3.27	3.81	2.72	1.61	1.88	1.38
CI-IT	1.61	1.7	2.05	4.36	4.17	4.72	2.07	1.94	2.0
CIU-ET	1.05	3.12	1.23	3.63	4.54	3.09	1.82	2.62	1.28
CIU-IT	2.06	2.85	2.22	5.08	4.72	4.54	2.4	2.46	1.9
CI-EDF	2.16	2.32	1.3	4.54	4.54	3.27	1.97	2.32	1.46
CI-LDF	2.99	2.76	2.32	5.26	5.08	4.54	2.43	2.51	2.53
CIU-EDF	4.34	5.0	1.23	5.63	5.44	3.09	2.86	2.9	1.38
CIU-LDF	3.85	4.4	2.33	5.99	5.99	4.54	3.07	2.96	2.51
CI-EDF-ET	1.91	2.5	1.28	4.17	4.36	3.09	1.97	2.07	1.46
CI-EDF-IT	3.31	4.09	1.99	5.44	5.63	4.72	2.59	2.66	2.08
CI-LDF-ET	1.02	1.7	1.38	3.27	3.81	3.27	1.68	1.96	1.49
CI-LDF-IT	5.51	3.53	2.1	5.99	5.99	4.9	2.88	2.91	2.69
CIU-EDF-ET	2.33	3.05	1.23	5.26	5.08	3.09	2.34	2.52	1.38
CIU-EDF-IT	3.54	4.68	2.22	5.44	5.81	4.36	2.68	2.75	1.81
CIU-LDF-ET	1.2	5.63	1.23	3.45	5.26	3.09	1.78	2.77	1.37
CIU-LDF-IT	5.81	5.52	2.22	5.99	5.99	5.26	2.93	2.99	2.76

TABLE A.7 – Ciao - Résultats obtenus en Top-20.

CIAO	F1@20			HR@20			MAP@20		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.61	1.93	1.71	9.26	9.98	10.2	2.13	2.25	2.44
ET	1.36	1.69	1.61	8.71	9.26	9.98	1.96	2.13	2.35
IT	3.08	2.72	2.2	13.1	12.9	11.8	2.76	2.75	2.51
EDF	2.24	2.35	2.51	11.1	11.4	12.3	2.28	2.56	2.65
LDF	2.05	1.99	2.7	10.2	10.2	13.2	2.74	3.0	3.78
CI	3.13	3.58	1.03	12.2	12.3	7.44	2.37	2.45	1.75
CIU	2.93	4.39	0.87	13.2	14.5	6.72	2.8	3.43	1.55
EDF-ET	2.01	2.01	1.69	10.7	10.3	11.4	2.0	2.36	2.41
EDF-IT	3.15	2.69	2.34	14.2	13.8	12.0	3.27	3.11	2.56
LDF-ET	1.44	1.69	1.61	8.71	9.26	9.98	2.02	2.22	2.36
LDF-IT	2.82	2.82	2.6	12.3	12.5	12.3	3.06	3.25	2.81
CI-ET	2.59	3.25	1.09	11.6	12.2	7.8	2.3	2.39	1.88
CI-IT	3.35	3.47	2.76	13.4	14.0	12.7	2.87	2.83	2.68
CIU-ET	3.78	4.39	1.03	14.0	14.5	7.26	2.73	3.51	1.63
CIU-IT	3.64	4.71	3.27	14.5	14.7	13.4	3.13	3.39	2.66
CI-EDF	4.81	3.95	1.48	15.1	13.8	9.26	2.77	3.01	1.84
CI-LDF	4.27	3.92	1.54	14.9	14.2	10.3	3.29	3.21	2.92
CIU-EDF	4.37	4.19	1.32	15.8	14.7	8.17	3.66	3.6	1.84
CIU-LDF	3.43	4.55	1.39	15.2	15.4	9.62	3.62	3.68	2.82
CI-EDF-ET	2.19	3.78	1.47	13.8	13.6	9.07	2.76	2.78	1.93
CI-EDF-IT	4.18	3.86	2.95	15.4	14.9	13.1	3.41	3.42	2.76
CI-LDF-ET	2.7	3.33	1.24	11.6	13.1	8.71	2.31	2.44	2.01
CI-LDF-IT	4.87	4.92	3.38	15.1	14.5	13.8	3.44	3.48	3.45
CIU-EDF-ET	3.7	4.38	1.32	15.8	14.5	8.17	3.24	3.42	1.84
CIU-EDF-IT	3.84	4.56	3.39	16.0	15.8	13.6	3.63	3.64	2.69
CIU-LDF-ET	4.2	4.5	1.21	14.5	15.4	8.35	2.79	3.69	1.69
CIU-LDF-IT	5.06	4.36	3.28	16.0	15.8	14.5	3.59	3.69	3.49

TABLE A.8 – Ciao - Résultats obtenus en Top-50.

CIAO	F1@50			HR@50			MAP@50		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.46	1.55	1.51	19.4	19.6	19.8	2.34	2.5	2.74
ET	1.58	1.52	1.47	19.6	19.2	20.0	2.2	2.43	2.67
IT	1.69	1.72	1.62	20.9	21.1	20.5	2.92	2.84	2.73
EDF	1.74	1.74	1.81	21.4	20.7	20.9	2.56	2.78	2.86
LDF	1.47	1.62	1.77	19.4	20.3	20.9	2.91	3.0	3.97
CI	2.27	2.18	0.97	22.1	22.5	15.2	2.55	2.71	1.92
CIU	2.37	2.65	0.88	24.0	24.9	14.3	3.03	3.67	1.72
EDF-ET	1.63	1.59	1.59	20.3	20.9	20.7	2.24	2.65	2.73
EDF-IT	1.98	1.79	1.7	22.1	21.8	21.2	3.39	3.23	2.81
LDF-ET	1.59	1.58	1.47	19.6	19.6	20.0	2.25	2.54	2.68
LDF-IT	1.9	1.69	1.65	21.8	20.5	20.5	3.18	3.29	2.99
CI-ET	2.34	2.07	0.98	23.0	23.0	15.4	2.51	2.68	1.92
CI-IT	2.53	2.29	1.57	23.6	22.7	20.5	2.98	2.91	2.76
CIU-ET	2.27	2.66	0.88	24.0	24.7	14.2	2.98	3.75	1.74
CIU-IT	2.52	2.86	1.58	24.7	25.8	21.2	3.26	3.62	2.88
CI-EDF	2.45	2.36	1.26	23.0	24.0	18.9	2.98	3.18	2.02
CI-LDF	2.56	2.23	1.17	23.4	22.9	17.6	3.45	3.29	3.12
CIU-EDF	2.57	2.71	1.03	24.0	25.4	16.5	3.87	3.87	2.07
CIU-LDF	2.62	2.78	1.13	24.0	24.7	16.9	3.74	3.82	3.01
CI-EDF-ET	2.63	2.34	1.29	24.1	23.8	18.9	3.01	3.04	2.05
CI-EDF-IT	2.6	2.44	1.59	24.5	24.1	20.7	3.53	3.56	2.94
CI-LDF-ET	2.31	2.21	0.99	23.4	23.8	15.4	2.63	2.75	2.05
CI-LDF-IT	2.51	2.42	1.64	23.8	23.8	21.1	3.61	3.56	3.61
CIU-EDF-ET	2.69	2.69	1.07	24.3	25.8	16.9	3.42	3.7	2.07
CIU-EDF-IT	2.82	2.69	1.61	25.4	25.4	21.4	3.69	3.73	2.91
CIU-LDF-ET	2.55	2.72	1.08	23.8	25.2	16.7	3.0	3.86	1.93
CIU-LDF-IT	2.99	2.88	1.61	24.9	24.9	20.9	3.69	3.76	3.51

TABLE A.9 – Ciao - Résultats obtenus en Top-100.

CIAO	F1@100			HR@100			MAP@100		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.11	1.15	1.06	27.9	28.3	27.8	2.39	2.49	2.64
ET	1.08	1.15	1.03	27.9	28.1	27.4	2.25	2.36	2.56
IT	1.16	1.15	1.07	28.7	28.7	27.9	2.88	2.85	2.73
EDF	1.25	1.27	1.18	30.5	31.0	29.9	2.46	2.64	2.86
LDF	1.23	1.14	1.17	29.6	28.5	29.2	2.96	3.05	3.94
CI	1.54	1.51	0.96	33.8	33.4	26.7	2.62	2.74	2.02
CIU	1.58	1.57	0.89	34.3	34.5	25.8	3.08	3.51	1.72
EDF-ET	1.21	1.27	1.11	29.2	30.3	28.3	2.18	2.57	2.62
EDF-IT	1.22	1.26	1.09	28.9	30.1	27.9	3.37	3.22	2.78
LDF-ET	1.11	1.15	1.03	28.7	28.7	27.6	2.32	2.47	2.57
LDF-IT	1.14	1.18	1.07	27.9	29.2	27.9	3.22	3.31	3.02
CI-ET	1.57	1.52	0.98	33.9	33.4	26.5	2.58	2.65	2.04
CI-IT	1.52	1.53	1.09	33.2	33.4	29.9	2.97	2.91	2.83
CIU-ET	1.57	1.56	0.9	34.8	35.2	25.8	3.03	3.6	1.8
CIU-IT	1.48	1.5	1.09	33.4	34.1	29.2	3.23	3.47	2.85
CI-EDF	1.66	1.56	1.01	34.5	33.6	28.5	2.92	3.13	2.14
CI-LDF	1.6	1.49	1.0	34.5	33.6	27.2	3.37	3.32	3.18
CIU-EDF	1.73	1.74	0.9	35.0	35.9	26.5	3.83	3.76	2.17
CIU-LDF	1.56	1.5	0.89	34.5	34.3	26.0	3.7	3.81	2.98
CI-EDF-ET	1.71	1.54	1.01	34.5	33.6	29.2	2.91	2.98	2.12
CI-EDF-IT	1.57	1.54	1.16	33.0	33.4	29.8	3.41	3.55	2.85
CI-LDF-ET	1.55	1.55	0.98	33.9	33.4	26.5	2.63	2.71	2.16
CI-LDF-IT	1.49	1.5	1.15	32.7	33.8	30.5	3.58	3.59	3.5
CIU-EDF-ET	1.84	1.76	0.91	35.8	35.9	26.9	3.33	3.66	2.17
CIU-EDF-IT	1.74	1.73	1.1	34.7	35.8	29.6	3.64	3.68	2.89
CIU-LDF-ET	1.59	1.57	0.9	34.7	34.7	25.8	3.04	3.72	1.98
CIU-LDF-IT	1.52	1.51	1.13	33.2	34.5	30.5	3.75	3.79	3.46

Meilleures performances

TABLE A.10 – Ciao - Meilleures performances en Top-5, -20, -50 et -100

Ciao Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@5	1	1.44	5.81	304%	CIU-LDF-IT	1.86	5.63	203%	CIU-LDF-ET	1.64	3.05	86%	LDF
	2	1.44	5.51	284%	CI-LDF-IT	1.86	5.52	196%	CIU-LDF-IT	1.64	2.41	47%	LDF-IT
	3	1.44	4.34	202%	CIU-EDF	1.86	5.0	169%	CIU-EDF	1.64	2.33	42%	CIU-LDF
H@5	1	3.27	5.99	83%	CIU-LDF	3.99	5.99	50%	CIU-LDF	4.17	5.44	30%	LDF
	2	3.27	5.99	83%	CI-LDF-IT	3.99	5.99	50%	CI-LDF-IT	4.17	5.26	26%	CIU-LDF-IT
	3	3.27	5.99	83%	CIU-LDF-IT	3.99	5.99	50%	CIU-LDF-IT	4.17	4.9	17%	LDF-IT
M@5	1	1.65	3.07	85%	CIU-LDF	1.8	2.99	66%	CIU-LDF-IT	1.91	3.03	58%	LDF
	2	1.65	2.93	76%	CIU-LDF-IT	1.8	2.96	64%	CIU-LDF	1.91	2.76	43%	CIU-LDF-IT
	3	1.65	2.88	74%	CI-LDF-IT	1.8	2.91	61%	CI-LDF-IT	1.91	2.69	40%	CI-LDF-IT

Ciao Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@20	1	1.61	5.06	215%	CIU-LDF-IT	1.93	4.92	155%	CI-LDF-IT	1.71	3.39	98%	CIU-EDF-IT
	2	1.61	4.87	202%	CI-LDF-IT	1.93	4.71	144%	CIU-IT	1.71	3.38	97%	CI-LDF-IT
	3	1.61	4.81	199%	CI-EDF	1.93	4.56	136%	CIU-EDF-IT	1.71	3.28	91%	CIU-LDF-IT
H@20	1	9.26	16.0	72%	CIU-EDF-IT	9.98	15.8	58%	CIU-EDF-IT	10.2	14.5	42%	CIU-LDF-IT
	2	9.26	16.0	72%	CIU-LDF-IT	9.98	15.8	58%	CIU-LDF-IT	10.2	13.8	35%	CI-LDF-IT
	3	9.26	15.8	70%	CIU-EDF	9.98	15.4	54%	CIU-LDF	10.2	13.6	33%	CIU-EDF-IT
M@20	1	2.13	3.66	72%	CIU-EDF	2.25	3.69	64%	CIU-LDF-ET	2.44	3.78	54%	LDF
	2	2.13	3.63	70%	CIU-EDF-IT	2.25	3.69	64%	CIU-LDF-IT	2.44	3.49	42%	CIU-LDF-IT
	3	2.13	3.62	70%	CIU-LDF	2.25	3.68	63%	CIU-LDF	2.44	3.45	41%	CI-LDF-IT

Ciao Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@50	1	1.46	2.99	104%	CIU-LDF-IT	1.55	2.88	85%	CIU-LDF-IT	1.51	1.81	20%	EDF
	2	1.46	2.82	93%	CIU-EDF-IT	1.55	2.86	84%	CIU-IT	1.51	1.77	17%	LDF
	3	1.46	2.69	83%	CIU-EDF-ET	1.55	2.78	79%	CIU-LDF	1.51	1.7	12%	EDF-IT
H@50	1	19.4	25.4	30%	CIU-EDF-IT	19.6	25.8	31%	CIU-IT	19.8	21.4	8%	CIU-EDF-IT
	2	19.4	24.9	28%	CIU-LDF-IT	19.6	25.8	31%	CIU-EDF-ET	19.8	21.2	7%	EDF-IT
	3	19.4	24.7	27%	CIU-IT	19.6	25.4	29%	CIU-EDF	19.8	21.2	7%	CIU-IT
M@50	1	2.34	3.87	65%	CIU-EDF	2.5	3.87	54%	CIU-EDF	2.74	3.97	44%	LDF
	2	2.34	3.74	60%	CIU-LDF	2.5	3.86	54%	CIU-LDF-ET	2.74	3.61	31%	CI-LDF-IT
	3	2.34	3.69	57%	CIU-LDF-IT	2.5	3.82	52%	CIU-LDF	2.74	3.51	28%	CIU-LDF-IT

Ciao Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@100	1	1.11	1.84	66%	CIU-EDF-ET	1.15	1.76	52%	CIU-EDF-ET	1.06	1.18	11%	EDF
	2	1.11	1.74	57%	CIU-EDF-IT	1.15	1.74	51%	CIU-EDF	1.06	1.17	10%	LDF
	3	1.11	1.73	56%	CIU-EDF	1.15	1.73	50%	CIU-EDF-IT	1.06	1.16	9%	CI-EDF-IT
H@100	1	27.9	35.8	27%	CIU-EDF-ET	28.3	35.9	26%	CIU-EDF	27.8	30.5	9%	CI-LDF-IT
	2	27.9	35.0	25%	CIU-EDF	28.3	35.9	26%	CIU-EDF-ET	27.8	30.5	9%	CIU-LDF-IT
	3	27.9	34.8	24%	CIU-ET	28.3	35.8	26%	CIU-EDF-IT	27.8	29.9	7%	EDF
M@100	1	2.39	3.83	60%	CIU-EDF	2.49	3.81	52%	CIU-LDF	2.64	3.94	49%	LDF
	2	2.39	3.75	57%	CIU-LDF-IT	2.49	3.79	52%	CIU-LDF-IT	2.64	3.5	32%	CI-LDF-IT
	3	2.39	3.7	55%	CIU-LDF	2.49	3.76	51%	CIU-EDF	2.64	3.46	31%	CIU-LDF-IT

A.2 Meilleures performances dans les autres jeux de données

Dans cette section, nous présentons les résultats obtenus avec les jeux de données CiteUlike, Delicious, Last.fm et Ponpare. Notons que nous ne disposons pas de données sur la confiance explicite dans ces jeux de données. Les statistiques sur les données utilisées sont présentées dans le tableau A.11.

TABLE A.11 – Statistiques sur les jeux de données de CiteUlike, Delicious, Last.fm et Ponpare utilisés en annexe.

	CiteUlike	**Delicious**	**Last.fm**	**Ponpare**
D. début	2010-01-01	2010-05-10	2005-02-14	2011-07-01
D. fin	2010-04-01	2010-09-10	2005-05-14	2011-09-01
|U|	2 158	801	87	1 508
|I|	1 824	2 198	3 796	2 449
|C|	7 233	5 818	700	153
Nb. (u,i)	10 858	5 275	15 504	11 353
Densité	0.28%	0.30%	4.69%	0.31%
|L|	28 115	24 453	39 912	152 714

Commentaires sur les résultats En observant les performances obtenus dans les jeux de données CiteUlike, Delicious, Last.fm et Ponpare nous faisons les constats suivants :

- Dans CiteUlike, les résultats sont un peu similaires à ceux obtenus avec Epinions et Ciao. Les graphes de recommandation qui intègrent les trois informations secondaires sont dominant dans le tableau A.12.
- Dans Delicious, les graphes de recommandation qui intègrent les trois informations secondaires sont moins fréquents excepté pour les Top-50 et -100. D'autre part, on remarque que le graphe LSG que nous avons proposé a généralement les meilleures performances dans le tableau A.13.
- Dans Last.fm, la prise en compte des informations sur la confiance est presque inutile. Ce sont les combinaisons des informations sur le contenu des produits et la dynamique temporelle qui dominent dans le tableau A.14. Ceci confirme une fois de plus, la pertinence de notre contribution *Time−weight Content−based Session−based Temporal Graph* [100].
- Dans Ponpare, toutes les trois informations secondaires sont utiles pour obtenir les meilleures performances avec les graphes BIP et STG, mais pour le cas du graphe LSG, les attributs de description des produits sont inutiles.

A.2.1 Cas de CiteUlike

TABLE A.12 – CiteUlike - Meilleures performances en Top-10, -20, -50 et -100.

CiteUlike Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@10	1	4.53	5.52	21%	CI-LDF-IT	4.55	5.6	23%	CI-LDF	4.15	4.45	7%	IT
	2	4.53	5.46	20%	CI-LDF	4.55	5.58	22%	CI-LDF-IT	4.15	4.45	7%	EDF-IT
	3	4.53	5.46	20%	CI-EDF-IT	4.55	5.57	22%	CI-EDF	4.15	4.45	7%	LDF-IT
H@10	1	20.1	26.6	32%	CI-EDF-IT	20.0	26.4	31%	CI-EDF	16.7	21.0	25%	IT
	2	20.1	26.5	31%	CI-EDF	20.0	26.4	31%	CI-EDF-IT	16.7	21.0	25%	EDF-IT
	3	20.1	26.1	29%	CI-LDF-IT	20.0	26.2	30%	CI-LDF-IT	16.7	21.0	25%	LDF-IT
M@10	1	7.05	11.5	62%	CI-EDF-IT	7.31	11.0	51%	CI-EDF-IT	6.16	8.27	34%	CI-IT
	2	7.05	11.4	61%	CI-EDF	7.31	11.0	50%	CI-EDF	6.16	8.27	34%	CI-EDF-IT
	3	7.05	11.2	58%	CI-LDF-IT	7.31	10.6	44%	CI-LDF-IT	6.16	8.27	34%	CI-LDF-IT

CiteUlike Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@20	1	3.66	4.66	27%	CI-EDF-IT	3.63	4.64	27%	CI-EDF	3.63	3.75	3%	CI-EDF-IT
	2	3.66	4.64	26%	CI-EDF	3.63	4.62	27%	CI-EDF-IT	3.63	3.73	2%	CI-IT
	3	3.66	4.58	24%	CI-LDF-IT	3.63	4.54	24%	CI-LDF	3.63	3.73	2%	CI-LDF-IT
H@20	1	29.0	37.5	29%	CI-EDF-IT	29.2	36.6	25%	CI-EDF	24.7	30.9	25%	CI-IT
	2	29.0	37.4	28%	CI-EDF	29.2	36.6	25%	CI-EDF-IT	24.7	30.9	25%	CI-EDF-IT
	3	29.0	36.4	25%	CI-LDF-IT	29.2	36.0	23%	CI-LDF	24.7	30.9	25%	CI-LDF-IT
M@20	1	7.36	11.6	57%	CI-EDF-IT	7.52	11.2	48%	CI-EDF-IT	6.56	8.7	32%	CI-EDF-IT
	2	7.36	11.5	56%	CI-EDF	7.52	11.1	48%	CI-EDF	6.56	8.7	32%	CI-LDF-IT
	3	7.36	11.4	54%	CI-LDF-IT	7.52	10.9	44%	CI-LDF-IT	6.56	8.69	32%	CI-IT

CiteUlike Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@50	1	2.62	3.11	18%	CI-LDF	2.62	3.15	20%	CI-LDF	3.33	3.39	1%	EDF
	2	2.62	3.11	18%	CI-EDF	2.62	3.13	19%	CI-EDF	3.33	3.39	1%	LDF
	3	2.62	3.1	18%	CI-EDF-IT	2.62	3.13	19%	CI-EDF-IT	3.33	3.33	0%	-
H@50	1	42.2	50.8	20%	CI-EDF	42.1	50.8	20%	CI-LDF-IT	37.7	45.9	21%	CI-IT
	2	42.2	50.8	20%	CI-EDF-IT	42.1	50.7	20%	CI-LDF	37.7	45.9	21%	CI-LDF-IT
	3	42.2	50.4	19%	CI-LDF	42.1	50.6	20%	CI-EDF	37.7	45.8	21%	CI-EDF-IT
M@50	1	7.41	11.1	49%	CI-EDF-IT	7.43	10.7	44%	CI-EDF-IT	6.58	8.63	31%	CI-EDF-IT
	2	7.41	11.0	48%	CI-LDF-IT	7.43	10.7	43%	CI-EDF	6.58	8.63	31%	CI-LDF-IT
	3	7.41	11.0	48%	CI-EDF	7.43	10.5	41%	CI-LDF-IT	6.58	8.63	31%	CI-IT

CiteUlike Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@100	1	1.83	2.16	18%	CI-EDF	1.86	2.16	16%	CI-EDF-IT	3.32	3.39	2%	EDF
	2	1.83	2.16	17%	CI-LDF	1.86	2.16	16%	CI-LDF-IT	3.32	3.39	1%	LDF
	3	1.83	2.16	17%	CI-EDF-IT	1.86	2.16	16%	CI-LDF	3.32	3.32	0%	-
H@100	1	52.5	62.0	17%	CI-EDF-IT	52.8	62.1	17%	CI-EDF	48.7	56.9	16%	CI-IT
	2	52.5	61.9	17%	CI-EDF	52.8	62.1	17%	CI-EDF-IT	48.7	56.9	16%	CI-EDF-IT
	3	52.5	61.5	16%	CI-LDF	52.8	61.8	16%	CI-LDF	48.7	56.9	16%	CI-LDF-IT
M@100	1	7.27	10.7	47%	CI-EDF-IT	7.25	10.4	42%	CI-EDF-IT	6.45	8.1	25%	CI-LDF-IT
	2	7.27	10.7	46%	CI-EDF	7.25	10.3	42%	CI-EDF	6.45	8.09	25%	CI-IT
	3	7.27	10.6	45%	CI-LDF-IT	7.25	10.1	38%	CI-LDF-IT	6.45	8.09	25%	CI-EDF-IT

A.2.2 Cas de Delicious

TABLE A.13 – Delicious - Meilleures performances en Top-10, -20, -50 et -100.

Delicious Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@10	1	7.8	7.8	0%	-	8.73	8.73	0%	-	11.6	11.7	0%	EDF
	2	7.8	7.8	0%	LDF	8.73	8.62	-1%	IT	11.6	11.7	0%	LDF
	3	7.8	7.78	0%	IT	8.73	8.27	-5%	LDF	11.6	11.6	0%	-
H@10	1	16.7	17.0	1%	LDF	17.6	18.1	2%	LDF-IT	18.1	18.5	2%	LDF-IT
	2	16.7	17.0	1%	EDF-IT	17.6	17.9	1%	EDF	18.1	18.2	0%	IT
	3	16.7	16.9	0%	EDF	17.6	17.9	1%	LDF	18.1	18.2	0%	LDF
M@10	1	8.68	10.3	18%	EDF	11.1	11.6	4%	EDF-IT	14.0	14.1	0%	EDF
	2	8.68	10.2	18%	EDF-IT	11.1	11.6	4%	EDF	14.0	14.1	0%	LDF
	3	8.68	10.1	15%	LDF-IT	11.1	11.2	1%	LDF	14.0	14.0	0%	-

Delicious Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@20	1	5.76	5.76	0%	-	5.82	5.82	0%	-	10.6	10.7	0%	LDF
	2	5.76	5.76	0%	IT	5.82	5.82	0%	IT	10.6	10.6	0%	EDF
	3	5.76	5.76	0%	LDF	5.82	5.76	0%	EDF	10.6	10.6	0%	-
H@20	1	20.2	20.6	2%	EDF	20.2	21.4	5%	CI-EDF	19.9	20.2	1%	LDF-IT
	2	20.2	20.6	2%	EDF-IT	20.2	21.4	5%	CI-EDF-IT	19.9	20.0	0%	IT
	3	20.2	20.6	2%	CI-EDF-IT	20.2	21.1	4%	EDF	19.9	20.0	0%	LDF
M@20	1	9.28	10.8	16%	EDF	11.2	11.5	3%	EDF-IT	13.9	14.1	1%	EDF
	2	9.28	10.7	15%	EDF-IT	11.2	11.5	3%	EDF	13.9	14.0	0%	LDF
	3	9.28	10.6	13%	LDF	11.2	11.3	1%	LDF	13.9	13.9	0%	-

Delicious Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@50	1	2.78	2.92	5%	CI-IT	2.79	2.89	3%	CI-IT	10.4	10.5	1%	LDF
	2	2.78	2.87	3%	CI-LDF-IT	2.79	2.86	2%	CI-LDF-IT	10.4	10.4	0%	EDF
	3	2.78	2.87	3%	CI-EDF-IT	2.79	2.84	1%	CI-EDF-IT	10.4	10.4	0%	-
H@50	1	24.1	28.9	19%	CI-EDF-IT	24.7	28.9	17%	CI-EDF	23.3	26.5	13%	CI-EDF
	2	24.1	28.5	18%	CI-EDF	24.7	28.9	17%	CI-LDF	23.3	26.5	13%	CI-EDF-IT
	3	24.1	28.5	18%	CI-LDF	24.7	28.9	17%	CI-EDF-IT	23.3	26.4	12%	CI
M@50	1	9.39	11.0	17%	EDF	11.3	11.7	3%	EDF-IT	13.9	14.1	1%	EDF
	2	9.39	10.9	16%	EDF-IT	11.3	11.6	2%	EDF	13.9	14.0	0%	LDF
	3	9.39	10.8	14%	LDF	11.3	11.5	1%	LDF	13.9	13.9	0%	-

Delicious Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@100	1	1.54	1.68	9%	CI-EDF-IT	1.54	1.73	12%	CIU-EDF-IT	10.4	10.5	1%	LDF
	2	1.54	1.68	9%	CI-IT	1.54	1.73	12%	CIU-LDF-IT	10.4	10.4	0%	EDF
	3	1.54	1.68	8%	CIU-LDF-IT	1.54	1.72	11%	CI-EDF-IT	10.4	10.4	0%	-
H@100	1	30.1	36.4	20%	CI-EDF	30.1	36.7	21%	CI-EDF	28.6	34.5	20%	CI-IT
	2	30.1	36.1	19%	CI-EDF-IT	30.1	36.6	21%	CI-EDF-IT	28.6	34.5	20%	CI-EDF-IT
	3	30.1	36.1	19%	CIU-EDF-IT	30.1	36.0	19%	CI-LDF	28.6	34.5	20%	CI-LDF-IT
M@100	1	9.45	11.1	17%	EDF	11.4	11.7	2%	EDF-IT	13.9	14.1	1%	EDF
	2	9.45	11.0	16%	EDF-IT	11.4	11.7	2%	EDF	13.9	14.0	0%	LDF
	3	9.45	10.8	14%	LDF-IT	11.4	11.5	1%	LDF	13.9	13.9	0%	-

A.2.3 Cas de Last.fm

TABLE A.14 – Last.fm - Meilleures performances en Top-10, -20, -50 et -100.

Last.fm Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@10	1	2.48	4.66	87%	CI-LDF	2.82	4.57	62%	CI-LDF	1.68	2.44	45%	CI-EDF
	2	2.48	4.53	82%	CI-LDF-IT	2.82	4.56	61%	CI-LDF-IT	1.68	2.43	45%	CI-EDF-IT
	3	2.48	4.53	82%	CIU-LDF	2.82	4.52	60%	CI-EDF	1.68	2.42	44%	CI-IT
H@10	1	20.1	42.7	112%	CIU-LDF	25.6	40.5	57%	CIU-EDF	16.1	22.6	40%	CIU-EDF
	2	20.1	42.0	108%	CIU-LDF-IT	25.6	40.5	57%	CIU-LDF	16.1	22.4	39%	CIU
	3	20.1	41.0	103%	CI-EDF	25.6	40.2	56%	CIU-EDF-IT	16.1	22.4	39%	CI-LDF
M@10	1	6.96	14.4	107%	CI-LDF-IT	9.75	15.4	58%	CIU-LDF-IT	6.51	10.1	54%	CI-LDF
	2	6.96	14.2	104%	CI-LDF	9.75	15.3	57%	CIU	6.51	9.91	52%	CI-EDF
	3	6.96	14.2	104%	CIU-LDF-IT	9.75	15.3	56%	CIU-LDF	6.51	9.85	51%	CI

Last.fm Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@20	1	2.98	6.55	119%	CI-LDF	3.51	6.36	81%	CI-LDF	2.18	2.81	29%	CI-LDF
	2	2.98	6.47	117%	CI-EDF	3.51	6.34	80%	CI-LDF-IT	2.18	2.81	28%	CIU-EDF
	3	2.98	6.28	110%	CI-LDF-IT	3.51	6.22	77%	CIU-EDF-IT	2.18	2.8	28%	CIU-LDF
H@20	1	29.9	56.8	89%	CIU-LDF	34.4	54.0	56%	CIU-LDF	25.1	30.9	23%	CIU
	2	29.9	56.5	89%	CIU-LDF-IT	34.4	54.0	56%	CIU-EDF-IT	25.1	30.9	23%	CIU-EDF
	3	29.9	56.0	87%	CIU-EDF	34.4	54.0	56%	CIU-LDF-IT	25.1	30.9	23%	CIU-LDF
M@20	1	7.14	14.6	105%	CI-LDF	9.98	15.3	53%	CIU-EDF	6.69	9.64	44%	CI-EDF
	2	7.14	14.5	103%	CIU-LDF	9.98	15.3	53%	CIU-EDF-IT	6.69	9.58	43%	CI-LDF
	3	7.14	14.5	102%	CIU-LDF-IT	9.98	15.2	52%	CIU-LDF-IT	6.69	9.42	40%	CI

Last.fm Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@50	1	3.66	8.42	130%	CI-LDF	4.27	8.07	88%	CIU-LDF	2.77	3.37	21%	CI-EDF
	2	3.66	8.41	129%	CI-EDF	4.27	8.07	88%	CIU-EDF	2.77	3.36	21%	CI-LDF
	3	3.66	8.2	124%	CIU-LDF	4.27	8.03	87%	CIU	2.77	3.29	18%	CI
H@50	1	47.0	72.6	54%	CIU-EDF	49.7	70.9	42%	CIU-EDF	42.7	45.5	6%	CIU-EDF
	2	47.0	71.4	51%	CI-EDF	49.7	70.6	41%	CIU	42.7	45.5	6%	CIU-LDF
	3	47.0	71.1	51%	CI-LDF	49.7	70.6	41%	CIU-LDF	42.7	45.2	5%	CIU
M@50	1	6.57	14.2	115%	CI-LDF	8.94	14.4	60%	CIU-LDF-IT	6.18	8.71	40%	CI-EDF
	2	6.57	13.7	108%	CIU-LDF-IT	8.94	14.4	60%	CIU-LDF	6.18	8.6	39%	CI-LDF
	3	6.57	13.7	108%	CIU-LDF	8.94	14.2	58%	CIU-EDF	6.18	8.5	37%	CIU-EDF

Last.fm Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@100	1	3.89	8.08	107%	CI-LDF	4.2	7.83	86%	CIU-EDF	2.87	3.38	17%	CI-EDF
	2	3.89	8.07	107%	CIU-EDF	4.2	7.79	85%	CIU-LDF	2.87	3.38	17%	CI-LDF
	3	3.89	8.06	107%	CI	4.2	7.79	85%	CIU	2.87	3.38	17%	CI
H@100	1	59.5	81.9	37%	CIU-EDF	62.1	80.7	29%	CIU-EDF	55.8	59.5	6%	CI
	2	59.5	81.4	36%	CI-EDF	62.1	80.2	29%	CIU	55.8	59.5	6%	CI-EDF
	3	59.5	81.2	36%	CIU-IT	62.1	80.2	29%	CIU-LDF	55.8	59.5	6%	CI-LDF
M@100	1	5.57	12.7	127%	CI-LDF	7.41	12.7	70%	CIU	6.01	8.59	42%	CI-EDF
	2	5.57	12.1	118%	CI-LDF-IT	7.41	12.5	68%	CIU-LDF	6.01	8.43	40%	CI-LDF
	3	5.57	12.0	116%	CI-EDF	7.41	12.5	68%	CIU-EDF	6.01	8.31	38%	CI

A.2.4 Cas de Ponpare

TABLE A.15 – Ponpare - Meilleures performances en Top-10, -20, -50 et -100.

Ponpare Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@10	1	4.0	6.9	72%	CIU-EDF	3.55	6.45	81%	CIU-EDF	3.35	5.26	56%	LDF
	2	4.0	6.9	72%	CIU-EDF-IT	3.55	6.45	81%	CIU-EDF-IT	3.35	5.26	56%	LDF-IT
	3	4.0	6.45	61%	CI-EDF	3.55	5.83	64%	CI-EDF	3.35	5.11	52%	IT
H@10	1	10.3	36.1	250%	CIU-EDF	10.3	33.3	222%	CIU-EDF	7.3	25.6	251%	LDF
	2	10.3	36.1	250%	CIU-EDF-IT	10.3	33.3	222%	CIU-EDF-IT	7.3	25.6	251%	LDF-IT
	3	10.3	32.4	213%	CI-EDF	10.3	28.1	172%	CI-EDF	7.3	21.5	195%	IT
M@10	1	2.97	20.2	580%	CIU-EDF	3.72	19.7	430%	CIU-EDF	2.56	12.4	383%	EDF-IT
	2	2.97	20.2	580%	CIU-EDF-IT	3.72	19.7	430%	CIU-EDF-IT	2.56	12.3	382%	LDF-IT
	3	2.97	18.7	528%	CI-EDF	3.72	16.3	338%	CIU-LDF	2.56	12.3	380%	IT

Ponpare Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@20	1	2.41	4.73	96%	CIU-EDF	2.5	4.34	73%	CIU-EDF	2.39	3.59	50%	LDF-IT
	2	2.41	4.73	96%	CIU-EDF-IT	2.5	4.34	73%	CIU-EDF-IT	2.39	3.59	50%	LDF
	3	2.41	4.1	70%	CI-EDF-IT	2.5	3.74	49%	CI-EDF	2.39	3.21	34%	IT
H@20	1	14.8	48.0	225%	CIU-EDF	14.9	44.7	198%	CIU-EDF	12.1	32.9	172%	LDF-IT
	2	14.8	48.0	225%	CIU-EDF-IT	14.9	44.7	198%	CIU-EDF-IT	12.1	32.7	170%	LDF
	3	14.8	40.7	175%	CI-EDF-IT	14.9	37.4	150%	CI-EDF	12.1	27.0	123%	EDF-IT
M@20	1	3.27	20.8	536%	CIU-EDF	3.94	20.3	415%	CIU-EDF	2.85	12.6	343%	EDF-IT
	2	3.27	20.8	535%	CIU-EDF-IT	3.94	20.3	415%	CIU-EDF-IT	2.85	12.6	343%	LDF-IT
	3	3.27	19.1	483%	CI-EDF-IT	3.94	16.9	328%	CIU-LDF	2.85	12.6	341%	IT

Ponpare Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@50	1	1.77	3.01	69%	CIU-EDF	1.78	2.87	60%	CIU-EDF	1.88	2.09	11%	LDF-IT
	2	1.77	3.0	69%	CIU-EDF-IT	1.78	2.87	60%	CIU-EDF-IT	1.88	2.06	9%	LDF
	3	1.77	2.74	54%	CI-EDF	1.78	2.41	35%	CI-EDF	1.88	1.88	0%	EDF
H@50	1	26.9	71.7	166%	CIU-EDF	26.9	67.4	150%	CIU-EDF	26.7	41.1	54%	LDF-IT
	2	26.9	71.5	166%	CIU-EDF-IT	26.9	67.4	150%	CIU-EDF-IT	26.7	40.7	52%	LDF
	3	26.9	64.6	140%	CI-EDF	26.9	56.8	111%	CI-EDF	26.7	38.4	44%	CIU-LDF-IT
M@50	1	3.49	21.4	512%	CIU-EDF	4.19	20.7	395%	CIU-EDF	3.23	12.8	296%	LDF
	2	3.49	21.4	511%	CIU-EDF-IT	4.19	20.7	395%	CIU-EDF-IT	3.23	12.7	293%	EDF-IT
	3	3.49	19.5	458%	CI-EDF	4.19	17.2	309%	CIU-LDF	3.23	12.7	293%	LDF-IT

Ponpare Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@100	1	1.04	1.88	80%	CIU-EDF-IT	1.04	1.8	72%	CIU-EDF	1.43	1.55	8%	LDF
	2	1.04	1.88	79%	CIU-EDF	1.04	1.8	72%	CIU-EDF-IT	1.43	1.44	0%	EDF
	3	1.04	1.72	64%	CI-EDF	1.04	1.56	48%	CI-EDF	1.43	1.43	0%	-
H@100	1	33.8	89.9	165%	CIU-EDF-IT	32.9	85.8	160%	CIU-EDF	32.9	55.3	68%	LDF
	2	33.8	89.7	165%	CIU-EDF	32.9	85.8	160%	CIU-EDF-IT	32.9	55.3	68%	LDF-IT
	3	33.8	81.9	142%	CI-EDF	32.9	73.8	124%	CI-EDF	32.9	54.6	65%	CIU-LDF
M@100	1	3.6	21.4	494%	CIU-EDF	4.29	20.8	383%	CIU-EDF-IT	3.3	12.8	286%	LDF-IT
	2	3.6	21.4	493%	CIU-EDF-IT	4.29	20.8	383%	CIU-EDF	3.3	12.8	286%	LDF
	3	3.6	19.5	442%	CI-EDF	4.29	17.0	295%	CIU-LDF-IT	3.3	12.7	283%	EDF-IT

Chapitre B

Liste des publications

B.1 Time Weight Content-based Extensions of Temporal Graphs for Personalized Recommendation

Time Weight Content-based Extensions of Temporal Graphs for Personalized Recommendation

Armel Jacques Nzekon Nzeko'o[1,2,3], Maurice Tchuente[1,2] and Matthieu Latapy[3]
[1]*Sorbonne Universités, UPMC Université Paris 06, IRD, UMI 209 UMMISCO, F-93143, Bondy, France*
[2]*CETIC, Université de Yaoundé I, Faculté des Sciences, Departement d'Informatique, BP 812, Yaoundé, Cameroon*
[3]*Sorbonne Universités, UPMC Université Paris 06, CNRS, UMR 7606, LIP6, F-75005, Paris, France*

Keywords: Temporal Recommendation, Long- and Short-term Preferences, Session-based Temporal Graph, Time Weight Content-based Graph, Time-averaged Hit Ratio, PageRank, Injected Preference Fusion.

Abstract: Recommender systems are an answer to information overload on the web. They filter and present to customers, small subsets of items that they are most likely to be interested in. Users' interests may change over time, and accurately capturing this dynamics in such systems is important. Sugiyama, Hatano and Yoshikawa proposed to take into account the user's browsing history. Ding and Li were among the first to address this problem, by assigning weights that decrease with the age of the data. Others authors such as Billsus and Pazzani, Li, Yang, Wang and Kitsuregawa proposed to capture long- and short- terms preferences and combine them for personalized search or news access. The Session-based Temporal Graph (STG) is a general model proposed by Xiang et al. to provide temporal recommendations by combining long- and short-term preferences. Later, Yu, Shen and Yang have introduced Topic-STG, which takes into account topics information extracted from data. In this paper, we propose Time Weight Content-based STG that generalizes Topic STG. Experiments show that, using Time-Averaged Hit Ratio as measure, this time weight content-based extension of STG leads to performance increases of 4%, 6% and 9% for CiteUlike, Delicious and Last.fm datasets respectively, in comparison to STG.

1 INTRODUCTION

The amount of information in web sites like Amazon, Netflix and Last.fm is considerable and fast-growing. For users, browsing and searching in such data has become very difficult. To solve this problem, recommender systems filter and present to customers, small subsets of items that they are most likely to be interested in.

Early recommender systems did not take into account temporal information (Herlocker et al., 1999). This is strong limitation because user's profiles and context evolve with time. In this regards Sugiyama et al., (2004) proposed to adapt results according to time-evolving user profiles, based for instance on browsing history. Ding and Li (2005) were among the first to take time explicitly into account by assigning decreasing weights according to their age. Later, some authors proposed to capture both long- and short-term preferences and combine them for recommendations (Billsus and Pazzani, 2000; Li et al., 2007).

Interest in temporal recommender systems increased considerably since the 2009 Netflix grand prize (Koren, 2009). Koren's solution was based on a dataset with items rated by users, whereas in practice, data often contains the history of users in terms of their interactions with items. For instance, Last.fm offers datasets in which each line indicates the fact that user *u* listened to song *i* at time *t*.

In this line of research, Xiang et al., (2010) propose Session-based Temporal Graphs (STG), which model long- and short-term preferences separately. However, they ignore features of items, and so they miss for instance the fact that interest in a piece of music is related to its author. In order to improve this, Yu et al., (2014) extend the STG into Topic-STG for personalized tweet recommendation. They add *topic nodes* to the STG and link tweets to their topics.

These recommender graphs process edges regardless of their age. This fails to capture the fact that in most situation, recent transactions are the most likely to reflect user preferences in the near

Nzeko'o, A., Tchuente, M. and Latapy, M.
Temporal Recommendation, Long- and Short-term Preferences, Session-based Temporal Graph, Time Weight Content-based Graph, Time-averaged Hit Ratio, PageRank, Injected Preference Fusion..
DOI: 10.5220/0006288202680275
In *Proceedings of the 13th International Conference on Web Information Systems and Technologies (WEBIST 2017)*, pages 268-275
ISBN: 978-989-758-246-2

future (Ding and Li, 2005). For example, the clothes that a boy wears have great variability related to his age, thus, the impact of his past dress style on his future dress style decreases with time. To take this into account, we propose here to weight edges according to their age, so that older edges have lower influence. We propose the Time Weight Content-based STG, in which edges are labelled with their last occurrence time, and we use an exponential decay function proposed by Ding and Li (2005) to weight edges accordingly.

The remaining of this paper is organized as follow: Section 2 presents Session-based Temporal Graphs and the two recommendation algorithms, PageRank (Page et al., 1999) and Injected Preference Fusion (IPF) (Xiang et al., 2010) on which our work is built. Section 3 introduces the Time Weight Content-based STG model. Section 4 is devoted to experiments. We discuss related work in Section 5, and we summarize our findings in Section 6.

2 BACKGROUND

We use the notations and definitions proposed by Xiang et al., (2010), together with some additional concepts related to content and time, summarized in Table 1.

2.1 Session-based Temporal Graph

We consider data under the form of a link stream, i.e. a set of triples (t, u, i) representing the fact that user u has selected item i at time t. For each user u (resp. each item i), we define user node v_u (resp. item node v_i). We denote by **T** the time span of the dataset and we divide **T** into time slices of equal duration. For each of these slices T, we define session node $v_{u,T}$. A session is not defined here as a time slice during which the user interacts with the system. It rather corresponds to a time slice during which a user has a specific behaviour.

A session-based temporal graph $G(U, S, I, E, w)$ is a directed bipartite graph with three types of nodes: U is the set of user nodes, S the set of session nodes and I the set of item nodes. The function w: $E \rightarrow \boldsymbol{R}$ is a non-negative weight function for edges. The set of edges, E, is obtained as follows. For each triplet (t, u, i), let us consider T the time slice to which t belongs. Then, E contains edges (v_u , v_i) and (v_i , v_u), which represent long-term preference between user u and item i; and E contains ($v_{u,T}$, v_i) and (v_i , $v_{u,T}$), which represent short-term preferences.

Table 1: Notations and definitions.

Symbol	Description
G	bipartite graph STG
CG	bipartite graph Content-based STG
TG	bipartite graph Time weight Content-based STG
E	edge set in any graph
V	set of all nodes in any graph
U, I, S, C	user node set, item node set, session node set, content node set
v_u , v_i , $v_{u,T}$, v_c	user node, item node, session node, content node
w	weight function defined on STG edges
w_C	weight function defined on Content-based STG edges
w_T	time weight function defined on Time weight Content-based STG edges
$\psi(v_k, v_{k+1})$	propagation function of IPF from v_k to v_{k+1}
$out(v)$	out node set of the node v
ρ	parameter to control the preference propagation
β	dose of long-term preference injected to user node
η	parameter to adjust the edge weight from item nodes to user/session nodes
η_c	parameter to control the influence of content features in the preference propagation
τ_0	parameter used to compute the time weight function
α	damping factor for PageRank personalization

The weight function is defined as:

$$w(v,v') = \begin{cases} 1 & v \in U \cup S, v' \in I \\ \eta_u & v \in I, v' \in U \\ \eta_s & v \in I, v' \in S \end{cases} \quad (1)$$

In (1), η_u models the influence of long-term preferences and η_s models the influence of short-term preferences. To simplify the model, we can use $\eta = \eta_u / \eta_s$ for η_u and 1 for η_s

Fig. 1 is an example of STG with 3 user nodes, 5 session nodes, 7 item nodes and 2 time slices. It shows that user u_1 has selected items i_1, i_2, user u_2 has selected items i_3, i_4 and user u_3 has selected item i_5 during the first time slice T_1. During the second time slice T_2, user u_1 has selected i_3 and user u_3 has selected i_6 and i_7.

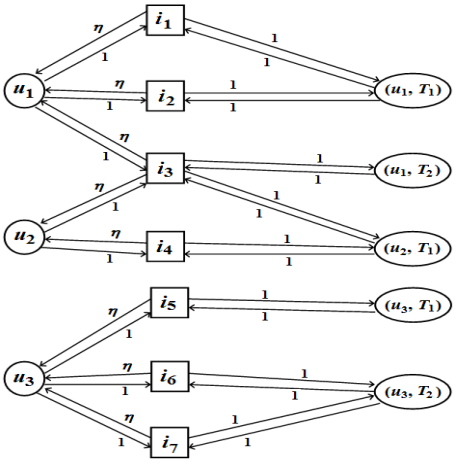


Figure 1: Example of STG.

2.2 Temporal Personalized Random Walk

The Temporal Personalized Random Walk (TPRW) (Xiang et al., 2010) is a personalization of the PageRank algorithm defined by Page et al. (Page et al., 1999) for nodes ranking in graphs. It was defined to tackle temporal recommendation using the idea of Haveliwala (Haveliwala, 2002). It corresponds to the following formula:

$$PR = \alpha \cdot M \cdot PR + (1 - \alpha) \cdot d \quad (2)$$

where α is the damping factor, M is a transition matrix and vector d is a user-specific personalized vector indicating which nodes the random walker will jump to after a restart.

When making recommendations for user u, vector d favors user node v_u and the most recent session node $v_{u,T}$ as follows:

$$d(v) = \begin{cases} \beta & v = v_u \\ 1 - \beta & v = v_{u,T} \\ 0 & otherwise \end{cases} \quad (3)$$

In other words, long-term preferences are injected to user node v_u and short-term preferences are injected to session node $v_{u,T}$ through vector d.

When we implement the PageRank with iterative power law method, we stop when the difference of two consecutives rank vectors is of norm less than or equal to a threshold ε.

2.3 Injected Preference Fusion

The IPF algorithm is an extension of the random walk with injection of preferences and customization of preference propagation. To recommend items to a user u, the algorithm proceeds in 3 steps:

- Injection of long-term preferences β on the user node v_u and injection of short-term preferences (1 – β) on the most recent session node $v_{u,T}$ of user u.
- Propagation of preferences by random walk of length 3 on the graph according to the formula.

$$\psi(v_k, v_{k+1}) = \left(\frac{w(v_k, v_{k+1})}{\sum_{v' \epsilon\, out(v_k)} w(v_k, v')}\right)^{\rho} \quad (4)$$

- where $out(v_k)$ denotes the set of out-neighbors of node v_k, ρ is a parameter used to tune the propagation process, $w(v_k, v_{k+1})$ is the weight of arc (v_k, v_{k+1}) and $\psi(v_k, v_{k+1})$ is the proportion of preference of v_k that is propagated to v_{k+1}.
- Recommendation of Top-N items that have received the greatest preference values and that user u has not yet selected.

The IPF random walk length is limited to 3 following experimental result (Xiang, et al., 2010).

3 TIME WEIGHT CONTENT-BASED EXTENSIONS OF STG

In this section, we first illustrate how to construct Content-based STG (CSTG) which is similar to Topic-STG (Yu et al., 2014). We end by showing how to construct Time Weight Content-based STG (TCSTG).

3.1 Content-based Session-based Temporal Graph

The basic STG model neglects item properties which can contain significant information for the prediction of user's behaviour. This motivated Phuong et al., (2008) to add to the user-item bipartite graph, new nodes corresponding to content. The same idea is applied here to obtain Content-based STG.

To construct the Content-based STG, we need to have item properties in our data, so we don't use a set of triples like in the construction of STG. We rather use a set of quadruples (t, u, i, c) where t, u and i have the same meaning as in STG, and c is a content feature of i.

Content-based STG $CG(U, S, I, C, E, w_C)$ is a directed graph obtained from the STG $G(U, S, I, E, w)$ by adding for any link (t, u, i, c), the six additional arcs (v_u , v_c), (v_c , v_u), ($v_{u,T}$, v_c), (v_c , $v_{u,T}$),

(v_i , v_c) and (v_c , v_i). With respective weights 1, η, 1, 1, η_c, η_c as illustrated in Fig. 2.

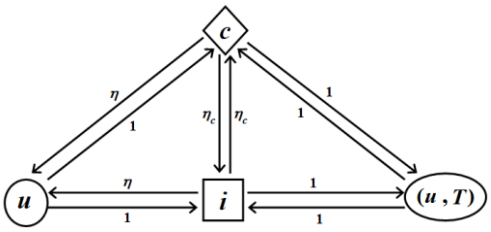


Figure 2: Edge weights in content-based STG.

3.2 Time Weight Content-based Session-based Temporal Graph

The Content-based STG neglects the ages of edges when assigning weights. So, it cannot capture the evolution of users' interest which we assume to be sensitive to time as suggested by Ding and Li (2005). The recommendation model presented here assigns a greater weight to recent edges and lower weight to older edges. More precisely, the weight of the arc (v, v') is defined by:

$$w_T(v, v') = f(t) \cdot w(v, v') \quad (5)$$

where $w(v, v')$ is the weight in the graph without time weight, t is the most recent time at which edge (v, v') appears and $f(t)$ is a time-dependent decay function as in (Ding and Li, 2005). Here we take

$$f(t) = e^{-\lambda \cdot (t_r - t)} \quad (6)$$

where λ is the decay rate and ($t_r - t$) is the difference in second between time t_r at which we are making recommendations and t.

The parameter λ can also be defined as $\lambda = 1/\tau_0$ where τ_0 is the delay after which the weight of an edge reduces by 1/2. τ_0 is also called half life parameter.

4 EXPERIMENTS

We have conducted a set of experiments to examine the performance of Time weight Content-based STG. For each model, we consider various values of parameters and we retain the best performance. We also implemented the classic bipartite user-item graph (BIP) to show the effects of taking into account long- and short-term preferences in graph models.

The experiment environment is as follow: the executions of our programs are done using a computer with 64GB of RAM and 16 processors Intel of 2.93GHz and 4MB of cache. For implementation, we have used the Python 2.7 language and the Networkx 1.11 module for graph manipulation. Note that we have changed the Networkx *PageRank* in order to stop when convergence is not reached after 100 iterations. We used SQLite 3 as DBMS and Matplotlib 1.4.0 to produce graphics.

4.1 Data Description

Following the example of Xiang et al, our goal is to make recommendations based on implicit data from various real world domains. To this effect, we first perform experiments on the social bookmarking datasets CiteUlike and Delicious (Cantador et al., 2011) which were used in (Xiang et al., 2010). This provides a good basis for the evaluation of the improvement obtained when temporal aspects are taken into account. We also use data from Lastfm (Celma, 2010), a web site where users can listen to music because in this domain the fashion effect plays an important role with the consequence that, the impact of past tastes on the future ones decreases with time.

We model our data as link streams {(t, u, i, c)}, where any quadruple has different interpretation depending on domains. In the case of CiteUlike and Delicious, each quadruplet means that user u has bookmarked page i at time t with tag c. And for the Last.fm data, this means that user u has listened to song i at time t and c is the author of i.

Before modeling our data as link streams, we performed a filtering by ignoring items and users that did not appear a number of times higher than a given threshold σ. Table 2 provides details on our data: date of the first link, date of the last link, total duration of link streams, threshold used, number of users, number of items, number of content features and number of links.

Table 2: Data statistics.

	Start date	End date	Duration	σ
CiteUlike	2010-01-01	2010-07-02	183 days	10
Delicious	2010-05-11	2010-11-09	183 days	7
Last.fm	2005-02-14	2005-08-16	183 days	8
	Users	**Items**	**Content**	**Links**
CiteUlike	1318	424	4216	16885
Delicious	894	298	2789	13825
Last.fm	135	1054	225	41604

4.2 Experiment and Evaluation

The evaluation process is done periodically as in (Li and Tang, 2008) and (Lathia et al., 2009). Before starting experiments, we have to divide the link streams into time windows of a fixed length Δ. We fix Δ to 15 days because humans live at a monthly pace, and the first 15 days are generally characterized by consumption behaviours just after getting salary, that are different from the ones observed during the last 15 days of the month. To simplify the experimentation process, we adopt the same Δ as the length of session when constructing STG. Here after, N denotes the number of time slices.

For each time window W_k, for k=1,..,N-1, we proceed as follows:

- Construct the graphs corresponding to data of W_1, W_2, .. ,W_k.
- Compute the Top-N recommendations for users who have selected at least one "new item" during the time window W_{k+1}.
- Evaluate the algorithm by computing the ratio of users for which at least one of these Top-N items recommends has been selected during W_{k+1}. This proportion is also call Hit Ratio HR_k (Karypis, 2001).

After determining the Hit Ratio for each window, compute the Time Averaged Hit Ratio (TAHR) that is a weighted combination of the N-1 values obtained above for the Hit Ratio. The weight of each Hit Ratio HR_k is the number of corresponding users U_k as in the following equation:

$$TAHR = \frac{\sum_k HR_k \cdot U_k}{\sum_k U_k} \quad (7)$$

Note that, although the convergence may be very slow for some examples, we have noticed that, in most cases, the convergence is achieved after at most 100 iterations.

4.3 Exploration of the Range of the Parameters

Let us see how the parameters are obtained in Table 3. We proceed as in (Xiang et al., 2010). The parameters correspond to the vector [$\tau 0$, β, η, η_c, ρ, α], whose components are numbered 1, 2, 3, 4, 5, 6 from left to right. This vector is initialized to [0, 0.5, 0.5, 0.5, 0.5, 0.5]. Then, we consider the values of $\tau 0$ shown in the second row of Table 3, while maintaining the other parameters at their initial value 0.5. We perform ten experiments and take for $\tau 0$ the value corresponding to the best performance. For instance we obtain for $\tau 0$ the interval [7, 30] for TPRW-TCSTG in CiteUlike dataset as shown in Table 4. Given this optimal value for $\tau 0$ we then give to β the eleven successive values shown in the third row of Table 3, while maintaining the other parameters at their initial value. We obtain for β the interval [0.4, 1] for TPRW-TCSTG in CiteUlike dataset as shown in Table 4. This process is repeated for the remaining parameters.

Figure 3 shows all the variations of Time-Averaged Hit Ratio with parameter values in the case of CiteUlike. The complete set of parameters explored is shown in Table 3 and the best values obtained with this procedure are shown in Table 4.

Table 3: Parameters values.

Parameters	Initial value	Set of values
$\tau 0$ (in days)	0	0, 1, 7, 15, 30, 45, 60, 90, 180, 365
β	0.5	0.1 × i for i = 0..10
η	0.5	0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.5, 3, 5, 10, 15, 20, 30, 50, 100
η_c	0.5	0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.5, 3, 5, 10, 15, 20, 30, 50, 100
ρ	0.5	0.1 × i for i = 0..10
α	0.5	0.1 × i for i = 0..10

4.4 Accuracy Comparison

The performances of PageRank and IPF applied to STG, Time-weight content-based STG, Content-based STG and classical bipartite graph, for the three datasets are presented in Table 5. It can be seen that PageRank applied to the Time weight content-based STG gives the best results, followed by Content-based STG. Moreover, STG is always better than the classical bipartite graph, which confirms the relevance of STG.

Moreover, for PageRank applied to the Time weight content-based STG, the optimal value of the half life parameter $\tau 0$, is less than one month for social bookmarking dataset, but is greater than one month for music dataset. This may be due to the fact that the impact of web pages that someone consulted in the past on those that he is likely to consult in the future decreases very quickly with time. On the contrary, tastes are more stable for music and the impact of a past music does not decrease very quickly.

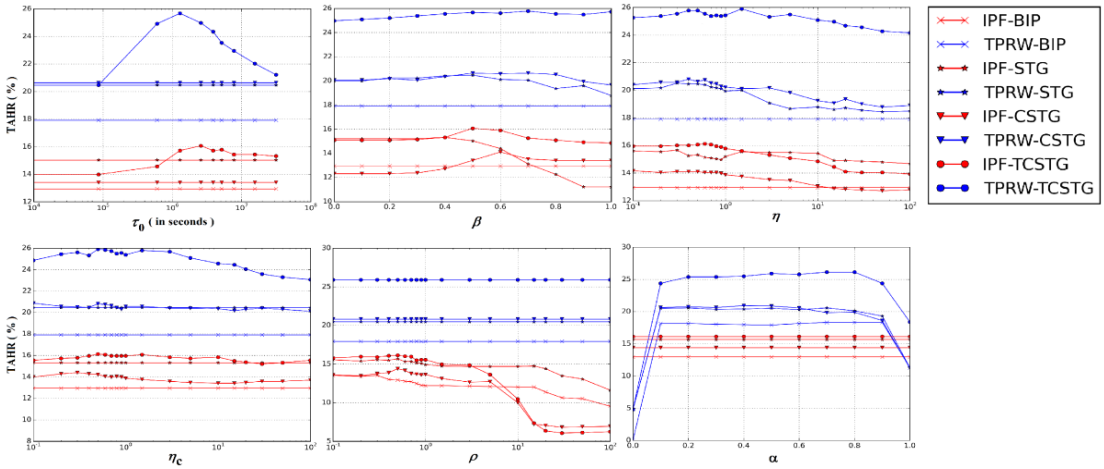


Figure 3: Variation of Time Averaged Hit Ratio with parameter values in the case of CiteUlike.

Table 4: Best parameter values.

CiteUlike	$\tau 0$	β	η	η_c	ρ	α
IPF-BIP	-	-	-	-	0.1-0.3	-
IPF-STG	-	0.0-0.5	0.1-0.3	-	0.1-0.7	-
IPF-CSTG	-	0.5-1	0.0-0.9	0.2-0.5	0.5-0.6	-
IPF-TCSTG	15-60	0.5-0.6	0.1-0.9	0.4-1.5	0.1-1	-
TPRW-BIP	-	-	-	-	-	0.1-0.9
TPRW-STG	-	0.0-0.7	0.3-0.6	-	-	0.1-0.7
TPRW-CSTG	-	0.5-0.8	0.3-0.6	0.1-0.7	-	0.1-0.6
TPRW-TCSTG	7-30	0.4-1	0.3-1.5	0.5-3	-	0.5-0.8

Delicious	$\tau 0$	β	η	η_c	ρ	α
IPF-BIP	-	-	-	-	0.1-10	-
IPF-STG	-	0.0-0.4	0-0.1	-	0.1-0.6	-
IPF-CSTG	-	0.5	15-50	0.3-0.9	0.4-1.5	-
IPF-TCSTG	1-7	0.5-0.6	0.5-0.8	50-100	0.5-1.5	-
TPRW-BIP	-	-	-	-	-	0.1-0.9
TPRW-STG	-	0.0-0.4	0.1-0.2	-	-	0.2-0.5
TPRW-CSTG	-	0.0-0.6	15-100	0.2-0.8	-	0.5-0.7
TPRW-TCSTG	7	0-1	0.5-0.8	0-0.1	-	0.1-04

Last.fm	$\tau 0$	β	η	η_c	ρ	α
IPF-BIP	-	-	-	-	0.1-0.8	-
IPF-STG	-	0.5-1	0.9-1.5	-	1.5-10	-
IPF-CSTG	-	0-0.4	0-0.3	30-100	0.4-0.6	-
IPF-TCSTG	1-15	0-0.4	0.1-0.3	1-100	10-50	-
TPRW-BIP	-	-	-	-	-	0.1-0.5
TPRW-STG	-	0.5-0.7	0.1-1.5	-	-	0.2-0.5
TPRW-CSTG	-	0.2-0.4	0.2-0.5	5-30	-	0.4-0.6
TPRW-TCSTG	30-90	0.5-0.7	0.4-0.6	1-5	-	0.4-0.7

We also think that PageRank has better performance because in this algorithm the propagation process is not limited to the proximity of the source node as in IPF. Indeed PageRank also favours the recommendation of the most popular items of the graph because, even when they are far from the source node, they can be reached and then have a great influence thanks to their high degree.

Table 5: Performances for the best parameters.

	CiteUlike	**Delicious**	**Last.fm**
	TAHR (%)	**TAHR (%)**	**TAHR (%)**
IPF-BIP	13.5	7.3	16.3
IPF-STG	15.7	8.6	18.2
IPF-CSTG	14.4	6.4	28.9
IPF-TCSTG	16.1	9.7	26.6
TPRW-BIP	18.3	8.8	27.9
TPRW-STG	20.6	9.2	30.2
TPRW-CSTG	20.9	10.7	37.7
TPRW-TCSTG	**26.1**	**13.2**	**38.9**

5 RELATED WORK

In this section, we present some work on time aware recommender systems followed by recommender systems that use item properties. Finally, we present some graph-based recommender systems.

5.1 Time Aware Recommender Systems

Ding et al. (Ding and Li, 2005) propose the use of an exponential decay function to assign greater weights to latest ratings when computing similarities in collaborative filtering. Subsequently, Liu et al., (2010) have proposed an incremental collaborative filtering where one decay function is used to compute similarities and another one is used for prediction. Recently, Karahodža et al., (2015) assumed that the importance of interest granted to an item decreases in a similar manner for similar users.

Some recommender systems are based on the

assumption that importance of information is ephemeral. Thus, Lathia et al., (2009) set a time window size, then, any information is used during one time slice and ignored at the next time window. Such recommender systems only capture short-term preferences.

Some studies are not based only on short-term preferences but also consider that importance of some information persists over time (Li et al., 2007). The STG model (Xiang et al., 2010) extends this work but ignores item properties.

5.2 Content-based Recommender Systems

The content-based recommender systems seek to recommend similar items to the one the user already like. As Lops et al. (Lops et al., 2011) argue, the basic idea is to match features associated to users' preferences and items so as to recommend new items that address their needs. This approach is already used in various domains such as books recommendation on Amazon website based on their description (Mooney and Roy, 2000), and web pages recommendation (Pazzani et al., 1996).

Although content-based recommender systems can propose items that have not already been purchased in the past, it is also useful to use user similarities by combining this approach with collaborative filtering techniques. Indeed, Balabanovic and Shoham (1997) and Basu et al., (1998) show that the combination of collaborative filtering and content-based filtering may result in a recommender system that eliminates the weaknesses of both approaches. In this paper, we have used a graph model to realize this combination.

5.3 Graph based Recommender Systems

The simplest graph-based recommender systems only use user-item links. A bidirectional edge is created between a user node and an item node if the user has purchased the concerned item. Finally, an item is recommended to a user if the user has not yet purchased that item and if there is a path from the user to that item. The most used recommender algorithms on the graphs are based on the random walk (Baluja et al., 2008), like PageRank and IPF which are used in this paper.

The use of graph paths to recommend new items can be seen as collaborative filtering where similaritys defined through node distance. However, such recommender graphs do not take into consideration item properties. To remedy this limitation, Phuong et al., (2008) have constructed a recommender graph in which they have added a third node type: the type "content". The obtained recommender system is actually a combined collaborative filtering and content-based filtering. The associated graph ignores the temporal aspect of data and therefore cannot accurately capture short- and long-term preferences. Yu et al., (2014) propose the Topic-STG which combines those two preferences and takes into account topics related to tweets. However, those models handle edges regardless of their age. This is not in accordance with concept drift. This is why we propose a new extension of STG where edge weights are decreased using a time decay function as in (Li and Tang, 2008).

6 CONCLUSIONS

This paper proposes time weight content-based extensions of the temporal graph model introduced by Xiang et al., As in Topic-STG introduced by Yu, Shen and Yang, we represent content by nodes, but we penalize older interactions. Experiments show that, using Time-Averaged Hit Ratio as measure, this time weight content-based extension of STG leads to performance increases of 4%, 6% and 9% for CiteUlike, Delicious and Last.fm datasets respectively, in comparison to STG. This gives evidence of the fact that the age of interactions is a relevant feature for recommender systems.

More experiments using datasets from various domains are needed in order to adjust the length of time windows and other parameters.

ACKNOWLEDGEMENTS

This work is funded in part by the African Center of Excellence in Information and Communication Technologies (CETIC), the UPMC-IRD PDI program, by the European Commission H2020 FETPROACT 2016-2017 program under grant 732942 (ODYCCEUS), by the ANR (French National Agency of Research) under grants ANR-15-CE38-0001 (AlgoDiv) and ANR-13-CORD-0017-01 (CODDDE), by the French program "PIA-Usages, services et contenus innovants" under grant O18062-44430 (REQUEST), and by the Ile-de-France program FUI21 under grant 16010629 (iTRAC).

REFERENCES

Balabanović, M., & Shoham, Y. (1997). Fab: content-based, collaborative recommendation. *Communications of the ACM , 40* (3), 66-72.

Baluja, S., Seth, R., Sivakumar, D., Jing, Y., Yagnik, J., Kumar, S., et al. (2008, April). Video suggestion and discovery for youtube: taking random walks through the view graph. *In Proceedings of the 17th international conference on World Wide Web. ACM* , 895-904.

Basu, C., Hirsh, H., & Cohen, W. (1998, July). Recommendation as classification: Using social and content-based information in recommendation. *In Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence* , 714-720.

Billsus, D., & Pazzani, M. J. (2000). User modeling for adaptive news access. *User modeling and user-adapted interaction , 10* (2-3), 147-180.

Cantador, I., Brusilovsky, P., & Kuflik, T. (2011). 2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011). *In Proceedings of the 5th ACM conference on Recommender systems* .

Celma, Ò. (2010). *Music Recommendation and Discovery in the Long Tail.* Springer.

Ding, Y., & Li, X. (2005, October). Time Weight Collaborative Filtering. *In Proceedings of the 14th ACM international conference on Information and knowledge management, ACM* , 485-492.

Haveliwala, T. H. (2002, May). Topic-sensitive pagerank. *In Proceedings of the 11th international conference on World Wide Web* , 517-526.

Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J. (1999). *An algorithmic framework for performing collaborative filtering.* In : Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. ACM.

Karahodža, B., Donko, D., & Šupić, H. (2015, May). Temporal Dynamics of Changes in Group Users Preferences in Recommender Systems. *In 38th International Convention on Information and Communication Technology, Electronics and Microelectronics, IEEE* , 1262-1266.

Karypis, G. (2001, October). Evaluation of Item-Based Top-N Recommendation Algorithms. *In Proceedings of the tenth international conference on Information and knowledge management* , 247-254.

Koren, Y. (2009). The bellkor solution to the netflix grand prize. *Netflix prize documentation , 81*, 1-10.

Lathia, N., Hailes, S., & Capra, L. (2009, July). Temporal Collaborative Filtering With Adaptive Neighbourhoods. *In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, ACM* , 796-797.

Li, L., Yang, Z., Wang, B., & Kitsuregawa, M. (2007). Dynamic adaptation strategies for long-term and short-term user profile to personalize search. *In Advances in Data and Web Management* , 228-240.

Li, Y., & Tang, J. (2008). Expertise Search in a Time-varying Social Network. *In The Ninth International Conference on Web-Age Information Management, IEEE* , 293-300.

Liu, N. N., Zhao, M., Xiang, E., & Yang, Q. (2010). Online Evolutionary Collaborative Filtering. *In Proceedings of the fourth ACM conference on Recommender systems. ACM* , 95-102.

Lops, P., De Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. *In Recommender systems handbook. Springer US* , 73-105.

Mooney, R. J., & Roy, L. (2000, June). Content-based book recommending using learning for text categorization. *In Proceedings of the fifth ACM conference on Digital libraries* , 195-204.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999, November). The PageRank citation ranking: bringing order to the web. *Technical Report, Stanford InfoLab* .

Pazzani, M., Muramatsu, J., & Billsus, D. (1996). Syskill & Webert: Identifying interesting web sites. *In Proceedings of the thirteenth American Association for Artificial Intelligence AAAI/IAAI , 1*, 54-61.

Phuong, N. D., Thang, L. Q., & Phuong, T. M. (2008). A Graph-Based Method for Combining Collaborative and Content-Based Filtering . *In Trends in Artificial Intelligence. Springer Berlin Heidelberg* , 859-869.

Sugiyama, K., Hatano, K., & Yoshikawa, M. (2004, May). Adaptive web search based on user profile constructed without any effort from users. *In Proceedings of the 13th international conference on World Wide Web, ACM* , 675-684.

Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., et al. (2010). Temporal recommendation on graphs via long-and short-term preference fusion. *In : Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM* , 723-732.

Yu, J., Shen, Y., & Yang, Z. (2014). Topic-STG : Extending the Session-based Temporal Graph Approach for Personalized Tweet Recommendation. *In Proceedings of the companion publication of the 23rd international conference on World wide web companion. International World Wide Web Conferences Steering Committee* , 413-414.

B.2 Link Stream Graph for Temporal Recommendations

LINK STREAM GRAPH FOR TEMPORAL RECOMMENDATIONS

Armel Jacques Nzekon Nzeko'o[1,2,3,*], Maurice Tchuente[2], Matthieu Latapy[3]

[1] Sorbonne Université, IRD, UMI 209 UMMISCO, F-93143, Bondy, France
[2] Université de Yaoundé I, CETIC, Faculté des Sciences, Département d'Informatique, BP 812, Yaoundé, Cameroun
[3] Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, LIP6, F-75005, Paris, France

ABSTRACT

Several researches on recommender systems are based on explicit rating data, but in many real world e-commerce platforms, ratings are not always available, and in those situations, recommender systems have to deal with implicit data such as users' purchase history, browsing history and streaming history. In this context, classical bipartite user-item graphs (BIP) are widely used to compute top-N recommendations. However, these graphs have some limitations, particularly in terms of taking temporal dynamic into account. This is not good because users' preference change over time. To overcome this limit, the Session-based Temporal Graph (STG) was proposed by Xiang *et al*. to combine long- and short-term preferences in a graph-based recommender system. But in the STG, time is divided into slices and therefore considered discontinuously. This approach loses details of the real temporal dynamics of user actions. To address this challenge, we propose the Link Stream Graph (LSG) which is an extension of link stream representation proposed by Latapy *et al*. and which allows to model interactions between users and items by considering time continuously. Experiments conducted on four real world implicit datasets for temporal recommendation, with 3 evaluation metrics, show that LSG is the best in 9 out of 12 cases compared to BIP and STG which are the most used state-of-the-art recommender graphs.

Keywords: *Recommender graph, Temporal recommendation, Session-based Temporal Graph, Link Streams Graph*

1. INTRODUCTION

The ever-growing number of items available on e-business, booking, or news platforms makes it crucial to help users finding the best ones for them. This is the goal of recommender systems, which are generally classified in two categories: rating prediction and top-N item recommendation [1]. The goal of the first one is to predict the rating value that a user is likely to give to an unrated item. The goal of the second one is to rank all items for a giving user and propose him/her the N most interesting ones.

Many state-of-the-art recommender systems tackle the rating prediction problem [2]–[4]. However, in many on-line platforms we observe the lack of explicit rating and the abundance of data in browsing history that link users to items and conserve the timestamps of the user action. Indeed, in those platforms, recommendations are offered to users in the form of list. All this justifies the fact that top-N recommendation is becoming the most used standard of the field [5].

To achieve the goal of top-N recommendation, the most used and the most studied approach is collaborative filtering (CF) which is based on correlation between user interests [5]–[7]. Some of CF techniques are graph-based and generally built on classical bipartite graph (BIP) [8]. This graph failed to accurately capture the users' interests' dynamics. This is an important limitation because the tastes and preferences of users evolve over time. It may be due to the fact that items are out the date or because user profiles are changing.

To tackle this issue, Xiang *et al.* [9] proposed Session-based Temporal Graph (STG) which simultaneously models users' long-term and short-term preferences over time. Time is divided in slices to model users' short-term preferences in the STG. This last remark shows that STG considers time discontinuously. Thus, details of the temporal or structural dynamics of user actions can be lost depending on whether the slice size is very large or very small. So, the problem of accurately capturing the dynamics of user preferences remains.

In this paper, we address this challenge by proposing the Link Stream Graph which is an extension of link stream representation proposed by Latapy *et al.* [10] and which allows to model interactions between users and items by considering time continuously. Before describing our model in Section 4, we first present the problem statement in Section 2, following by related work on graph-based recommender systems in Section 3. Section 5 presents experiments setup and results on four real world datasets. Section 6 is dedicated to discussion and Section 7 concludes this work.

2. PROBLEM STATEMENT AND DATA MODELING

In this section, we present the top-N recommendation problem, for which we propose a new graph, and the data modeling formalism adopted for input data.

2.1. Problem statement

We follow here the time-aware top-N recommendation definition of Stefanidis *et al.* [11]: we consider an integer N, a set U of users, a set I of items, and the recommender system aims at finding a set $R_{u,t} \subseteq I$ of N items that are most likely to be of interest for user u at time t.

2.1. Data modeling

The recommendations rely on data regarding past interest of users in U regarding items in I. We model these as a bipartite link stream $L = (T, U, I, E)$ where U and I are the sets of users and items defined above, T is the time interval during which the interests of users in U regarding items in I were observed, and $E \subseteq T \times U \times I$ is a set of links (t, u, i) indicating that user u was interested in item i at time t. If (t, u, v) is in E, then we say that u and v are linked at time t in L, or that (t, u, v) in L. Such a link stream typically models purchases (u bought product i at time t), interest in a cultural item (like movie watching or song listening), etc. See [12], [13] for a full description of the link stream formalism.

2.1. Guiding example

In the following, we will always use the following guiding example. The set of users is $U = \{u_1, u_2\}$, the set of items is $I = \{i_1, i_2, i_3, i_4\}$, the observation period is $T = [\alpha , \omega]$ where α and ω are constant, and $E = \{ (t_1, u_1, i_1), (t_1, u_2, i_3), (t_2, u_1, i_2), (t_2, u_2, i_3), (t_3, u_2, i_4), (t_4, u_1, i_3), (t_5, u_2, i_4), (t_6, u_1, i_2) \}$, which means for instance that u_1 was interested in item i_2 at time t_2. Figure 1 illustrates the link stream describe above.

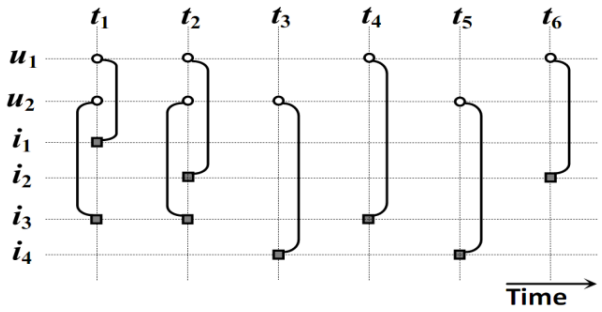


Figure 1. Example of link stream describes in Section 2.3.

3. RELATED WORK

In the following, we make a formal description of the classical bipartite graph (BIP) and its extension by the Session-based Temporal Graph (STG). This section ends with the description of personalized PageRank used to compute top-N recommendations on graphs.

3.1. Classical bipartite graph

The first graph we consider is the classical bipartite graph [8], that we denote by BIP. It is a directed bipartite graph (U, I, E) where U is the set of users, I the set of items, $E \subseteq U \times I$ the set of links such that $(u, i) \in E$ if user u was interested in item i during the observation period, *i.e.*, there exists $t \in T$ such that (t, u, v) in L. In addition, and to ensure consistence with other recommender graphs defined below, we consider a trivial weight function w such that $w(x, y) = 1$ for all $(x, y) \in E$. See Figure 2(a) for an illustration.

We can deduce an user- or item-projection of BIP [14]. However, projected graphs lose information about user-item interactions, especially the aspect: who selected what? In addition, the update of those graphs is more expensive in dynamic contexts as time aware systems. These remarks justify the fact that the classical bipartite graph is the most used in recommender systems, particularly when temporal aspects are taken into account. Figure 2 illustrates the classical bipartite graph, the user-projected and the item-projected graphs of the link stream describes in Section 2.3.

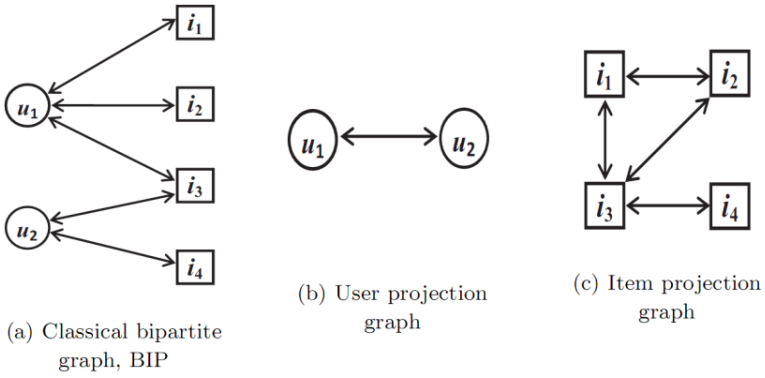


Figure 2. Classical bipartite graph, User projection and Item projection graphs obtained from our guiding example. The weight of each edge is 1.

3.2. Session-based Temporal Graph

Session-based Temporal Graph proposed by Xiang *et al.* [15] is a recommender graph designed to overcome the fact that BIP does not integrate time dimension. Indeed, STG combines short-term and long-term users' behaviors to accurately capture user's preferences over time. Thus, in addition to the sets of user and item nodes already present in BIP, a set S of session nodes is added, and each of those nodes represents the short term preferences of a user.

This graph encodes time information with session nodes defined as follows. First, for a given Δ, the observation interval T is divided into $|T|/\Delta$ time slices $T_k = [(k\text{-}1)\cdot\Delta\ ,\ k\cdot\Delta]$ of equal duration Δ. Then, S contains the couples (u, T_k) such that there exists a link (t, u, i) in E with $t \in T_k$. In other words, each user leads to a session node $(u, T_k) \in S$ for each time interval T_k during which this user was active. This leads to the definition of STG as a tripartite graph (U, I, S, E) where U and I are defined as for BIP, E contains the links of BIP and in addition the links between (u, T_k) and the items selected by user u during time slice T_k.

In addition, we consider the directed weight function w such that, for a given constant η_s, for all (x, y) in E, $w(x, y) = \eta_s$ if $x \in I$ and $y \in S$, $w(x, y) = 1$ otherwise. See Figure 3 for an illustration with $\Delta = 3$, leading to two time slices T_1 and T_2 and three session nodes.

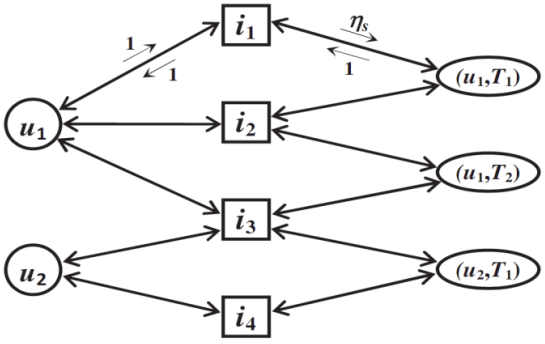


Figure 3. Session-based temporal graph (STG) for the guiding example.

3.3. Recommendation on graphs

In the previous section, we present some recommender graphs; we now show how to use them to actually solve the top-N recommendation problem. Once we have a graph, it is necessary to apply an algorithm to get the top-N recommendation. Most of these algorithms rely on the random walk, for example we have PageRank [16] and Injected Preference Fusion (IPF) [15], but also other algorithms like Hypertext-Induced Topic Selection (HITS) can be used [17].

In this paper, we focus on PageRank because it is very efficient and widely used in recommender systems [18], [19]. More precisely, we use the Temporal Personalized Random Walk (TPRW) defined by Xiang *et al.* [15], which is a

personalized extension of the classical PageRank [16]. It is defined according to the idea of Haveliwala *et al*. [20] using the following equation:

$$\mathrm{PR} = \alpha \cdot M \cdot \mathrm{PR} + (1 - \alpha) \cdot d$$

where α is the damping factor, M is the transition matrix of the graph and d is a user-specific personalized vector indicating which nodes the random walker will jump to after a restart.

To recommend new items to user u at time t, vector d assumes that users tend to jump to their own user nodes and session nodes after a restart. We obtain for a given user u in BIP: $d(u) = 1$ and $d(v) = 0$ if $v \neq u$; in STG: $d(u) = \beta$, $d(u, T_k) = 1 - \beta$ if T_k is u most recent session node, and $d(v) = 0$ for any other node v. Then, we run TPRW over the recommender graph to get the preferences of user u for each item i. The output is the set of N items with higher preference.

4. LINK STREAM GRAPH

STG takes time dimension into account but it divides time into slices. However, slicing time loses details about dynamics of events. If slices are very short, structural aspects are lost and if very long, details on dynamics are lost. To overcome this limit, we are inspired by the link stream representation of Latapy *et al*. [10], [13] and we design the bipartite link stream graph (LSG), which considers time in a continuous way.

4.1. Model description

This graph is first defined by a set of nodes representing users and items over time: $\{(t, u) : \exists i, (t, u, i) \in E\} \cup \{(t, i) : \exists u, (t, u, i) \in E\}$. In other words, each user u is represented by the nodes (t, u) such that a link involves u in L a time t, and each item is represented similarly. We then define the set of links $\{((t, u), (t, i)) : (t, u, i) \in E\} \cup \{((t, u), (t', u)) : \exists i, (t, u, i) \in E, t' = min\{x : x > t \text{ and } \exists i', (x, u, i') \in E\} \cup \{((t, i), (t', i)) : \exists u, (t, u, i) \in E, t' = min\{x : x > t \text{ and } \exists u', (x, u', i) \in E\}$. In other words, each user node (t, u) is linked to both the item nodes (t, i) such that $(t, u, i) \in E$ and to the next user node representing u. Item nodes are linked similarly.

In addition, we consider the directed weight function w such that, for a given constant η_s, for all $((t, x), (t', y)) \in E$, $w((t, x), (t', y)) = \eta_s$ if $x = y$ and $t > t'$, and $w((t, x), (t', y)) = 1$ otherwise. See Figure 4 for an illustration on our guiding example.

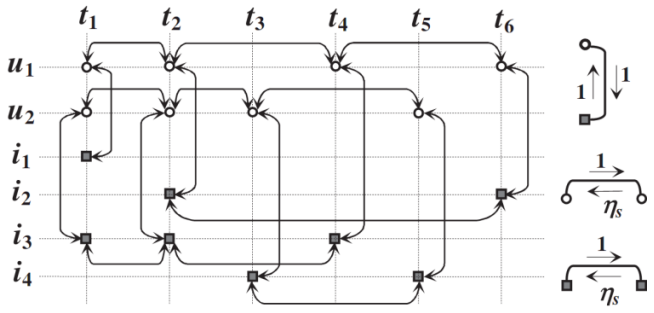


Figure 4. Link Stream Graph for the guiding example in Section 2.3.

4.2. Recommendation on Link Stream Graph

The LSG recommendation process follows the same principle as for BIP and STG in section [16]. To recommend new items to user u at time t, vector d is personalized as follow: $d(t_k, u) = 1$ if t_k is the largest value such that $t_k \leq t$ and node (t_k, u) exists, and $d(v) = 0$ for any other node v. After running TPRW on LSG, the preference of u for an item i is the sum of preferences for nodes (t_k, i) for all t_k.

5. EXPERIMENTS

We performed experiments to compare the performances of LSG, BIP and STG. The data used are those from 4 real-world platforms. In the rest of this section, we present datasets, experiment protocol and obtained results.

5.1. Data description

We use publicly available datasets, the first one is extracted from product reviews Ciao[1][21], the second one is extracted from on-line shopping website Ponpare[2]. We also used data from Delicious[3] and CiteUlike[4], which are social bookmarking websites.

Data are modeled as link stream $\{(t, u, i)\}$ and the interpretation of any triplet depends on the domain. In the case of on-line shopping, each triplet means that the customer u has bought item i at time t. In social bookmarking data, this means that user u has bookmarked page i at time t. In Ciao, we have set of tuples (t, u, i, r) meaning that user u has assigned the rating $r \in \{0, 1, 2, 3, 4, 5\}$ to item i at time t. Since we work only with positive links between users and items, we discard all tuples such that the rating it contains is lower than 2.5 or the average rating of involved user.

Data are filtered such that only users and items that appear respectively at least σ_u and σ_i times are considered. Table 1 provides details on data used: date of the first link, date of the last link, total duration of link stream, user-threshold, item-threshold, numbers of users, items, content features and links.

Table 1. Data statistics.

	Ciao	Ponpare	Delicious	CiteUlike
Start date	2007-01-01	2011-07-01	2010-05-10	2010-01-01
End date	2010-12-31	2011-11-01	2010-11-09	2010-09-01
σ_u	1	10	4	10
σ_i	1	10	4	10
Users	879	1 322	1 175	1 726
Items	6 005	1 115	1 352	619
User-Item	8 109	12 863	7 652	9 360
Sparsity	99.85%	99.13%	99.52%	99.12%
Links	8 109	177 005	35 558	24 772

5.2. Experiment and Evaluation

Evaluating recommender systems is a difficult task. In this paper, we use three classical metrics for top-N recommendations: F1-score (F1), Hit Ratio (HR) and Mean Average Precision (MAP) [22]. Higher values of these metrics indicate better recommendation performance.

F1-score is a trade-off between ranking precision and recall such that optimizing F1-score is more robust than optimizing precision or recall. Precision is the fraction of good recommendations over all recommended items and recall is the fraction of good recommendations over all relevant items to recommend. For one user u, $Precision = \frac{hit_N(u)}{N}$, $Recall = \frac{hit_N(u)}{I_{new}(u)}$ and $F1 = 2 \cdot \frac{Precision \times Recall}{Precision + Recall} = \frac{2 \cdot hit_N(u)}{I_{new}(u) + N}$ where N is the length of recommendation list, $hit_N(u)$ denotes the number of good recommendations to u in the top-N items and $I_{new}(u)$ is the set of new items to recommend to u. For all users the equation of F1-score is: $F@N = \frac{\sum_{u \in U} 2 \cdot hit_N(u)}{\sum_{u \in U} (I_{new}(u) + N)}$.

Hit Ratio is the fraction of users to whom the recommender system has made at least one good recommendation over all users: $H@N = \frac{\sum_{u \in U} (hit_N(u) > 0)}{|U|}$.

Mean Average Precision considers the order of items in the top-N recommendation in order to give better evaluation scores to results that recommend better items first: $M@N = \frac{\sum_{u \in U} AP_N(u)}{|U|}$ where $AP_N(u) = \frac{1}{hit_N(u)} \sum_{k=1}^{N} \frac{hit_k(u) \times h(k)}{k}$ is the average precision of top-N recommendations done to user u and $h(k) = 1$ if the k-th recommended item is a good recommendation and 0 otherwise.

These metrics evaluate a given top-N recommendation. Since we actually can't perform recommendations on live users, we perform evaluation on past data described above. Following a periodically evaluation process established by previous works [7], [23], [24]. For each dataset, the input link stream is divided into 8 time windows of equal length. For each time window W_k, for $k = 1 .. 7$, we proceed as follow:

[1] https://www.cse.msu.edu/~tangjili/trust.html
[2] https://www.kaggle.com/c/coupon-purchase-prediction
[3] https://grouplens.org/datasets/hetrec-2011/
[4] http://www.citeulike.org/faq/data.adp

- Build graphs that correspond to data of W_1, W_2, .. ,W_k (training set)
- Compute Top-N recommendations for users who have selected at least one new item during the time window W_{k+1} (test set)
- Compute for each evaluation metric M the numerator $M_{num\ k}$ and the denominator $M_{deno\ k}$ of its definition, given above.

After determining $M_{num\ k}$ and $M_{deno\ k}$ of each time window, we compute the Time Averaged (TA) value of the concerned evaluation metric: $TA(M) = \frac{\sum_k M_{num\ k}}{\sum_k M_{deno\ k}}$. This leads to a time-averaged value of F1-score, Hit ratio and MAP, which we all use for evaluation. Indeed, evaluation metrics can be in disagreement [25], and so using several metrics is essential to obtain accurate insight on result quality.

5.3. Optimal tuning parameter estimation

There are several metrics to evaluate these graphs and some of them have more than one parameter. So, exhaustive search for the best values is out of reach. Many subtle techniques exist to explore the parameter space in search for good values. Since this search is not the focus of this paper, we use a simple approach called Randomized Search Cross-Validation [26]. This method randomly selects parameter values in a predefined set of possible values, usually designed to span well the whole set of values. Here, we use 50 such random settings, sampled in the set defined by Table 2.

Table 2. Predefined values of parameters.

	parameter meaning	predefined values
Δ	STG session duration	7, 30, 60, 90, 180, 365, 540, 730 days
β	STG long-term preference	0.1, 0.3, 0.5, 0.7, 0.9
η_s	weight connected to the past	0, 0.1, 0.2, 0.5, 1, 2, 5, 10
α	damping factor for PageRank	0.05, 0.1, 0.15, 0.3, 0.5, 0.7, 0.9

5.4. Accuracy Comparison

Table 3 shows the best results for each recommender graph. We have dataset names in the rows, graphs and evaluation metrics used in columns. The values in bold are the best performances obtained. This table shows the relevance of the LSG which is the best 9 out of 12 times.

Table 3. Best recommender graphs results.

	F1-score (%)			Hit ratio (%)			MAP (%)		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
Ciao	1.78	1.73	**3.0**	5.63	**6.35**	**6.35**	2.18	**2.23**	2.11
Ponpare	5.52	6.32	**6.67**	11.8	12.4	**29.5**	4.03	4.29	**11.7**
Delicious	5.08	5.13	**6.56**	10.7	11.7	**13.7**	5.15	5.49	**5.96**
CiteUlike	**7.65**	7.39	7.48	28.7	**29.5**	28.5	10.7	**10.9**	**10.9**

Regarding the analysis of the LSG parameters, we found that the parameter η_s is generally less than 0.5 for all datasets of this study. In the particular case of Ponpare, η_s is always nil for the best performance. This observation confirms the importance of concept drift [4], [27] which suggest the penalization of old data to achieve good qualities of recommendation.

6. DISCUSSION AND PERSPECTIVES

LSG has a high space complexity, because for each link (t, u, i), two nodes and three bidirectional edges are systematically created. However, in BIP (resp. STG), two nodes (resp. three nodes) and one (resp. two) bidirectional edges if they do not already exist in the graph. This complexity is more remarkable when each user selects the same product several times. Thus, a future work on LSG can be the reduction of the space complexity while maintaining or improving its performance.

In this paper, the LSG is used for top-N recommendation, but it can also be applied for link prediction problem or for any other problem with temporal and/or structural features. In such cases, other algorithms to compute prediction may be proposed, and other way of weighting edges can be proposed to optimize the LSG performance according to the problem studied.

7. CONCLUSION

This paper presents the Link Stream Graph (LSG) which takes time into account in a continuous way. We used it to capture temporal dynamic of users' preferences in a top-N recommender system. Comparisons with classical bipartite graph (BIP) and Session-based Temporal Graph (STG), which are the main state-of-the-art recommender graphs, using 3 evaluation metrics, confirm the relevance of the LSG which is the best 9 out of 12 possible cases. For example, there is an improvement from: 1.73 to 3.0% of the F1-score in the Ciao dataset; 11.7 to 13.7% of the Hit ration in the Delicious dataset; and 4.29 to 11.7% of the MAP in the Ponpare dataset. As future work, someone can focus on reducing the space complexity of the LSG or propose other way of weighting edges while maintaining or improving its performance.

8. ACKNOWLEDGEMENTS

This work is funded in part by the African Center of Excellence in Information and Communication Technologies (CETIC), the Sorbonne University-IRD PDI program, and by the ANR (French National Agency of Research) under grant ANR-15-CE38-0001 (AlgoDiv).

9. REFERENCES

[1] H. Steck, "Evaluation of recommendations: rating-prediction and ranking," in *Proceedings of the 7th ACM conference on Recommender systems*, 2013, pp. 213–220.

[2] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 1999, pp. 230–237.

[3] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput.*, no. 1, pp. 76–80, 2003.

[4] Y. Koren, "Collaborative filtering with temporal dynamics," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 447–456.

[5] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *Proceedings of the fourth ACM conference on Recommender systems*, 2010, pp. 39–46.

[6] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 143–177, 2004.

[7] A. J. Nzekon Nzeko'o, M. Tchuente, and M. Latapy, "Time Weight Content-Based Extensions of Temporal Graphs for Personalized Recommendation," in *WEBIST 2017-13th International Conference on Web Information Systems and Technologies*, 2017.

[8] S. Baluja *et al.*, "Video suggestion and discovery for youtube: taking random walks through the view graph," in *Proceedings of the 17th international conference on World Wide Web*, 2008, pp. 895–904.

[9] L. Xiang *et al.*, "Temporal recommendation on graphs via long-and short-term preference fusion," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 723–732.

[10] M. Latapy, T. Viard, and C. Magnien, "Stream graphs and link streams for the modeling of interactions over time," *arXiv Prepr. arXiv1710.04073*, 2017.

[11] K. Stefanidis, I. Ntoutsi, K. Nørvåg, and H.-P. Kriegel, "A framework for time-aware recommendations," in *International Conference on Database and Expert Systems Applications*, 2012, pp. 329–344.

[12] M. Latapy, T. Viard, and C. Magnien, "Stream graphs and link streams for the modeling of interactions over time," *Soc. Netw. Anal. Min.*, vol. 8, no. 1, p. 61, 2018.

[13] T. Viard, M. Latapy, and C. Magnien, "Computing maximal cliques in link streams," *Theor. Comput. Sci.*, vol. 609, pp. 245–252, 2016.

[14] T. Zhou, J. Ren, M. Medo, and Y.-C. Zhang, "Bipartite network projection and personal recommendation," *Phys. Rev. E*, vol. 76, no. 4, p. 46115, 2007.

[15] L. Xiang *et al.*, "Temporal recommendation on graphs via long- and short-term preference fusion," *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discov. data Min. - KDD '10*, p. 723, 2010.

[16] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web.," 1999.

[17] B. J. Mirza, B. J. Keller, and N. Ramakrishnan, "Studying recommendation algorithms by graph analysis," *J. Intell. Inf. Syst.*, vol. 20, no. 2, pp. 131–160, 2003.

[18] M. Gori, A. Pucci, V. Roma, and I. Siena, "ItemRank: A Random-Walk Based Scoring Algorithm for Recommender Engines.," in *IJCAI*, 2007, vol. 7, pp. 2766–2771.

[19] I. Şora, "A PageRank based recommender system for identifying key classes in software systems," in *2015 IEEE 10th Jubilee International Symposium on Applied Computational Intelligence and Informatics*, 2015, pp. 495–500.

[20] T. H. Haveliwala, "Topic-sensitive pagerank," in *Proceedings of the 11th international conference on World Wide Web*, 2002, pp. 517–526.

[21] J. Tang, H. Gao, and H. Liu, "mTrust: Discerning multi-faceted trust in a connected world," in *Proceedings of the fifth ACM international conference on Web search and data mining*, 2012, pp. 93–102.

[22] R. Baeza-Yates, B. de A. N. Ribeiro, and others, *Modern information retrieval*. 2011.

[23] Y. Li and J. Tang, "Expertise search in a time-varying social network," in *Web-Age Information Management, 2008. WAIM'08. The Ninth International Conference on*, 2008, pp. 293–300.

[24] N. Lathia, S. Hailes, and L. Capra, "Temporal collaborative filtering with adaptive neighbourhoods," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, 2009, pp. 796–797.

[25] A. Gunawardana and G. Shani, "A survey of accuracy evaluation metrics of recommendation tasks," *J. Mach. Learn. Res.*, vol. 10, no. Dec, pp. 2935–2962, 2009.

[26] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, no. Feb, pp. 281–305, 2012.

[27] Y. Ding and X. Li, "Time weight collaborative filtering," in *Proceedings of the 14th ACM international conference on Information and knowledge management*, 2005, pp. 485–492.

B.3 A general graph-based framework for top-N recommendation using content, temporal and trust information

JOURNAL OF INTERDISCIPLINARY METHODOLOGIES AND ISSUES IN SCIENCE

A general graph-based framework for top-N recommendation using content, temporal and trust information

Armel Jacques NZEKON NZEKO'O[*1,2,3], **Maurice TCHUENTE**[1,2], **Matthieu LATAPY**[3]

[1]Sorbonne Université, IRD, UMMISCO, F-93143, Bondy, France
[2]Université de Yaoundé I, CETIC, FS, Département d'Informatique, BP 812, Yaoundé, Cameroun
[3]Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, LIP6, F-75005, Paris, France

*Corresponding author: armel.nzekon@lip6.fr

DOI: 10.18713/JIMIS-300519-5-2
Submitted: February 27, 2019 - Published: May 12, 2019
Volume: **5** - Year: **2019**
Issue: **Graph and network analysis**
Editor: *Vincent Labatut*

Abstract

Recommending appropriate items to users is crucial in many e-commerce platforms. One common approach consists in selecting the N most relevant items for each user. To achieve this, recommender systems rely on various kinds of information, like item and user features, past interest of users for items and trust between users. Current systems generally use only one or two such pieces of information, which limits their performance. In this paper, we design and implement GraFC2T2, a general graph-based framework to easily combine various kinds of information for top-N recommendation. It encodes content-based features, temporal and trust information into a graph model, and uses personalized PageRank on this graph to perform recommendation. Experiments are conducted on Epinions and Ciao datasets, and comparisons are done with systems based on matrix factorization and deep learning using F1-score, Hit ratio and MAP evaluation metrics. The results show that combining different kinds of information generally improves recommendation. This shows the relevance of the proposed framework.

Keywords

Top-N Recommendation; Graph; Collaborative Filtering; Content; Temporal information; Trust; PageRank; Link streams

I INTRODUCTION

Many e-commerce platforms have large and fast growing sets of items to present to users. For instance, Amazon had a total of 53.38 millions books as on January 10th, 2018[1]. Such huge

[1]https://www.scrapehero.com/many-products-amazon-sell-january-2018/

quantities of products make it challenging for users to search and find interesting items for them. Then, they often rely on the help provided by recommender systems.

Various approaches co-exist, the most classical ones being rating prediction and top-N recommendation (Steck, 2013). Rating prediction estimates the rating value that a user is likely to give to items. Top-N recommendation ranks items for a given user and selects the N most interesting ones, for a given N. Many research works are dedicated to rating prediction. This requires explicit rating data whereas, in many platforms dedicated for instance to e-commerce, ratings are not available, and recommender systems have to deal with implicit data such as users' purchase, browsing and streaming history. In such situations, top-N recommendation can still be carried out (Cremonesi *et al.*, 2010).

In addition to the previous remark, top-N recommender systems are everywhere from on-line shopping websites to video portals (Christakopoulou and Karypis, 2016). For all these reasons, we focus here on top-N recommendation problem from positive implicit feedback, a problem already considered in many papers such as Rendle *et al.* (2009); Ning and Karypis (2011); Shi *et al.* (2012) and Guo *et al.* (2017).

One of the main families of techniques, called Collaborative Filtering (CF), takes benefit from correlations between user interests. Initially, CF recommender systems focused only on user-item interactions (Konstan *et al.*, 1997; Herlocker *et al.*, 1999; Sarwar *et al.*, 2001) and did not integrate side information among the following list: item features like the genre of a movie or the author of a song, context of interactions like location, timestamps or weather, and trust between users. Since such side information strongly influences user choices (for instance, users may listen to a new song because they like the singer), performances of such systems may be limited. In addition, side information helps solving problems like cold start and data sparsity (Burke, 2002; Adomavicius and Tuzhilin, 2005; Massa and Avesani, 2007; Campos *et al.*, 2014).

For these reasons, much effort was devoted to the inclusion of side information into CF techniques. For instance, hybrid systems incorporate item features in order to combine CF and content-based filtering (CBF) (Burke, 2002; Chen *et al.*, 2016; Shu *et al.*, 2018). Likewise, a winning team of the Netflix competition (Koren *et al.*, 2009; Koren, 2009) included temporal information into a CF system in order to track the dynamics of user interests and increase recommendation accuracy. Including trust information in order to take into account the fact that people tend to adopt items already chosen by trusted friends is also possible (Papagelis *et al.*, 2005; Massa and Avesani, 2007; Guo *et al.*, 2017).

Some previous works consider only one type of side information, and therefore fail to capture the combined influence of several types of side information on user interests. Others works suggest that progress in this direction may significantly improve recommendation, and combine two kinds of side information into CF (Ning and Karypis, 2012; Yu *et al.*, 2014; Strub *et al.*, 2016; Nzekon Nzeko'o *et al.*, 2017). However, to the best of our knowledge, none of these approaches include content-based features, users' preferences temporal dynamics and trust relationships between users simultaneously.

Our goal in this paper is to propose a general graph-based recommender framework that makes it easy to combine variety of side information. However, recommender systems are used in very diverse situations, which makes the design of a fully general system out of reach. We therefore made several assumptions which, although very general, do not apply to some contexts. First, we focus on top-N recommendation task because it is prevalent in many on-line

shopping recommender systems like video portals. In addition, we considered the situations where the recommender system aims at offering each user a product that he/she has not yet selected in the past. In some situations, clients may repeatedly buy the same product, but this is a quite different problem. We also we assumed that recent activities are more important than older ones, a situation known as concept drift. This is often but not always true in practice; interest in a given kind of product may for instance be periodic, like for birthday gifts or seasonal needs. Extending our work in this direction is promising, when data is available. Finally, we consider positive links only (that typically represent a purchase), as this is the most prevalent case in practice; considering more subtle feedback from users, and in particular negative feedback, is a very promising direction for future work.

Contribution

In this paper, we propose GraFC2T2, a general graph-based framework for top-N recommendation combining content-based features, temporal information, and trust into a personalized PageRank system. The design of this framework is very modular in order to make it easy to include other side information and/or replace personalized PageRank by another graph-based method. Thanks to GraFC2T2, it becomes easy to explore the benefit of using various kinds of side information, and then to find appropriate parameters for combining them for particular applications. We conduct experiments on Epinions and Ciao datasets to illustrate the use of GraFC2T2, and we show that it outperforms state-of-the-art thanks to the increased use of side information.

Figure 1 summarizes the global architecture of GraFC2T2, made of two big parts: the recommender graph construction, and the use of this graph to perform recommendation. The recommender graph encodes available information by combining a basic graph, which we detail in Section II, with methods to capture content-based features and edge weight capturing time information, which we detail in Section III. Then, we use the obtained recommender graph to perform recommendation, with a trust-aware personalized PageRank detailed in Section IV.

Notice that our framework makes it possible to explore wide sets of modeling choices, as well as to incorporate additional possibilities if needed. We illustrate this on two real-world datasets from Epinions and Ciao in Sections V and VI. Section VII discusses related work.

This work builds upon our previous paper (Nzekon Nzeko'o *et al.*, 2017), which extends the Session-based Temporal Graph proposed by Xiang *et al.* (2010) by adding time-weight and content-based information. On the other hand, the data representation that we use is the link stream formalism, presented by Latapy *et al.* (2018). This model allowed us to propose the Link Stream Graph (Nzekon Nzeko'o *et al.*, 2019).

We provide an online implementation of our framework[2] in order to help other researchers and practitioners to conduct experiments on their own datasets, and to test the relevance of new ideas and features.

II DATA MODELING

We consider a set U of users, a set I of items, and a time interval T, and we assume that we observed the past interest of users in U for items in I during T. We model this data by a bipartite link stream $L = (T, U, I, E)$ where $E \subseteq T \times U \times I$ is a set of links: each link (t, u, i) in E represents a purchase (u bought product i at time t), an interest in a cultural item (like movie

[2] https://github.com/nzekonarmel/GraFC2T2

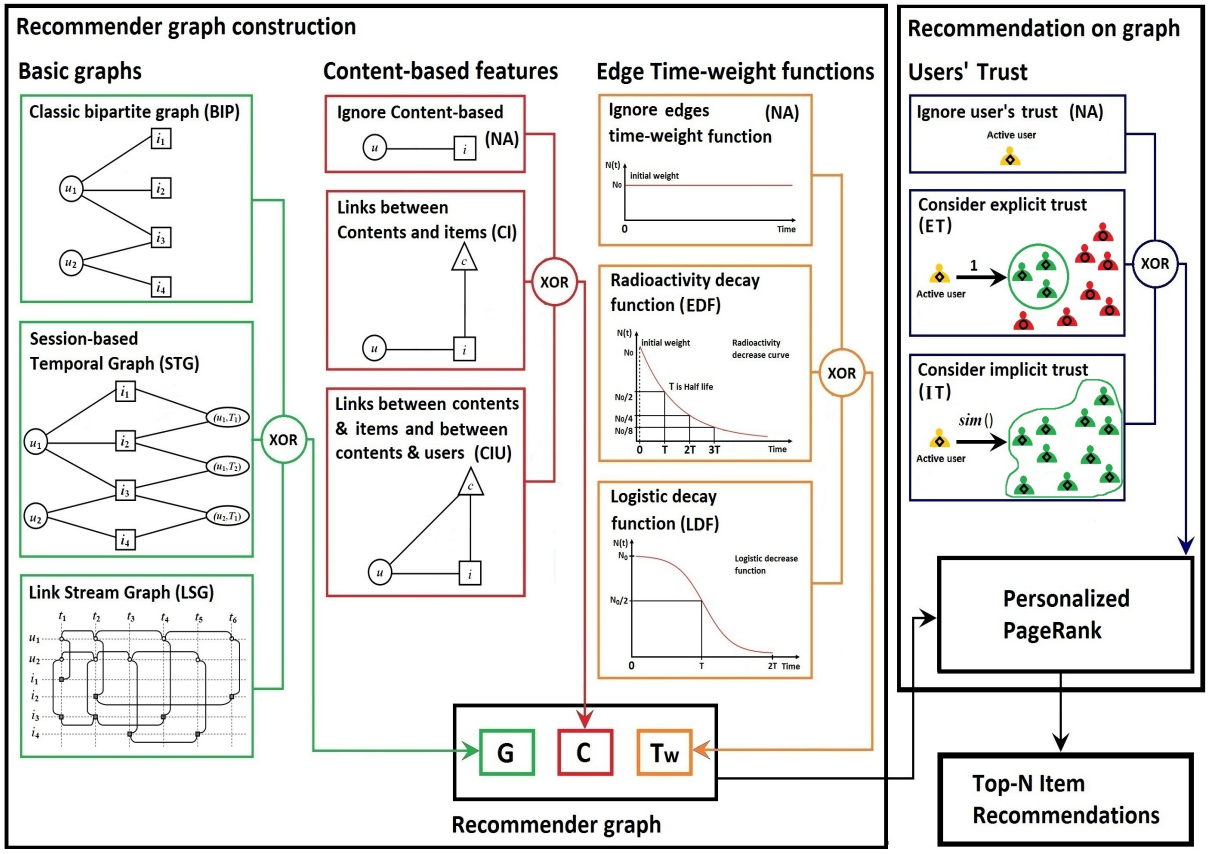


Figure 1: **The global architecture of GraFC2T2**, our general purpose graph-based recommender framework. Recommender graphs are built from three components: a basic graph that models user-item relations, content-based features that enrich basic graph, and link time-weight function that penalizes old edges, see Sections II and III. Then, we perform top-N recommendation over this graph using user trust and personalized PageRank, see Section IV.

watching or song listening), or another user-item relational event, depending on the application context. See Viard *et al.* (2016); Latapy *et al.* (2018) for a full description of the link stream formalism. In the following, we will illustrate definitions with the guiding example of Figure 2.

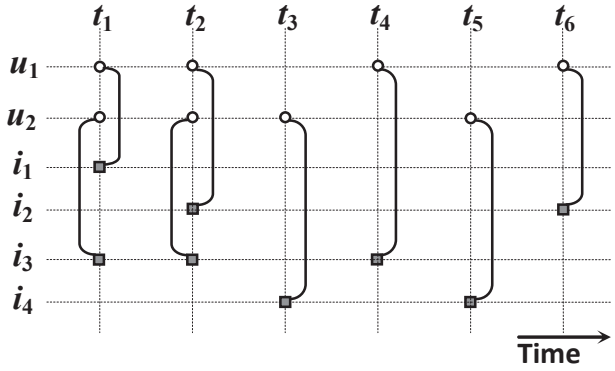


Figure 2: **Guiding example**: we consider the link stream $L = (T, U, I, E)$ in which the set of users is $U = \{u_1, u_2\}$, the set of items is $I = \{i_1, i_2, i_3, i_4\}$, the observation period is $T = [t_1, t_6]$, and $E = \{(t_1, u_1, i_1), (t_1, u_2, i_3), (t_2, u_1, i_2), (t_2, u_2, i_3), (t_3, u_2, i_4), (t_4, u_1, i_3), (t_5, u_2, i_4), (t_6, u_1, i_2)\}$. This means for instance that user u_1 was interested in item i_2 at time t_2.

2.1 Classical bipartite graph

We first consider the most classical recommender graph introduced in the literature (Huang *et al.*, 2004; Baluja *et al.*, 2008), that we denote by BIP. It is a directed bipartite graph (U, I, E') where U and I are the set of users and items defined above, and $E' \subseteq U \times I$ is the set of links defined by $E' = \{(u, i) : \exists t \in T, (t, u, i) \in E\}$. In other words, u is linked to i in BIP if user u was interested in item i during the observation period. Figure 3a displays the BIP graph for the guiding example.

2.2 Session-based temporal graph

In a first attempt to capture time information, we then consider Session-based Temporal Graphs proposed by Xiang *et al.* (2010), that we denote by STG.

This graph encodes time information using a set S of session nodes defined as follows. First, for a given Δ, the observation interval T is divided into $\frac{|T|}{\Delta}$ time slices $T_k = [(k-1) \cdot \Delta, k \cdot \Delta]$ of equal duration Δ. Then, S contains the couples (u, T_k) such that there exists a link (t, u, i) in E with $t \in T_k$. In other words, each user leads to a session node (u, T_k) in S for each time interval T_k during which this user was active.

This finally leads to the definition of STG as a tripartite graph (U, I, S, E'') with U, I, and S defined above, and $E'' = E' \cup \{((u, T_k), i) : \exists t \in T_k, (t, u, i) \in E\}$. In other words, we add to BIP the nodes in S, and a link between each session node (u, T_k) and the items selected by user u during time slice T_k. Figure 3b shows the STG representation for the guiding example.

Notice that in the original model Xiang *et al.* (2010), any link from u to i has a weight 1 and any link from i to u has a weight η, where η is a parameter. For simplicity, we do not consider this parameter here (or, equivalently, $\eta = 1$), but it may easily be added if needed.

2.3 Link stream graph

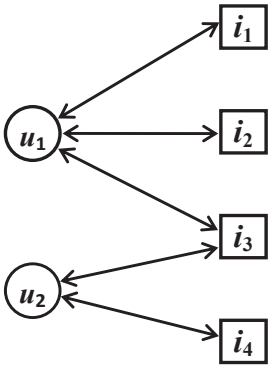
In order to capture time information while avoiding the drawbacks of choosing a time window size Δ like for STG, we introduce the following link stream graph, that we denote by LSG (Nzekon Nzeko'o *et al.*, 2019).

This graph is first defined by a set of nodes representing users and items over time: $\{(t, u) : \exists i, (t, u, i) \in E\} \cup \{(t, i) : \exists u, (t, u, i) \in E\}$. In other words, each user u is represented by the nodes (t, u) such that a link involves u in L a time t, and each item is represented similarly.

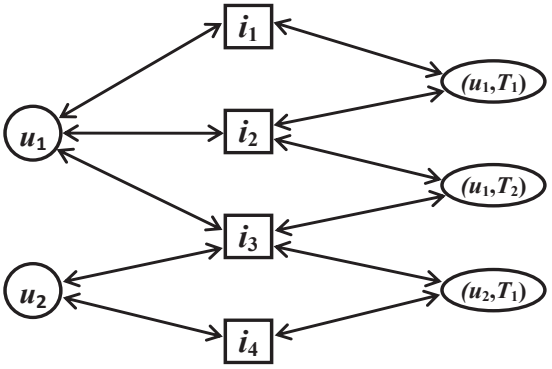
We then define the set of links $\{((t, u), (t, i)) : (t, u, i) \in E\} \cup \{((t, u), (t', u)) : \exists i, (t, u, i) \in E, t' = \min\{x : x > t \text{ and } \exists i', (x, u, i') \in E\} \cup \{((t, i), (t', i)) : \exists u, (t, u, i) \in E, t' = \min\{x : x > t \text{ and } \exists u', (x, u', i) \in E\}$. In other words, each user node (t, u) is linked to both the item nodes (t, i) such that $(t, u, i) \in E$ and to the next user node representing u. Item nodes are linked similarly. See Figure 3c for an illustration on our guiding example.

III ADDING CONTENT-BASED FEATURES AND TIME-WEIGHT FUNCTIONS

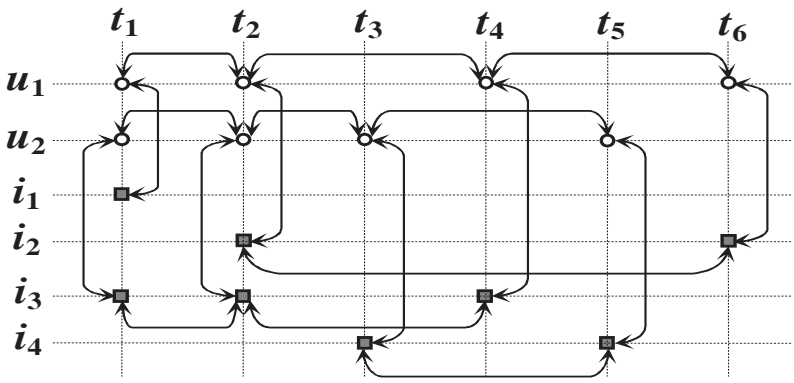
Once a basic recommender graph is built as explained in previous section, the GraFC2T2 framework adds elements to capture content-based and temporal features. Again, we propose several choices, and we present them below.



(a) Classical bipartite graph, BIP



(b) Session-based temporal graph, STG



(c) Link stream graph, LSG

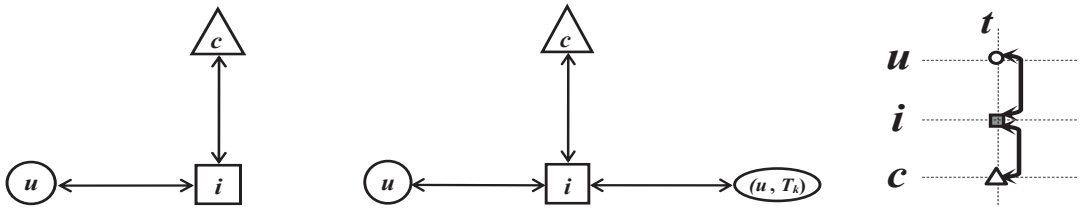
Figure 3: Classical bipartite graph, Session-based temporal graph and Link stream graph obtained from our guiding example. The weight of each edge is 1.

3.1 Content-based features

Let C be the set of all possible content-based features and let $g(i) \subseteq C$ be the subset of content-based features associated with item i, for any i. One element of $g(i)$ can be the category, the brand or the color of item i. Following the method proposed in (Nguyen *et al.*, 2008; Yu *et al.*, 2014; Nzekon Nzeko'o *et al.*, 2017), we model these features by content nodes that we link to item nodes in basic recommender graphs.

In the cases of BIP and STG, we add a content node c for each content-based feature c in C, and we link each item node i to the content node c for each c in $g(i)$. For LSG, we add a content node (t, c) for each (t, i) in the basic graph such that c is in $g(i)$, and we link (t, c) to (t, i). We call this inclusion of content-based features CI because it adds links only between content and item nodes. See Figure 4.

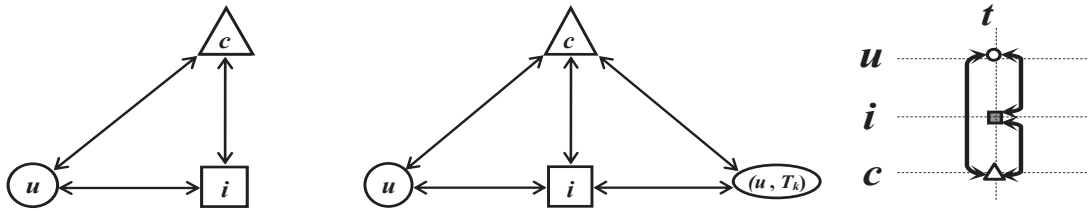
We also propose a strategy linking content nodes to both item and user nodes, that we call CIU. The idea is to link user nodes to the content nodes of the items they are interested in. Therefore, in addition to CI additions, CIU adds to BIP a link (u, c) between each user node u and content node c whenever there is an item node linked to both u and c; to STG a link between each session node (u, T_k) and content node c whenever there is an item node linked to both (u, T_k)



(a) BIP + CI (b) STG + CI (c) LSG + CI

Figure 4: Inclusion of nodes and links representing content-based features with the CI strategy, for each basic recommender graph.

and c; and to LSG a link between each user node (t, u) and content node (t, c) whenever there is an item node linked to both. See Figure 5.



(a) BIP + CIU (b) STG + CIU (c) LSG + CIU

Figure 5: Inclusion of nodes and links representing content-based features with the CIU strategy, for each basic recommender graph.

Compared to CI, the CIU method increases the influence of content-based features linked to items that the target user has already selected in the past. In other words, the CIU method do a better promotion of items that have the same features as the choices of the target user.

3.2 Time-weight functions

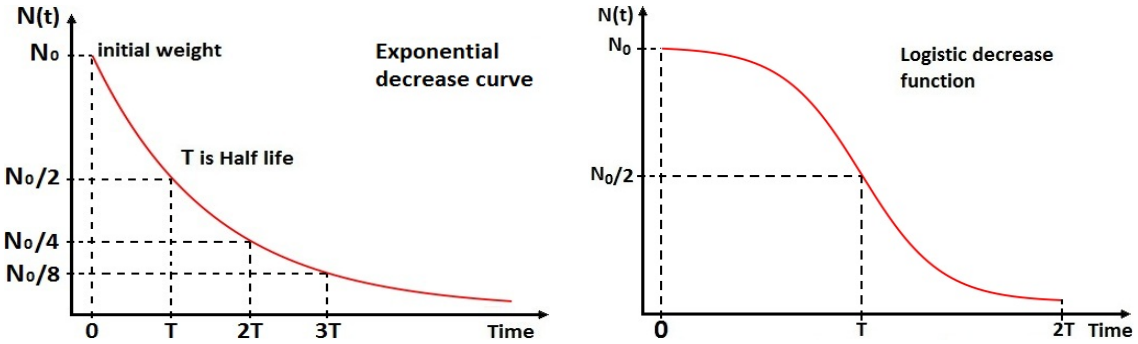
Until now, we modeled time information directly within the structure of STG and LSG graphs, but their edge weights give a static view of previous user interests. Since such interests evolve over time, as pointed out for instance by Ding and Li (2005), this is not sufficient. We therefore follow the methodology proposed in that paper, consisting in adding time-dependent weights to the links of recommender graphs.

The idea is to give a high weight to recent links, and to decrease this weight with their age: the weight at time t of any link (a, b) whose most recent appearance time is $t_e \leq t$, is of the form $w_t(a, b) = f(t - t_e) \cdot w(a, b)$, where $f()$ is a decay function. Many different decay functions may make sense, and we designed GraFC2T2 to make it easy to integrate those functions. We consider here the two following classical choices.

- Our first example is the exponential decay function (EDF) illustrated in Figure 6a: $f(x) = e^{-x \cdot \ln(2)/\tau_0}$, where τ_0 is the radioactivity half life; after a delay of τ_0, the link weight is divided by 2.

Volume: 5 - Year: 2019, DOI: 10.18713/JIMIS-300519-5-2

- We also consider the logistic decay function (LDF) illustrated in Figure 6b: $f(x) = 1-1/(e^{-K(x-\tau_0)}+1)$ where K is the steepness of the curve and τ_0 is the sigmoid midpoint; if $x = \tau_0$ then $f(x) = 0.5$.



(a) Exponential decay function, EDF (b) Logistic decay function, LDF

Figure 6: Edge time-weight functions.

IV RECOMMENDATION WITH PERSONALIZED PAGERANK AND TRUST

Once a recommender graph is built with a combination of choices proposed in previous sections, we are ready to perform top-N recommendation from this graph. We present below the personalized PageRank approach and an extension to include the concept of trust between users.

4.1 Personalized PageRank

Personalized PageRank algorithm is defined by Page *et al.* (1999) for node ranking in graphs so that nodes can be ranked efficiently in order of importance. The first application was on web pages, especially in the Google search engine. Then this algorithm has been widely used in recommender systems because of the good prediction quality obtained (Gori *et al.*, 2007; Kim and El Saddik, 2011; Şora, 2015).

Following this last observation, Xiang *et al.* (2010) proposed the Temporal Personalized Random Walk (TPRW) to compute recommendations on STG. It was defined to tackle temporal recommendation using the personalization idea of Haveliwala (2002), corresponding to the following formula:

$$PR = \alpha \cdot M \cdot PR + (1-\alpha) \cdot d \tag{1}$$

Where PR is PageRank vector that contains the importance of each node at the end of the propagation process that we want to compute; M is the transition matrix of the considered graph; α is the damping factor; and d is the personalization vector indicating which nodes the random walker will jump to after a restart. In other words, d allows to initialize the weight of source nodes. This process favors the recommendation of products that are close to source nodes: items close to source nodes with large weights in vector d, are favored (see below).

For a given user u at time t, we define the personalized temporal vector d as follows, depending on the type of basic graph:

- for BIP, the walker always restarts from u: $d(u) = 1$ and $d(v) = 0$ if $v \neq u$;
- for STG, the walker either restarts from u or from its most recent session node (u, T_k): $d(u) = \beta$, $d(u, T_k) = 1 - \beta$, $d(v) = 0$ if $v \neq u$ and $v \neq (u, T_k)$;

Volume: 5 - Year: 2019, DOI: 10.18713/JIMIS-300519-5-2

- for LSG, the walker always restarts from the most recent temporal node representing u, (t', u): $d(t', u) = 1$ and $d(t'', v) = 0$ if $(t'', v) \neq (t', u)$.

Then, we run PageRank over the recommender graph to compute the interest of each user u for item i at time t, and output the N items with highest interest (in LSG, the interest for item i is the sum of interests for (t, i), for all t).

4.2 Trust integration

Trust relationships are interesting for improving recommendation, especially for cold users and cold items (users or items for which very limited information is available). Some systems incorporate trust information explicitly specified by users (Jamali and Ester, 2009; Guo *et al.*, 2017; Pan *et al.*, 2017), but since such explicit information is rarely available, several approaches infer implicit trust (Pitsilis and Marshall, 2004; Papagelis *et al.*, 2005; Hwang and Chen, 2007; Lathia *et al.*, 2008). In this section, we describe how to include these both types of trust in our framework.

We assume trust relationships are modeled for each user u by a set TR_u of users trusted by u, and that $trust(u, v)$ gives the trust level of u for all v in TR_u, with $\sum_{v \in TR_u} trust(u, v) = 1$. We denote the method where explicit trust relationships are given by ET (Explicit Trust). We also use an implicit trust metric based on similarity measures as proposed by Papagelis *et al.* (2005) and denote this method by IT (Implicit Trust). In this method, $TR_u = U$ is the set of all users, and $trust(u, v) = |I_u \cap I_v|/|I_u \cup I_v|$ is the Jaccard similarity between users u and v. Note that other similarity measures may be used, such as cosine index.

We then update the personalized temporal vector d definition as follows (with the same notations as in the initial definition above):

- for BIP, $d(u) = 1-\gamma$, $d(v) = (\gamma \cdot trust(u, v))/|TR_u|$ if $v \in TR_u$ and $d(v) = 0$ otherwise;
- for STG, we share the jumping probability β between u and its trusted users: $d(u) = \beta \cdot (1 - \gamma)$, $d(v) = (\beta \cdot \gamma \cdot trust(u, v))/|TR_u|$ for all $v \in TR_u$; and we share the probability $1 - \beta$ between u most recent session node and the ones of trusted users: $d(u, T_k) = (1 - \beta) \cdot (1 - \gamma)$, $d(v, T_v) = (1 - \beta) \cdot \gamma \cdot trust(u, v)/|TR_u|$ where $v \in TR_u$ and (v, T_v) is the most recent session node of v. We set all other entries of d to 0.
- for LSG, $d(t_k, u) = 1 - \gamma$, $d(t_v, v) = \gamma \cdot trust(u, v)/|TR_u|$ if $v \in TR_u$ and (t_v, v) is the most recent node representing v, and all other entries of d are 0.

V EXPERIMENTAL SETUP

Previous sections defined our general graph-based framework GraFC2T2, that gives wide levels of freedom for selecting and combining its various components into a top-N recommender system. These component capture several kinds of side information, in particular content-based, temporal, and trust features. In this section, we describe an experimental setup that we use in the next section to evaluate our framework. This setup consists in two real-world datasets, an evaluation method relying on three metrics, and a parameter selection method to optimize results.

5.1 Datasets

We use publicly available datasets extracted from product reviews Epinions and Ciao[3] (Tang *et al.*, 2012), where users can write reviews and give their opinions on a wide category of products like Home, Health, Computers and Media. We model each dataset as a set of review tuples (u, i, c, r, t) meaning that user u has assigned the rating $r \in \{0, 1, 2, 3, 4, 5\}$ to item i at time t, with c being a content-based feature of item i. The explicit trust networks of these datasets are considered such that for each user u, the set TR_u is given for the ET method. Table 1 provides key information on these datasets: start and end dates, as well as numbers of distinct users, items, content-based features, ratings, explicit trust relationships, ratings density and trust relationships density.

	Start date	**End date**	$\|U\|$	$\|I\|$	$\|C\|$	**Ratings**	**Trust**	δ_r	δ_t
Epinions	2010-01-01	2010-12-31	1 843	15 899	24	17 722	4 867	0.06%	0.14%
Ciao	2007-01-01	2010-12-31	879	6 005	6	8 109	23 121	0.15%	3.00%

Table 1: Basic data statistics

Since our framework does not use ratings but only positive links between users and items, we discard all tuples such that the rating it contains is lower than 2.5 or the average rating of involved user.

5.2 Evaluation

Evaluating recommender systems is a difficult task. In this paper, we use three classical metrics for top-N recommendations: F1-score (F1), Hit Ratio (HR) and Mean Average Precision (MAP) (Baeza-Yates and Ribeiro-Neto, 2011). Higher values of these metrics indicate better recommendation performance.

F1-score is a trade-off between ranking precision and recall such that optimizing F1-score is more robust than optimizing precision or recall. Precision is the fraction of good recommendations over all recommended items and recall is the fraction of good recommendations over all relevant items to recommend. For one user u,

$$Precision = \frac{hit_N(u)}{N}, \tag{2}$$

$$Recall = \frac{hit_N(u)}{I_{new}(u)} \tag{3}$$

and

$$F1 = 2 \cdot \frac{Precision \times Recall}{Precision + Recall} = 2 \cdot \frac{hit_N(u)}{I_{new}(u) + N}, \tag{4}$$

where N is the length of recommendation list, $hit_N(u)$ denotes the number of good recommendations to u in the top-N items and $I_{new}(u)$ is the set of new items to recommend to u. For all users the equation of F1-score is:

$$F@N = \frac{\sum_{u \in U} 2 \times hit_N(u)}{\sum_{u \in U} (I_{new}(u) + N)}. \tag{5}$$

[3] https://www.cse.msu.edu/~tangjili/trust.html

Hit Ratio is the fraction of users to whom the recommender system has made at least one good recommendation over all users:

$$H@N = \frac{\sum_{u \in U}(hit_N(u) > 0)}{|U|}. \tag{6}$$

Mean Average Precision considers the order of items in the top-N recommendation in order to give better evaluation scores to results that recommend better items first:

$$M@N = \frac{\sum_{u \in U} AP_N(u)}{|U|}, \tag{7}$$

where

$$AP_N(u) = \frac{1}{hit_N(u)} \sum_{k=1}^{N} \frac{hit_k(u)}{k} \times h(k) \tag{8}$$

is the average precision of top-N recommendations done to user u and $h(k) = 1$ if the k-th recommended item is a good recommendation and 0 otherwise.

These metrics evaluate a given top-N recommendation. Since we actually can't perform recommendations on live users, we perform evaluation on past data described above. Following the classical method established by previous works (Li and Tang, 2008; Lathia *et al.*, 2009; Campos *et al.*, 2014; Nzekon Nzeko'o *et al.*, 2017), we partition data according to $k + 1$ time windows of equal duration, and we use them as follow. For each of the k first slices:

- we build recommender graphs that correspond to data of this slice and all previous slices (training set),
- we compute top-N recommendations for users who have selected at least one new item in the next time slice (test set),
- we compute for each evaluation metric M the numerator M_{num_k} and the denominator M_{deno_k} of its definition, given above.

Once we have the values of M_{num_k} and M_{deno_k} of each of the k first windows, we combine them into the Time Averaged (TA) value of the metric under concern:

$$TA(M) = \frac{\sum_k M_{num_k}}{\sum_k M_{deno_k}}. \tag{9}$$

This leads to a time-averaged value of F1-score, Hit ratio and MAP, that we all use for evaluation. Indeed, evaluation metrics can be in disagreement (Gunawardana and Shani, 2009), and so using several metrics is essential to obtain accurate insight on result quality.

In our experiments, we set k to 7 in order to have large enough data slices and meaningful averages. We consider exploring the role of this parameter, as well as the use of more advanced evaluation metrics, as future work.

5.3 Parameter estimation

For each basic graph type, GraFC2T2 defines and implements 27 possible combinations of side information modelings, see Figure 1. Our priority is to explore the behaviors and differences of all these variants, and so we did our best to keep the number of other parameters reasonable. Still, the different version of recommender systems encoded in GraFC2T2 call for several parameter selection.

Exhaustive search for the best values is out of reach, and many subtle techniques exist to explore the parameter space in search for good values. Since this search is not the focus of this paper, we use a simple approach called Randomized Search Cross-Validation (more advanced methods may easily be included in our framework, though) (Bergstra and Bengio, 2012). This method randomly selects parameter values in a predefined set of possible values, usually designed to span well the whole set of values. Here, we use 50 such random settings, sampled in the set defined by Table 2.

	Parameter meaning	**Predefined values**
Δ	STG session duration	7, 30, 90, 180, 365, 540, 730 days
β	STG long-term preference	0.1, 0.3, 0.5, 0.7, 0.9
τ_0	half life of EDF and LDF	7, 30, 90, 180, 365, 540, 730 days
K	decay slope of LDF	0.1, 0.5, 1, 5, 10, 50, 100
γ	influence of trusted users	0.05, 0.1, 0.15, 0.3, 0.5, 0.7, 0.9
α	damping factor for PageRank	0.05, 0.1, 0.15, 0.3, 0.5, 0.7, 0.9

Table 2: Predefined values of parameters

VI EXPERIMENTAL RESULTS

This section presents extensive experimentations on our GraFC2T2 framework, in order to study its performances in practice, to explore the contribution of each side information in these cases, and to compare obtained results to state-of-the-art recommender systems.

6.1 Performances of GraFC2T2

Table 3 presents the results we obtained for Top-10 item recommendation for Epinions and Ciao datasets. We chose $N = 10$ as for instance Deshpande and Karypis (2004), Xiang *et al.* (2010) and Bernardes *et al.* (2015), and other values we tested gave similar results as one may see in the appendix (Section B). In these tables, each column corresponds to a metric and a basic recommender graph, and each row corresponds to a combination of side information added to this recommender graph. Each cell contains the value of the evaluation metric for the recommender graph made of basic graph in column and side information in row. White color of cell corresponds to the best result and dark color indicates lower performance.

We summarize the insight obtained from these results in Table 4. For each basic recommender graph (vertically) and each evaluation metric (horizontally), we selected the three recommender graphs that achieve the best performances and we display on the corresponding row the performances obtained on the basic graph (without side information), the best obtained performances (with side information), the improvement percentage, and the name of the corresponding version of recommender graph with side information.

All best improvements thanks to side information in GraFC2T2 are at least 46% for Epinions and at least 41% for Ciao. Table 4 also shows that the best combination of side information

EPINIONS	F1@10			HR@10			MAP@10		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	2.18	2.0	1.14	5.17	4.77	4.24	2.23	2.17	1.71
ET	1.81	1.86	1.14	4.64	4.64	4.11	2.04	2.07	1.66
IT	2.17	2.42	1.61	5.44	5.44	5.31	2.29	2.34	2.24
EDF	3.74	3.32	2.25	6.1	5.97	5.7	2.31	2.35	2.32
LDF	3.16	2.63	2.26	5.97	5.31	5.97	2.24	2.09	2.54
CI	2.53	3.0	0.86	5.97	6.23	3.32	2.48	2.73	1.74
CIU	3.29	3.51	0.66	6.37	6.63	2.79	2.66	2.88	1.66
EDF-ET	2.77	3.32	1.77	5.44	5.84	5.04	2.13	1.97	2.09
EDF-IT	3.88	4.43	2.74	7.03	7.16	6.37	2.64	2.57	2.42
LDF-ET	2.12	2.63	1.58	4.91	5.31	4.77	2.03	2.1	2.13
LDF-IT	3.1	2.77	3.29	6.1	6.5	7.16	2.98	2.98	3.01
CI-ET	2.44	3.03	0.92	5.84	6.23	3.45	2.28	2.67	1.74
CI-IT	3.12	3.14	1.77	6.37	6.37	5.44	2.49	2.66	2.15
CIU-ET	3.03	3.51	0.66	6.1	6.63	2.79	2.43	2.9	1.66
CIU-IT	3.89	3.72	1.99	7.03	7.03	5.44	2.79	2.88	2.25
CI-EDF	4.88	4.84	0.91	7.69	6.9	3.32	3.06	2.73	1.74
CI-LDF	4.91	3.56	1.1	7.03	6.5	3.98	2.61	2.67	1.75
CIU-EDF	4.51	6.48	0.7	7.69	7.69	2.92	3.23	3.03	1.66
CIU-LDF	5.29	4.49	0.9	7.16	6.76	3.58	2.89	2.85	1.72
CI-EDF-ET	3.84	4.73	0.98	6.63	6.76	3.45	2.44	2.46	1.74
CI-EDF-IT	4.85	4.47	1.69	7.43	6.76	5.31	2.8	2.7	2.17
CI-LDF-ET	3.02	3.39	0.92	5.97	6.37	3.45	2.28	2.66	1.74
CI-LDF-IT	3.81	4.02	3.41	6.76	6.63	7.29	3.0	3.0	2.66
CIU-EDF-ET	4.75	6.13	0.7	7.16	7.43	2.92	2.64	2.95	1.66
CIU-EDF-IT	6.34	7.66	1.99	7.82	7.96	5.44	3.32	3.18	2.27
CIU-LDF-ET	4.06	3.78	0.7	6.63	6.63	2.92	2.48	2.78	1.66
CIU-LDF-IT	5.69	4.49	3.68	7.82	7.03	7.03	3.18	3.07	3.17

CIAO	F1@10			HR@10			MAP@10		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.18	1.48	1.45	5.26	5.63	6.53	1.9	1.99	2.24
ET	1.08	1.44	1.5	5.08	5.44	6.72	1.74	1.9	2.17
IT	2.18	1.92	2.25	7.62	7.62	7.26	2.39	2.39	2.28
EDF	1.63	1.7	2.0	6.35	5.99	7.44	2.03	2.26	2.34
LDF	2.02	1.74	3.27	7.26	7.44	9.26	2.63	2.84	3.51
CI	1.25	2.14	1.25	6.53	6.35	5.26	2.04	2.14	1.7
CIU	2.38	4.56	1.24	7.08	8.53	4.9	2.39	3.01	1.48
EDF-ET	1.17	1.47	1.41	5.26	5.81	6.53	1.68	2.1	2.18
EDF-IT	2.13	3.08	2.37	9.98	9.44	7.26	3.04	2.84	2.32
LDF-ET	1.18	1.49	1.5	5.44	5.63	6.72	1.81	2.01	2.18
LDF-IT	2.66	2.76	2.76	8.53	8.53	8.71	2.91	2.96	2.66
CI-ET	1.39	1.66	1.51	5.99	6.17	5.63	1.97	2.06	1.72
CI-IT	2.41	2.25	2.84	7.8	7.8	8.53	2.47	2.42	2.43
CIU-ET	2.02	4.0	1.31	6.9	8.53	5.08	2.27	3.12	1.5
CIU-IT	2.76	4.21	2.96	8.53	8.53	8.35	2.82	2.96	2.44
CI-EDF	2.58	3.15	1.66	7.62	8.17	6.17	2.33	2.76	1.77
CI-LDF	3.38	2.91	2.37	8.35	8.89	7.62	2.87	2.86	2.85
CIU-EDF	3.46	4.46	1.44	8.71	9.26	5.44	3.24	3.29	1.65
CIU-LDF	7.74	5.79	2.11	9.98	9.8	6.9	3.46	3.31	2.68
CI-EDF-ET	1.86	2.11	1.63	7.26	6.72	5.99	2.27	2.37	1.81
CI-EDF-IT	3.42	3.54	2.9	9.8	9.8	7.99	3.09	3.14	2.46
CI-LDF-ET	1.66	2.67	1.74	6.35	7.08	5.81	1.98	2.12	1.77
CI-LDF-IT	4.56	4.89	2.64	10.7	11.3	8.53	3.19	3.16	3.16
CIU-EDF-ET	2.69	4.79	1.53	8.53	9.26	5.63	2.76	3.09	1.65
CIU-EDF-IT	4.62	5.1	2.88	10.3	10.5	8.35	3.32	3.37	2.38
CIU-LDF-ET	2.73	6.42	1.45	8.35	9.98	5.44	2.29	3.34	1.64
CIU-LDF-IT	5.07	6.11	2.51	11.1	11.1	9.07	3.34	3.35	3.18

Table 3: Epinions and Ciao - Performance with optimal settings. Each cell contains the value of an evaluation metric for the recommender graph made of basic graph in column and side information in row. White color of cell corresponds to the best result and dark color indicates lower performance

Epinions Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@10	1	2.18	6.34	190%	CIU-EDF-IT	2.0	7.66	282%	CIU-EDF-IT	1.14	3.68	221%	CIU-LDF-IT
	2	2.18	5.69	160%	CIU-LDF-IT	2.0	6.48	223%	CIU-EDF	1.14	3.41	197%	CI-LDF-IT
	3	2.18	5.29	142%	CIU-LDF	2.0	6.13	206%	CIU-EDF-ET	1.14	3.29	187%	LDF-IT
H@10	1	5.17	7.82	51%	CIU-EDF-IT	4.77	7.96	66%	CIU-EDF-IT	4.24	7.29	71%	CI-LDF-IT
	2	5.17	7.82	51%	CIU-LDF-IT	4.77	7.69	61%	CIU-EDF	4.24	7.16	68%	LDF-IT
	3	5.17	7.69	48%	CI-EDF	4.77	7.43	55%	CIU-EDF-ET	4.24	7.03	65%	CIU-LDF-IT
M@10	1	2.23	3.32	48%	CIU-EDF-IT	2.17	3.18	46%	CIU-EDF-IT	1.71	3.17	85%	CIU-LDF-IT
	2	2.23	3.23	45%	CIU-EDF	2.17	3.07	41%	CIU-LDF-IT	1.71	3.01	76%	LDF-IT
	3	2.23	3.18	42%	CIU-LDF-IT	2.17	3.03	39%	CIU-EDF	1.71	2.66	55%	CI-LDF-IT

Ciao Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@10	1	1.18	7.74	556%	CIU-LDF	1.48	6.42	332%	CIU-LDF-ET	1.45	3.27	125%	LDF
	2	1.18	5.07	330%	CIU-LDF-IT	1.48	6.11	311%	CIU-LDF-IT	1.45	2.96	104%	CIU-IT
	3	1.18	4.62	291%	CIU-EDF-IT	1.48	5.79	290%	CIU-LDF	1.45	2.9	99%	CI-EDF-IT
H@10	1	5.26	11.1	110%	CIU-LDF-IT	5.63	11.3	100%	CI-LDF-IT	6.53	9.26	41%	LDF
	2	5.26	10.7	103%	CI-LDF-IT	5.63	11.1	96%	CIU-LDF-IT	6.53	9.07	38%	CIU-LDF-IT
	3	5.26	10.3	96%	CIU-EDF-IT	5.63	10.5	87%	CIU-EDF-IT	6.53	8.71	33%	LDF-IT
M@10	1	1.9	3.46	82%	CIU-LDF	1.99	3.37	69%	CIU-EDF-IT	2.24	3.51	57%	LDF
	2	1.9	3.34	76%	CIU-LDF-IT	1.99	3.35	68%	CIU-LDF-IT	2.24	3.18	42%	CIU-LDF-IT
	3	1.9	3.32	74%	CIU-EDF-IT	1.99	3.34	67%	CIU-LDF-ET	2.24	3.16	41%	CI-LDF-IT

Table 4: Best recommender graphs - Comparison of the three best recommender graph combinations with the associated basic graph. We display the obtained improvement percentage.

for Epinions is CIU-EDF-IT for BIP and STG basic graphs and CIU-LDF-IT for LSG basic graph. For Ciao, good results are obtained with CIU-LDF-IT for all basic graphs. These results clearly confirm the relevance of graphs extended simultaneously with content, time and trust information.

6.2 Impact of side information

We now give details on the impact of side information and their combination in GraFC2T2. This is context dependent, as observed behaviors vary with datasets; one may however easily test the GraFC2T2 framework with his/her own datasets and discover the best choices for the case under concern. The discussion provided here is mostly an illustration of this.

When we consider the basic graphs with no side information, in the case of Epinions, BIP gives the best results for all evaluation metrics. Instead, LSG gives the best Hit ratio and MAP, while STG gives the best F1-score in the case of Ciao.

If we include only one kind of side information, we observe that explicit trust (ET) does not improve the results, but implicit trust (IT) does for all basic graphs. The insertion of time-weight always produces improvements. Finally, content-based features increase performances for BIP and STG but not for LSG. For Epinions, the best graph with one kind of side information is BIP-EDF in F1-score and STG-CIU in Hit ratio and MAP. In Ciao, the best one is LSG-LDF in Hit ratio and MAP, and STG-CIU is the best in F1-score. This shows that the impact of a unique kind of side information highly depends on the basic graph and on the data.

Recommendations using two kinds of side information perform significantly better than with only one kind of side information. For instance, in the Epinions case, performances increase

from 3.74% to 6.48% in F1-score, from 6.63% to 7.69% in Hit ratio and from 2.88% to 3.23% in MAP. Combining time-weight with implicit trust performs better than time-weight and trust taken separately. Similarly, combining content-based features with implicit trust is better than content-based features or trust taken separately, but generally less interesting than combining time-weight and implicit trust. Combining content-based features and time-weight usually produces better improvements for BIP and STG but no improvement for LSG. In Epinions, BIP-CI-EDF and BIP-CIU-EDF perform best. In Ciao, BIP-CIU-LDF is always better. This confirms the relevance of graphs that integrate content-based features and time, like time-weight content-based STG proposed by Nzekon Nzeko'o *et al.* (2017).

Using three kinds of side information does not greatly improve the best performances achieved with two kinds of side information. For instance, in Epinions, the performances increase from 6.48 to 7.66% in F1-score, from 7.69 to 7.96% in Hit ratio and 3.23 to 3.32% in MAP. Nevertheless, Table 4 shows that recommender graphs with three kinds of side information are by far the most frequent among the best ones. For this reason, we recommend the use of content-based, time and trust information simultaneously in order to increase the chances to achieve good results.

6.3 Best values of parameters

In this section, we focus only on recommender graphs with CIU-EDF-IT and CIU-LDF-IT combination that are most common in the best performance in Table 4. We have made the following observations:

- In Epinions dataset, for the combination CIU-EDF-IT, $\Delta = 7$, $\beta = 0.5$, $\tau_0 = 90$ for BIP and STG and 180 for LSG, $\gamma \in \{0.15, 0.3\}$ for BIP and STG and 0.9 for LSG, and $\alpha = 0.9$. For the combination CIU-LDF-IT, $\Delta = 365$, $\beta = 0.7$, $\tau_0 \in \{30, 90\}$ for BIP and STG and 7 for LSG, $K = 0.5$ for BIP, 100 for STG and 5 for LSG, $\gamma \in \{0.1, 0.15\}$ for BIP and STG and 0.9 for LSG, and $\alpha \in \{0.7, 0.9\}$;
- In Ciao dataset, for the combination CIU-EDF-IT, $\Delta = 180$, $\beta = 0.3$, $\tau_0 = 180$, $\gamma = 0.9$ and $\alpha = 0.9$. For the combination CIU-LDF-IT, $\Delta = 540$, $\beta = 0.1$, $\tau_0 = 365$ for BIP and STG and 180 for LSG, $K = 10$ for BIP and STG and 100 for LSG, $\gamma \in \{0.7, 0.9\}$, and $\alpha = 0.9$;

The values of these parameters indicate that in Epinions, the weights of the data used (edge weights) decrease faster than in Ciao; τ_0 is small in Epinions $\{7, 30, 90\}$ and is larger in Ciao $\{180, 365\}$. Regarding trust, γ is still high in Ciao $\{0.7, 0.9\}$ and is smaller in Epinions $\{0.1, 0.15, 0.3\}$ which shows that the influence of implicit trust is more important in Ciao. However, this influence must always be great for the graph LSG $\{0.9\}$ in all datasets.

6.4 Comparison with state-of-the-art systems without side information

We now compare the performances of GraFC2T2 with those of some state-of-the-art top-N recommender systems that don't take into account side information. The considered models are: Most-Popular-Item (MPI) that computes the ranking score of an item by its popularity; the ranking oriented collaborative filtering, user-based (UBCF) and item-based (IBCF) collaborative filtering (Karypis, 2001; McLaughlin and Herlocker, 2004); some recommender systems for positive implicit feedback scenarios, Bayesian Personalized Ranking (BPR) (Rendle *et al.*, 2009), Sparse linear methods for top-N recommender systems (SLIM) (Ning and Karypis, 2011), collaborative less-is-more filtering (CLiMF) (Shi *et al.*, 2012) and Matrix factorization with Alternating Least Squares (ALS) (Hu *et al.*, 2008).

We use Randomized Search Cross-Validation to have good performances of the considered recommender systems. For UBCF and IBCF models, 10 settings are generated such that the neighborhood size $k \in \{10, 20, 30, 40, 50, 80, 100, 150, 200, 500\}$. For BPR, SLIM, CLIMF and ALS models, 50 settings are generated such that the number of latent factors $l \in \{10, 20, 30, 50, 100, 200, 500\}$, learning rate and all regularization bias are taken in $\{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05\}$.

Table 5 presents the best results obtained for these recommender systems that don't take into account side information and those obtained with our framework. This shows that GraFC2T2 outperforms these systems and illustrates the relevance of a general framework in which various kinds of side information can be added to improve recommendations.

		MPI	UBCF	IBCF	BPR	SLIM	CLIMF	ALS	GraFC2T2
Epinions	F@10	1.79	0.30	0.70	0.15	0.82	1.97	2.27	**7.66**
	H@10	4.91	1.46	2.79	0.80	2.92	5.17	4.91	**7.96**
	M@10	2.07	0.61	1.29	0.45	1.16	2.15	2.26	**3.32**
Ciao	F@10	2.26	0.31	0.94	0.22	1.49	3.38	2.10	**7.74**
	H@10	7.62	1.63	4.17	1.27	5.08	8.71	6.90	**11.3**
	M@10	2.62	0.59	1.65	0.56	2.09	3.06	2.46	**3.51**

Table 5: Experiment results on Epinions and Ciao datasets for Top-10. Performances are given in percentage and best ones are highlighted in bold.

A future step in our research is to compare the results obtained by GraFC2T2 to those produce by state-of-the-art systems that include side information. We have already observed that the results produced by GraFC2T2 are comparable to those presented by Xiao *et al.* (2017) where both item and social visibilities are modeled. Moreover, we have also made a comparison with Trust aware Denoising Auto Encoder (TDAE) technique based on deep learning (Pan *et al.*, 2017). The results for Epinions (M@10 = 1.32%) and Ciao (M@10 = 3.07%) confirm the relevance of GraFC2T2.

Notice that the most basic, non-personalized approach MPI is able to achieve better results compared to BPR, SLIM, UBCF and IBCF. This indicates that users tend to consume popular items. This is not the first work in which MPI is better than BPR or other matrix factorization models, Zhao *et al.* (2014) and Guo *et al.* (2017) have made the same observation.

VII RELATED WORK

As we already said, many contributions improve collaborative filtering (CF) recommender systems with the inclusion of side information, and we used several ideas proposed in these previous works. In the rest of this section, we shortly review key related references.

7.1 Trust-based recommender systems

CF usually suffers from data sparsity and cold start problems, which may be solved in part with user trust. For instance, Papagelis *et al.* (2005) used trust inference by transitive associations between users in a social network. Ma *et al.* (2017) use explicit trust and distrust to improve clustering-based CF recommendation, while Guo *et al.* (2014) merge ratings of trusted neighbors to infer probable preferences of other users, and identify similar users for item recommendations.

In some cases, trust can be explicitly provided by users, as do Massa and Avesani (2007), but in other ones, this information is not given and it can be inferred from user behaviors. For example, (Papagelis *et al.*, 2005) use Pearson correlation to compute implicit trust using ratings dataset and in cases where there is only implicit data, measure like Jaccard and Cosine can be used. In other works, trust enhancement is done by trust propagation on trust network where the weight of an link (u, u') is the trust of u to u' Deng *et al.* (2014).

Note that work on influencers can also be considered here, as there is a trust relationship between influencers and their followers (Liu *et al.*, 2015; Grafström *et al.*, 2018). Our framework is able to integrate the impact of influencers in the same way as trust between users. The main difference is who influences who and how much. Once you have the answers to these questions, the customization of PageRank is done according to these answers. The impact of influencers or influencer-based recommendation is not studied in this work, but it is a good issue for future work.

The concept of influence is a good example of other side information that may be included in our system (Liu *et al.*, 2015; Grafström *et al.*, 2018). Similarly to trust (although these two concepts are different) influence may be used to customize PageRank, once it is correctly quantified. For instance, influence may be seen as a trust relationship between influencers and their followers.

7.2 Time aware recommender systems

Most recommender systems that take temporal aspects into account are based on concept drift: older information is less important than recent information for predicting future user purchases. For this reason, Ding and Li (2005) proposed the use of the time-weight decay functions we used in this paper, in order to assign greater weight to the most recent ratings in similarity computations. In addition, Gaillard and Renders (2015) propose a incremental matrix completion method, that automatically allows the factors related to both users and items to adapt "on-line" to concept drift hypothesis. Going further, Liu *et al.* (2010) propose an online incremental CF in which a decay function is used for similarity computations and another one is used for rating prediction. Time-weight functions are also used in other studies as by Koren (2009); Karahodža *et al.* (2015) and Nzekon Nzeko'o *et al.* (2017).

Other approaches to concept drift assume that the importance of information used for recommendations is ephemeral, e.g. Lathia *et al.* (2009) divide time into slices and use data only within a single slice. Such recommender systems therefore focus on user short-term preferences. It however seems that some preferences are stable and persist over time, and so that old information should also be included. For this reason, some works (Xiang *et al.*, 2010; Li *et al.*, 2007) capture both short-term preferences and long-term preferences and combine them in the recommendation process. For example, Xiang *et al.* (2010) propose STG to incorporate temporal aspects by separately modeling long-term preferences and short-term preferences within a graph model.

7.3 Content-based recommender systems

These systems aim at recommending items similar to the ones the user liked in the past. A way to achieve this, developed by Lops *et al.* (2011), is to match features associated to user preferences with those of items. Then, recommendation is performed in three steps: extracting relevant features from items, build user preference profiles based on item features, and finally select new items that fit user preferences. This approach is used in several domains such as recommendation of books (Mooney and Roy, 2000) and recommendation of web pages (Pazzani *et al.*, 1996).

Using content-based features may improve CF techniques by allowing more details on user favorite item features and increase the possibility to reach items that have not been selected in the past by other users. Some works indeed show that these hybrid recommender systems solve weaknesses of both approaches (Balabanović and Shoham, 1997; Basu *et al.*, 1998; Burke, 2002).

Recent work on content-based approaches are dedicated to the Social Book Search (SBS). The SBS Lab investigates book search in scenarios where users search with more than just a query, and look for more than objective metadata. It has two tracks. The first one is a Suggestion Track aiming at developing test collections for evaluating ranking effectiveness of book retrieval and recommender systems. The second one is an Interactive Track aimed at developing user interfaces that support users through each stage during complex search tasks and to investigate how users exploit professional metadata and user-generated content (Koolen *et al.*, 2015).

7.4 Graph-based recommender systems

The simplest graph-based recommender system rely on the classical bipartite graph (BIP) in which only user-item links are used. Most used algorithms are based on random walk (Baluja *et al.*, 2008), like Injected Preference Fusion (Xiang *et al.*, 2010) and PageRank which is used in this paper; they compute a probability to reach items from the user under concern, and recommend the ones with highest probability.

Graph-based systems may be seen as CF systems, and so one may use the same idea as in hybrid recommender systems to improve them (Burke, 2002). Nguyen *et al.* (2008) achieve this by adding a third node type: content nodes. The resulting graph ignores temporal aspects, though. To improve this, Yu *et al.* (2014) propose the Topic-STG which incorporate content-based features and the temporal dynamic of STG. However these graphs handle each link regardless of its age, which contradicts the concept drift assumption. This is why we propose the Time-weight and content-based STG (Nzekon Nzeko'o *et al.*, 2017), where old links have a lower weight than recent ones. Up to our konwledge, none of these graph-based works takes advantage of content-based, time and trust information simultaneously.

We note that, despite the fact that recommender graphs are not much studied compared to model-based techniques such as matrix factorization or neural networks, they remain relevant. For example Pixie recommender system proposed by Eksombatchai *et al.* (2018) is the recent scalable graph-based real-time system developed and deployed at Pinterest. Given a set of user-specific pins as a query, Pixie selects in real-time from billions of possible pins that are most related to the query. To generate recommendations, Eksombatchai et al. develop Pixie Random Walk algorithm that uses the Pinterest object graph of 3 billion nodes and 17 billion edges. This has been made possible thanks to the technological evolution of Random Access Memories.

CONCLUSION

Our main goal with this paper was to show that including several side information improves the quality of recommender graphs built for top-N recommendation task. For this purpose, we designed and implemented GraFC2T2, a recommender graph framework which makes it easy to explore various approaches for modeling and combining many features of interests for recommendation. In particular, GraFC2T2 extends classical bipartite graphs, session-based temporal graphs and link stream graphs by integrating content-based features, time-weight functions, and user trust into a personalized PageRank system.

The experiments we conducted on Epinions and Ciao datasets with F1-score, Hit ratio and MAP evaluation metrics show that best performances are always reached by graphs that integrate at least two side information and that graphs with time-weight always outperform the others. The resulting improvements are of at least 41%. Moreover, comparison with state-of-the-art matrix factorization and classical user-based and item-based collaborative filtering methods confirms the relevance of GraFC2T2 framework for top-N recommendation. Good improvements obtained in recommender graphs by integration of side information do not guarantee such improvement for other types of recommender systems such as matrix factorization and neural network. We therefore consider inclusion of content-based, time and trust information simultaneously in such system as a key perspective.

References

Adomavicius G., Tuzhilin A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering 17*(6), 734–749. doi:10.1109/TKDE.2005.99.

Baeza-Yates R. A., Ribeiro-Neto B. (2011). *Modern information retrieval*. Addison-Wesley.

Balabanović M., Shoham Y. (1997). Fab: content-based, collaborative recommendation. *Communications of the ACM 40*(3), 66–72. doi:10.1145/245108.245124.

Baluja S., Seth R., Sivakumar D., Jing Y., Yagnik J., Kumar S., Ravichandran D., Aly M. (2008). Video suggestion and discovery for youtube: taking random walks through the view graph. In *17th international conference on World Wide Web*, pp. 895–904. ACM. doi:10.1145/1367497.1367618.

Basu C., Hirsh H., Cohen W. (1998). Recommendation as classification: Using social and content-based information in recommendation. In *Aaai/iaai*, pp. 714–720.

Bergstra J., Bengio Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research 13*, 281–305.

Bernardes D., Diaby M., Fournier R., FogelmanSoulié F., Viennet E. (2015). A social formalism and survey for recommender systems. *ACM SIGKDD Explorations Newsletter 16*(2), 20–37. doi:10.1145/2783702.2783705.

Burke R. (2002). Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction 12*(4), 331–370. doi:10.1023/A:1021240730564.

Campos P. G., Díez F., Cantador I. (2014). Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction 24*(1-2), 67–119. doi:10.1007/s11257-012-9136-x.

Chen Y., Zhao X., Gan J., Ren J., Hu Y. (2016). Content-based top-N recommendation using heterogeneous relations. In *Australasian Database Conference*, Volume 9877 of *Lecture Notes in Computer Science*, pp. 308–320. Springer. doi:10.1007/978-3-319-46922-5_24.

Christakopoulou E., Karypis G. (2016). Local item-item models for top-n recommendation. In *10th ACM Conference on Recommender Systems*, pp. 67–74. ACM. doi:10.1145/2959100.2959185.

Cremonesi P., Koren Y., Turrin R. (2010). Performance of recommender algorithms on top-n recommendation tasks. In *4th ACM conference on Recommender systems*, pp. 39–46. ACM. doi:10.1145/1864708.1864721.

Deng S., Huang L., Xu G. (2014). Social network-based service recommendation with trust enhancement. *Expert Systems with Applications 41*(18), 8075–8084. doi:10.1016/j.eswa.2014.07.012.

Deshpande M., Karypis G. (2004). Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS) 22*(1), 143–177. doi:10.1145/963770.963776.

Ding Y., Li X. (2005). Time weight collaborative filtering. In *14th ACM international conference on Information and knowledge management*, pp. 485–492. ACM. doi:10.1145/1099554.1099689.

Eksombatchai C., Jindal P., Liu J. Z., Liu Y., Sharma R., Sugnet C., Ulrich M., Leskovec J. (2018). Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *World Wide Web Conference on World Wide Web*, pp. 1775–1784. International World Wide Web Conferences Steering Committee. doi:10.1145/3178876.3186183.

Gaillard J., Renders J.-M. (2015). Time-sensitive collaborative filtering through adaptive matrix completion. In *European Conference on Information Retrieval*, Volume 9022 of *Lecture Notes in Computer Science*, pp. 327–332. Springer. doi:10.1007/978-3-319-16354-3_35.

Gori M., Pucci A., Roma V., Siena I. (2007). ItemRank: A Random-Walk Based Scoring Algorithm for Recommender Engines. In *20th international joint conference on Artifical intelligence*, Volume 7, pp. 2766–2771.

Grafström J., Jakobsson L., Wiede P. (2018). The Impact of Influencer Marketing on Consumers' Attitudes. Technical report, Jnkping University.

Gunawardana A., Shani G. (2009). A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research 10*, 2935–2962.

Guo G., Zhang J., Thalmann D. (2014). Merging trust in collaborative filtering to alleviate data sparsity and cold start. *Knowledge-Based Systems 57*, 57–68. doi:10.1016/j.knosys.2013.12.007.

Guo G., Zhang J., Zhu F., Wang X. (2017). Factored similarity models with social trust for top-N item recommendation. *Knowledge-Based Systems 122*, 17–25. doi:10.1016/j.knosys.2017.01.027.

Haveliwala T. H. (2002). Topic-sensitive PageRank. In *11th international conference on World Wide Web*, pp. 517–526. ACM. doi:10.1145/511446.511513.

Herlocker J. L., Konstan J. A., Borchers A., Riedl J. (1999). An algorithmic framework for performing collaborative filtering. In *22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 230–237. ACM. doi:10.1145/312624.312682.

Hu Y., Koren Y., Volinsky C. (2008). Collaborative filtering for implicit feedback datasets. In *8th IEEE International Conference on Data Mining*, pp. 263–272. Ieee. doi:10.1109/ICDM.2008.22.

Huang Z., Chen H., Zeng D. (2004). Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems 22*(1), 116–142. doi:10.1145/963770.963775.

Hwang C.-S., Chen Y.-P. (2007). Using trust in collaborative filtering recommendation. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, Volume 4570 of *Lecture Notes in Computer Science*, pp. 1052–1060. Springer. doi:10.1007/978-3-540-73325-6_105.

Jamali M., Ester M. (2009). Using a trust network to improve top-N recommendation. In *3rd ACM conference on Recommender systems*, pp. 181–188. ACM. doi:10.1145/1639714.1639745.

Karahodža B., Donko D., Šupić H. (2015). Temporal dynamics of changes in group user's preferences in recommender systems. In *38th International Convention on Information and Communication Technology, Electronics and Microelectronics*, pp. 1262–1266. IEEE. doi:10.1109/MIPRO.2015.7160469.

Karypis G. (2001). Evaluation of item-based top-n recommendation algorithms. In *10th international conference on Information and knowledge management*, pp. 247–254. ACM.

Kim H.-N., El Saddik A. (2011). Personalized PageRank vectors for tag recommendations: inside FolkRank. In *5th ACM conference on Recommender systems*, pp. 45–52. ACM. doi:10.1145/2043932.2043945.

Konstan J. A., Miller B. N., Maltz D., Herlocker J. L., Gordon L. R., Riedl J. (1997). GroupLens: applying collaborative filtering to Usenet news. *Communications of the ACM 40*(3), 77–87. doi:10.1145/245108.245126.

Koolen M., Bogers T., Gäde M., Hall M., Huurdeman H., Kamps J., Skov M., Toms E., Walsh D. (2015). Overview of the CLEF 2015 social book search lab. In *International conference of the cross-language evaluation forum for European languages*, Volume 9283 of *Lecture Notes in Computer Science*, pp. 545–564. doi:10.1007/978-

3-319-24027-5_51.

Koren Y. (2009). Collaborative filtering with temporal dynamics. In *15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 447–456. doi:10.1145/1557019.1557072.

Koren Y., Bell R., Volinsky C. (2009). Matrix factorization techniques for recommender systems. *Computer 42*(8), 30–37. doi:10.1109/MC.2009.263.

Latapy M., Viard T., Magnien C. (2018). Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining 8*, 61. doi:10.1007/s13278-018-0537-7.

Lathia N., Hailes S., Capra L. (2008). Trust-based collaborative filtering. In *IFIP International Conference on Trust Management*, Volume 263 of *IFIP The International Federation for Information Processing*, pp. 119–134. Springer. doi:10.1007/978-0-387-09428-1_8.

Lathia N., Hailes S., Capra L. (2009). Temporal collaborative filtering with adaptive neighbourhoods. In *32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 796–797. ACM. doi:10.1145/1571941.1572133.

Li L., Yang Z., Wang B., Kitsuregawa M. (2007). Dynamic adaptation strategies for long-term and short-term user profile to personalize search. In *Asia-Pacific Web Conference / International Conference on Web-Age Information Management*, Volume 4505 of *Lecture Notes in Computer Science*, pp. 228–240. Springer. doi:10.1007/978-3-540-72524-4_26.

Li Y., Tang J. (2008). Expertise search in a time-varying social network. In *9th International Conference on Web-Age Information Management*, pp. 293–300. IEEE. doi:10.1109/WAIM.2008.100.

Liu N. N., Zhao M., Xiang E., Yang Q. (2010). Online evolutionary collaborative filtering. In *4th ACM conference on Recommender systems*, pp. 95–102. ACM. doi:10.1145/1864708.1864729.

Liu S., Jiang C., Lin Z., Ding Y., Duan R., Xu Z. (2015). Identifying effective influencers based on trust for electronic word-of-mouth marketing: A domain-aware approach. *Information sciences 306*, 34–52. doi:10.1016/j.ins.2015.01.034.

Lops P., De Gemmis M., Semeraro G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, Chapter 3, pp. 73–105. Springer. doi:10.1007/978-0-387-85820-3_3.

Ma X., Lu H., Gan Z., Zeng J. (2017). An explicit trust and distrust clustering based collaborative filtering recommendation approach. *Electronic Commerce Research and Applications 25*, 29–39. doi:10.1016/j.elerap.2017.06.005.

Massa P., Avesani P. (2007). Trust-aware recommender systems. In *ACM Conference on Recommender systems*, pp. 17–24. ACM. doi:10.1145/1297231.1297235.

McLaughlin M. R., Herlocker J. L. (2004). A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *27th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 329–336. ACM. doi:10.1145/1008992.1009050.

Mooney R. J., Roy L. (2000). Content-based book recommending using learning for text categorization. In *5th ACM conference on Digital libraries*, pp. 195–204. ACM. doi:10.1145/336597.336662.

Nguyen D. P., Le Q. T., Tu M. P. (2008). A graph-based method for combining collaborative and content-based filtering. In *Pacific Rim International Conference on Artificial Intelligence*, pp. 859–869. Springer. doi:10.1007/978-3-540-89197-0_80.

Ning X., Karypis G. (2011). Slim: Sparse linear methods for top-n recommender systems. In *11th IEEE International Conference on Data Mining*, pp. 497–506. IEEE. doi:10.1109/ICDM.2011.134.

Ning X., Karypis G. (2012). Sparse linear methods with side information for top-n recommendations. In *6th ACM conference on Recommender systems*, pp. 155–162. ACM. doi:10.1145/2365952.2365983.

Nzekon Nzeko'o A. J., Tchuente M., Latapy M. (2017). Time Weight Content-Based Extensions of Temporal Graphs for Personalized Recommendation. In *13th International Conference on Web Information Systems and Technologies*.

Nzekon Nzeko'o A. J., Tchuente M., Latapy M. (2019). Link Stream Graph for Temporal Recommendations. In *Colloquium of Mathematics and Computer Science*.

Page L., Brin S., Motwani R., Winograd T. (1999). The PageRank citation ranking: Bringing order to the web.

Technical report, Stanford InfoLab.

Pan Y., He F., Yu H. (2017). Trust-aware Collaborative Denoising Auto-Encoder for Top-N Recommendation. *arXiv cs.IR*, 1703.01760.

Papagelis M., Plexousakis D., Kutsuras T. (2005). Alleviating the sparsity problem of collaborative filtering using trust inferences. In *Trust management*, Volume 3477 of *Lecture Notes in Computer Science*, pp. 224–239. Springer. doi:10.1007/11429760_16.

Pazzani M. J., Muramatsu J., Billsus D. (1996). Syskill & Webert: Identifying interesting web sites. In *13th AAAI national conference on Artificial intelligence*, pp. 54–61.

Pitsilis G., Marshall L. F. (2004). A model of trust derivation from evidence for use in recommendation systems. Technical report, University of Newcastle upon Tyne, School of Computing Science.

Rendle S., Freudenthaler C., Gantner Z., Schmidt-Thieme L. (2009). BPR: Bayesian personalized ranking from implicit feedback. In *25th conference on uncertainty in artificial intelligence*, pp. 452–461. AUAI Press.

Sarwar B., Karypis G., Konstan J., Riedl J. (2001). Item-based collaborative filtering recommendation algorithms. In *10th international conference on World Wide Web*, pp. 285–295. ACM. doi:10.1145/371920.372071.

Shi Y., Karatzoglou A., Baltrunas L., Larson M., Oliver N., Hanjalic A. (2012). CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering. In *6th ACM conference on Recommender systems*, pp. 139–146. ACM. doi:10.1145/2365952.2365981.

Shu J., Shen X., Liu H., Yi B., Zhang Z. (2018). A content-based recommendation algorithm for learning resources. *Multimedia Systems 24*(2), 163–173. doi:10.1007/s00530-017-0539-8.

Şora I. (2015). A PageRank based recommender system for identifying key classes in software systems. In *IEEE 10th Jubilee International Symposium on Applied Computational Intelligence and Informatics*, pp. 495–500. IEEE. doi:10.1109/SACI.2015.7208254.

Steck H. (2013). Evaluation of recommendations: rating-prediction and ranking. In *7th ACM conference on Recommender systems*, pp. 213–220. ACM. doi:10.1145/2507157.2507160.

Strub F., Gaudel R., Mary J. (2016). Hybrid recommender system based on autoencoders. In *1st Workshop on Deep Learning for Recommender Systems*, pp. 11–16. ACM. doi:10.1145/2988450.2988456.

Tang J., Gao H., Liu H. (2012). mTrust: Discerning multi-faceted trust in a connected world. In *5th ACM international conference on Web search and data mining*, pp. 93–102. ACM. doi:10.1145/2124295.2124309.

Viard T., Latapy M., Magnien C. (2016). Computing maximal cliques in link streams. *Theoretical Computer Science 609*, 245–252. doi:10.1016/j.tcs.2015.09.030.

Xiang L., Yuan Q., Zhao S., Chen L., Zhang X., Yang Q., Sun J. (2010). Temporal recommendation on graphs via long-and short-term preference fusion. In *16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 723–732. ACM. doi:10.1145/1835804.1835896.

Xiao L., Min Z., Yongfeng Z., Yiqun L., Shaoping M. (2017). Learning and transferring social and item visibilities for personalized recommendation. In *ACM Conference on Information and Knowledge Management*, pp. 337–346. ACM. doi:10.1145/3132847.3132911.

Yu J., Shen Y., Yang Z. (2014). Topic-STG: Extending the session-based temporal graph approach for personalized tweet recommendation. In *23rd International Conference on World Wide Web*, pp. 413–414. ACM. doi:10.1145/2567948.2577328.

Zhao T., McAuley J., King I. (2014). Leveraging social connections to improve personalized ranking for collaborative filtering. In *23rd ACM International Conference on Conference on Information and Knowledge Management*, pp. 261–270. ACM. doi:10.1145/2661829.2661998.

A ACKNOWLEDGEMENTS

We thank Raphaël Fournier, Tiphaine Viard and JIMIS reviewers for their helpful comments on previous versions. This work is funded in part by the African Center of Excellence in Information and Communication Technologies (CETIC), the Sorbonne University-IRD PDI program, and by the ANR (French National Agency of Research) under grant ANR-15-CE38-0001 (AlgoDiv).

B APPENDIX

In this section, we present the results obtained for top-20, -50 and -100. The section is divided in two parts: the first one presents performances obtained for all combinations of side information and basic graphs of the framework; the second highlights the 3 best combinations, according to basic graph and evaluation metric.

These two parts confirm observations made on top-10 results in the Section VI. For example, recommender graphs that integrate simultaneously content-based, users' preferences temporal dynamic and trust relationship between users, are usually the best. Thus, we recommend the simultaneous integration of these three side information in order to increase the chances to achieve good performances.

EPINIONS	F1@20			HR@20			MAP@20		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.56	1.59	1.11	8.22	8.22	7.16	2.38	2.4	1.84
ET	1.31	1.49	1.05	7.43	7.96	6.9	2.18	2.24	1.8
IT	1.73	1.77	1.39	8.75	9.02	8.89	2.51	2.49	2.41
EDF	2.7	2.15	2.55	10.6	9.55	10.6	2.62	2.56	2.61
LDF	1.88	1.68	1.53	8.89	8.49	8.49	2.38	2.23	2.67
CI	1.95	2.15	0.76	9.28	9.68	5.44	2.64	2.98	1.89
CIU	2.16	2.36	0.63	9.95	10.1	4.64	2.9	3.13	1.73
EDF-ET	1.8	1.68	2.46	8.75	8.62	10.3	2.36	2.15	2.41
EDF-IT	2.47	2.5	2.43	9.95	10.1	10.3	2.84	2.67	2.64
LDF-ET	1.39	1.57	1.21	7.69	8.22	7.56	2.17	2.24	2.32
LDF-IT	1.92	1.95	2.18	9.55	9.81	10.5	3.12	3.12	3.06
CI-ET	1.67	2.12	0.79	8.49	9.55	5.57	2.4	2.83	1.9
CI-IT	2.05	2.19	1.49	9.95	10.1	8.75	2.63	2.8	2.35
CIU-ET	2.07	2.12	0.64	9.68	9.95	4.64	2.67	3.13	1.73
CIU-IT	2.19	2.35	1.53	10.1	10.1	9.15	2.9	3.13	2.43
CI-EDF	3.25	2.88	0.81	11.7	11.1	5.7	3.33	2.97	1.9
CI-LDF	2.57	2.38	0.85	10.6	9.68	5.97	2.76	2.83	1.9
CIU-EDF	3.45	2.93	0.67	11.8	11.3	5.04	3.45	3.26	1.79
CIU-LDF	3.02	2.39	0.74	11.0	10.6	5.31	3.01	3.09	1.82
CI-EDF-ET	2.6	2.58	0.81	10.5	10.7	5.7	2.7	2.71	1.9
CI-EDF-IT	3.25	3.26	1.52	11.3	11.4	8.89	3.03	2.99	2.4
CI-LDF-ET	1.98	2.17	0.86	9.15	9.68	5.97	2.42	2.83	1.9
CI-LDF-IT	2.63	2.3	2.09	10.5	10.2	9.81	3.15	3.14	2.77
CIU-EDF-ET	2.92	2.85	0.67	11.0	10.9	5.04	2.87	3.11	1.79
CIU-EDF-IT	3.44	3.33	1.48	11.7	11.5	9.02	3.55	3.41	2.46
CIU-LDF-ET	2.43	2.34	0.67	10.1	10.5	5.04	2.73	3.01	1.74
CIU-LDF-IT	2.81	2.51	1.93	10.7	10.5	9.42	3.29	3.24	3.26

Table 6: Epinions Dataset - Performances with optimal settings for Top-20.

EPINIONS	F1@50			HR@50			MAP@50		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.06	1.08	0.91	15.3	15.4	13.9	2.5	2.55	2.03
ET	0.95	1.05	0.87	14.2	15.1	13.5	2.3	2.4	1.99
IT	1.16	1.16	0.97	16.0	16.4	14.7	2.52	2.63	2.55
EDF	1.25	1.28	1.04	17.1	16.4	15.4	2.64	2.71	2.7
LDF	1.14	1.15	0.99	16.0	16.0	14.7	2.5	2.4	2.72
CI	1.55	1.48	0.56	19.0	18.4	9.68	2.87	3.15	1.93
CIU	1.59	1.53	0.48	19.0	18.6	8.62	3.01	3.26	1.83
EDF-ET	1.09	1.11	0.89	15.5	15.5	14.2	2.43	2.33	2.52
EDF-IT	1.36	1.34	0.97	17.5	16.8	15.0	2.83	2.74	2.73
LDF-ET	0.97	1.08	0.87	14.6	15.4	13.5	2.28	2.4	2.38
LDF-IT	1.18	1.16	0.98	16.6	16.4	14.9	3.09	3.09	3.07
CI-ET	1.43	1.47	0.56	18.2	18.3	9.68	2.63	3.05	1.93
CI-IT	1.49	1.47	0.89	18.4	18.4	14.2	2.77	3.03	2.41
CIU-ET	1.49	1.49	0.48	18.3	18.6	8.49	2.76	3.27	1.83
CIU-IT	1.51	1.55	0.91	18.6	18.6	14.6	3.07	3.29	2.5
CI-EDF	1.74	1.66	0.58	19.9	19.9	9.95	3.29	3.06	1.94
CI-LDF	1.67	1.51	0.6	19.6	18.7	10.1	2.89	3.05	1.95
CIU-EDF	1.63	1.77	0.55	19.4	19.9	9.81	3.5	3.36	1.84
CIU-LDF	1.59	1.55	0.54	19.0	19.0	9.95	3.2	3.33	1.92
CI-EDF-ET	1.72	1.63	0.59	19.8	19.8	9.95	2.71	2.93	1.94
CI-EDF-IT	1.52	1.66	0.88	18.8	19.9	14.2	3.1	2.99	2.46
CI-LDF-ET	1.56	1.52	0.56	19.0	18.8	9.68	2.65	3.06	1.93
CI-LDF-IT	1.49	1.49	0.93	18.6	18.6	14.5	3.14	3.14	2.82
CIU-EDF-ET	1.52	1.75	0.55	18.7	19.9	9.81	3.02	3.29	1.83
CIU-EDF-IT	1.62	1.73	0.91	19.6	19.8	14.6	3.65	3.48	2.54
CIU-LDF-ET	1.47	1.55	0.51	18.2	19.0	9.42	2.83	3.18	1.84
CIU-LDF-IT	1.52	1.53	0.97	18.7	18.8	15.1	3.36	3.45	3.27

Table 7: Epinions Dataset - Performances with optimal settings for Top-50.

EPINIONS	F1@100			HR@100			MAP@100		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	0.7	0.72	0.67	22.5	22.4	21.9	2.54	2.59	2.13
ET	0.69	0.68	0.66	22.3	22.0	21.5	2.35	2.43	2.09
IT	0.74	0.77	0.66	23.3	23.6	21.9	2.6	2.68	2.61
EDF	0.77	0.77	0.67	22.9	23.3	21.9	2.66	2.66	2.72
LDF	0.71	0.72	0.67	22.5	22.3	21.9	2.54	2.42	2.65
CI	0.89	0.91	0.39	26.3	26.3	14.3	2.84	3.13	1.83
CIU	0.87	0.92	0.39	26.0	27.5	14.3	3.03	3.35	1.82
EDF-ET	0.71	0.7	0.65	22.1	22.0	21.2	2.44	2.33	2.54
EDF-IT	0.76	0.79	0.67	23.5	23.9	22.3	2.9	2.77	2.77
LDF-ET	0.71	0.69	0.66	22.4	21.9	21.5	2.33	2.42	2.36
LDF-IT	0.75	0.79	0.66	23.6	23.9	21.9	3.1	3.1	3.02
CI-ET	0.87	0.88	0.39	25.9	25.7	14.3	2.6	2.96	1.83
CI-IT	0.88	0.88	0.58	26.1	26.5	20.0	2.78	2.93	2.48
CIU-ET	0.86	0.92	0.39	25.9	27.5	14.3	2.85	3.36	1.82
CIU-IT	0.89	0.92	0.59	26.5	27.3	20.2	3.15	3.35	2.52
CI-EDF	0.92	0.98	0.42	26.8	27.6	15.1	3.29	3.15	1.85
CI-LDF	0.89	0.88	0.42	26.3	26.0	15.1	2.88	2.95	1.89
CIU-EDF	0.91	1.03	0.4	26.3	28.5	14.5	3.58	3.34	1.83
CIU-LDF	0.89	0.94	0.41	26.3	27.6	15.0	3.12	3.33	1.9
CI-EDF-ET	0.91	0.95	0.42	26.5	27.2	15.0	2.71	2.89	1.84
CI-EDF-IT	0.92	0.95	0.58	26.7	27.2	20.0	3.2	3.06	2.49
CI-LDF-ET	0.88	0.89	0.41	25.7	25.7	14.6	2.61	2.95	1.83
CI-LDF-IT	0.88	0.88	0.57	26.1	26.5	20.0	3.16	3.16	2.82
CIU-EDF-ET	0.91	1.02	0.39	26.1	28.5	14.3	2.94	3.21	1.82
CIU-EDF-IT	0.91	1.02	0.59	27.2	28.5	20.2	3.73	3.44	2.55
CIU-LDF-ET	0.88	0.94	0.39	26.0	27.6	14.3	2.79	3.17	1.82
CIU-LDF-IT	0.89	0.92	0.59	26.5	27.3	20.3	3.37	3.43	3.26

Table 8: Epinions Dataset - Performances with optimal settings for Top-100.

CIAO	F1@20			HR@20			MAP@20		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.61	1.93	1.71	9.26	9.98	10.2	2.13	2.25	2.44
ET	1.36	1.69	1.61	8.71	9.26	9.98	1.96	2.13	2.35
IT	3.08	2.72	2.2	13.1	12.9	11.8	2.76	2.75	2.51
EDF	2.24	2.35	2.51	11.1	11.4	12.3	2.28	2.56	2.65
LDF	2.05	1.99	2.7	10.2	10.2	13.2	2.74	3.0	3.78
CI	3.13	3.58	1.03	12.2	12.3	7.44	2.37	2.45	1.75
CIU	2.93	4.39	0.87	13.2	14.5	6.72	2.8	3.43	1.55
EDF-ET	2.01	2.01	1.69	10.7	10.3	11.4	2.0	2.36	2.41
EDF-IT	3.15	2.69	2.34	14.2	13.8	12.0	3.27	3.11	2.56
LDF-ET	1.44	1.69	1.61	8.71	9.26	9.98	2.02	2.22	2.36
LDF-IT	2.82	2.82	2.6	12.3	12.5	12.3	3.06	3.25	2.81
CI-ET	2.59	3.25	1.09	11.6	12.2	7.8	2.3	2.39	1.88
CI-IT	3.35	3.47	2.76	13.4	14.0	12.7	2.87	2.83	2.68
CIU-ET	3.78	4.39	1.03	14.0	14.5	7.26	2.73	3.51	1.63
CIU-IT	3.64	4.71	3.27	14.5	14.7	13.4	3.13	3.39	2.66
CI-EDF	4.81	3.95	1.48	15.1	13.8	9.26	2.77	3.01	1.84
CI-LDF	4.27	3.92	1.54	14.9	14.2	10.3	3.29	3.21	2.92
CIU-EDF	4.37	4.19	1.32	15.8	14.7	8.17	3.66	3.6	1.84
CIU-LDF	3.43	4.55	1.39	15.2	15.4	9.62	3.62	3.68	2.82
CI-EDF-ET	2.19	3.78	1.47	13.8	13.6	9.07	2.76	2.78	1.93
CI-EDF-IT	4.18	3.86	2.95	15.4	14.9	13.1	3.41	3.42	2.76
CI-LDF-ET	2.7	3.33	1.24	11.6	13.1	8.71	2.31	2.44	2.01
CI-LDF-IT	4.87	4.92	3.38	15.1	14.5	13.8	3.44	3.48	3.45
CIU-EDF-ET	3.7	4.38	1.32	15.8	14.5	8.17	3.24	3.42	1.84
CIU-EDF-IT	3.84	4.56	3.39	16.0	15.8	13.6	3.63	3.64	2.69
CIU-LDF-ET	4.2	4.5	1.21	14.5	15.4	8.35	2.79	3.69	1.69
CIU-LDF-IT	5.06	4.36	3.28	16.0	15.8	14.5	3.59	3.69	3.49

Table 9: Ciao Dataset - Performances with optimal settings for Top-20.

CIAO	F1@50			HR@50			MAP@50		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.46	1.55	1.51	19.4	19.6	19.8	2.34	2.5	2.74
ET	1.58	1.52	1.47	19.6	19.2	20.0	2.2	2.43	2.67
IT	1.69	1.72	1.62	20.9	21.1	20.5	2.92	2.84	2.73
EDF	1.74	1.74	1.81	21.4	20.7	20.9	2.56	2.78	2.86
LDF	1.47	1.62	1.77	19.4	20.3	20.9	2.91	3.0	3.97
CI	2.27	2.18	0.97	22.1	22.5	15.2	2.55	2.71	1.92
CIU	2.37	2.65	0.88	24.0	24.9	14.3	3.03	3.67	1.72
EDF-ET	1.63	1.59	1.59	20.3	20.9	20.7	2.24	2.65	2.73
EDF-IT	1.98	1.79	1.7	22.1	21.8	21.2	3.39	3.23	2.81
LDF-ET	1.59	1.58	1.47	19.6	19.6	20.0	2.25	2.54	2.68
LDF-IT	1.9	1.69	1.65	21.8	20.5	20.5	3.18	3.29	2.99
CI-ET	2.34	2.07	0.98	23.0	23.0	15.4	2.51	2.68	1.92
CI-IT	2.53	2.29	1.57	23.6	22.7	20.5	2.98	2.91	2.76
CIU-ET	2.27	2.66	0.88	24.0	24.7	14.2	2.98	3.75	1.74
CIU-IT	2.52	2.86	1.58	24.7	25.8	21.2	3.26	3.62	2.88
CI-EDF	2.45	2.36	1.26	23.0	24.0	18.9	2.98	3.18	2.02
CI-LDF	2.56	2.23	1.17	23.4	22.9	17.6	3.45	3.29	3.12
CIU-EDF	2.57	2.71	1.03	24.0	25.4	16.5	3.87	3.87	2.07
CIU-LDF	2.62	2.78	1.13	24.0	24.7	16.9	3.74	3.82	3.01
CI-EDF-ET	2.63	2.34	1.29	24.1	23.8	18.9	3.01	3.04	2.05
CI-EDF-IT	2.6	2.44	1.59	24.5	24.1	20.7	3.53	3.56	2.94
CI-LDF-ET	2.31	2.21	0.99	23.4	23.8	15.4	2.63	2.75	2.05
CI-LDF-IT	2.51	2.42	1.64	23.8	23.8	21.1	3.61	3.56	3.61
CIU-EDF-ET	2.69	2.69	1.07	24.3	25.8	16.9	3.42	3.7	2.07
CIU-EDF-IT	2.82	2.69	1.61	25.4	25.4	21.4	3.69	3.73	2.91
CIU-LDF-ET	2.55	2.72	1.08	23.8	25.2	16.7	3.0	3.86	1.93
CIU-LDF-IT	2.99	2.88	1.61	24.9	24.9	20.9	3.69	3.76	3.51

Table 10: Ciao Dataset - Performances with optimal settings for Top-50.

CIAO	F1@100			HR@100			MAP@100		
	BIP	STG	LSG	BIP	STG	LSG	BIP	STG	LSG
-	1.11	1.15	1.06	27.9	28.3	27.8	2.39	2.49	2.64
ET	1.08	1.15	1.03	27.9	28.1	27.4	2.25	2.36	2.56
IT	1.16	1.15	1.07	28.7	28.7	27.9	2.88	2.85	2.73
EDF	1.25	1.27	1.18	30.5	31.0	29.9	2.46	2.64	2.86
LDF	1.23	1.14	1.17	29.6	28.5	29.2	2.96	3.05	3.94
CI	1.54	1.51	0.96	33.8	33.4	26.7	2.62	2.74	2.02
CIU	1.58	1.57	0.89	34.3	34.5	25.8	3.08	3.51	1.72
EDF-ET	1.21	1.27	1.11	29.2	30.3	28.3	2.18	2.57	2.62
EDF-IT	1.22	1.26	1.09	28.9	30.1	27.9	3.37	3.22	2.78
LDF-ET	1.11	1.15	1.03	28.7	28.7	27.6	2.32	2.47	2.57
LDF-IT	1.14	1.18	1.07	27.9	29.2	27.9	3.22	3.31	3.02
CI-ET	1.57	1.52	0.98	33.9	33.4	26.5	2.58	2.65	2.04
CI-IT	1.52	1.53	1.09	33.2	33.4	29.9	2.97	2.91	2.83
CIU-ET	1.57	1.56	0.9	34.8	35.2	25.8	3.03	3.6	1.8
CIU-IT	1.48	1.5	1.09	33.4	34.1	29.2	3.23	3.47	2.85
CI-EDF	1.66	1.56	1.01	34.5	33.6	28.5	2.92	3.13	2.14
CI-LDF	1.6	1.49	1.0	34.5	33.6	27.2	3.37	3.32	3.18
CIU-EDF	1.73	1.74	0.9	35.0	35.9	26.5	3.83	3.76	2.17
CIU-LDF	1.56	1.5	0.89	34.5	34.3	26.0	3.7	3.81	2.98
CI-EDF-ET	1.71	1.54	1.01	34.5	33.6	29.2	2.91	2.98	2.12
CI-EDF-IT	1.57	1.54	1.16	33.0	33.4	29.8	3.41	3.55	2.85
CI-LDF-ET	1.55	1.55	0.98	33.9	33.4	26.5	2.63	2.71	2.16
CI-LDF-IT	1.49	1.5	1.15	32.7	33.8	30.5	3.58	3.59	3.5
CIU-EDF-ET	1.84	1.76	0.91	35.8	35.9	26.9	3.33	3.66	2.17
CIU-EDF-IT	1.74	1.73	1.1	34.7	35.8	29.6	3.64	3.68	2.89
CIU-LDF-ET	1.59	1.57	0.9	34.7	34.7	25.8	3.04	3.72	1.98
CIU-LDF-IT	1.52	1.51	1.13	33.2	34.5	30.5	3.75	3.79	3.46

Table 11: Ciao Dataset - Performances with optimal settings for Top-100.

Epinions Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@20	1	1.56	3.45	121%	CIU-EDF	1.59	3.33	109%	CIU-EDF-IT	1.11	2.55	129%	EDF
	2	1.56	3.44	120%	CIU-EDF-IT	1.59	3.26	105%	CI-EDF-IT	1.11	2.46	121%	EDF-ET
	3	1.56	3.25	109%	CI-EDF-IT	1.59	2.93	84%	CIU-EDF	1.11	2.43	118%	EDF-IT
H@20	1	8.22	11.8	43%	CIU-EDF	8.22	11.5	40%	CIU-EDF-IT	7.16	10.6	48%	EDF
	2	8.22	11.7	41%	CI-EDF	8.22	11.4	38%	CI-EDF-IT	7.16	10.5	46%	LDF-IT
	3	8.22	11.7	41%	CIU-EDF-IT	8.22	11.3	37%	CIU-EDF	7.16	10.3	44%	EDF-ET
M@20	1	2.38	3.55	49%	CIU-EDF-IT	2.4	3.41	41%	CIU-EDF-IT	1.84	3.26	77%	CIU-LDF-IT
	2	2.38	3.45	45%	CIU-EDF	2.4	3.26	35%	CIU-EDF	1.84	3.06	66%	LDF-IT
	3	2.38	3.33	40%	CI-EDF	2.4	3.24	34%	CIU-LDF-IT	1.84	2.77	50%	CI-LDF-IT

Epinions Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@50	1	1.06	1.74	64%	CI-EDF	1.08	1.77	62%	CIU-EDF	0.91	1.04	14%	EDF
	2	1.06	1.72	61%	CI-EDF-ET	1.08	1.75	61%	CIU-EDF-ET	0.91	0.99	9%	LDF
	3	1.06	1.67	57%	CI-LDF	1.08	1.73	59%	CIU-EDF-IT	0.91	0.98	7%	LDF-IT
H@50	1	15.3	19.9	30%	CI-EDF	15.4	19.9	29%	CI-EDF	13.9	15.4	10%	EDF
	2	15.3	19.8	29%	CI-EDF-ET	15.4	19.9	29%	CIU-EDF	13.9	15.1	8%	CIU-LDF-IT
	3	15.3	19.6	28%	CI-LDF	15.4	19.9	29%	CI-EDF-IT	13.9	15.0	7%	EDF-IT
M@50	1	2.5	3.65	45%	CIU-EDF-IT	2.55	3.48	36%	CIU-EDF-IT	2.03	3.27	61%	CIU-LDF-IT
	2	2.5	3.5	40%	CIU-EDF	2.55	3.45	35%	CIU-LDF-IT	2.03	3.07	51%	LDF-IT
	3	2.5	3.36	34%	CIU-LDF-IT	2.55	3.36	31%	CIU-EDF	2.03	2.82	38%	CI-LDF-IT

Epinions Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@100	1	0.7	0.92	30%	CI-EDF-IT	0.72	1.03	44%	CIU-EDF	0.67	0.67	0%	-
	2	0.7	0.92	30%	CI-EDF	0.72	1.02	42%	CIU-EDF-ET	0.67	0.67	0%	LDF
	3	0.7	0.91	29%	CIU-EDF-IT	0.72	1.02	42%	CIU-EDF-IT	0.67	0.67	0%	EDF-IT
H@100	1	22.5	27.2	20%	CIU-EDF-IT	22.4	28.5	27%	CIU-EDF	21.9	22.3	1%	EDF-IT
	2	22.5	26.8	18%	CI-EDF	22.4	28.5	27%	CIU-EDF-ET	21.9	21.9	0%	-
	3	22.5	26.7	18%	CI-EDF-IT	22.4	28.5	27%	CIU-EDF-IT	21.9	21.9	0%	IT
M@100	1	2.54	3.73	46%	CIU-EDF-IT	2.59	3.44	32%	CIU-EDF-IT	2.13	3.26	53%	CIU-LDF-IT
	2	2.54	3.58	41%	CIU-EDF	2.59	3.43	32%	CIU-LDF-IT	2.13	3.02	41%	LDF-IT
	3	2.54	3.37	33%	CIU-LDF-IT	2.59	3.36	29%	CIU-ET	2.13	2.82	32%	CI-LDF-IT

Table 12: Epinions Dataset - Best recommender graphs for Top-20, -50 and -100. Comparison of the three best recommender graph combinations with the associated basic graph.

Ciao Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@20	1	1.61	5.06	215%	CIU-LDF-IT	1.93	4.92	155%	CI-LDF-IT	1.71	3.39	98%	CIU-EDF-IT
	2	1.61	4.87	202%	CI-LDF-IT	1.93	4.71	144%	CIU-IT	1.71	3.38	97%	CI-LDF-IT
	3	1.61	4.81	199%	CI-EDF	1.93	4.56	136%	CIU-EDF-IT	1.71	3.28	91%	CIU-LDF-IT
H@20	1	9.26	16.0	72%	CIU-EDF-IT	9.98	15.8	58%	CIU-EDF-IT	10.2	14.5	42%	CIU-LDF-IT
	2	9.26	16.0	72%	CIU-LDF-IT	9.98	15.8	58%	CIU-LDF-IT	10.2	13.8	35%	CI-LDF-IT
	3	9.26	15.8	70%	CIU-EDF	9.98	15.4	54%	CIU-LDF	10.2	13.6	33%	CIU-EDF-IT
M@20	1	2.13	3.66	72%	CIU-EDF	2.25	3.69	64%	CIU-LDF-ET	2.44	3.78	54%	LDF
	2	2.13	3.63	70%	CIU-EDF-IT	2.25	3.69	64%	CIU-LDF-IT	2.44	3.49	42%	CIU-LDF-IT
	3	2.13	3.62	70%	CIU-LDF	2.25	3.68	63%	CIU-LDF	2.44	3.45	41%	CI-LDF-IT

Ciao Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@50	1	1.46	2.99	104%	CIU-LDF-IT	1.55	2.88	85%	CIU-LDF-IT	1.51	1.81	20%	EDF
	2	1.46	2.82	93%	CIU-EDF-IT	1.55	2.86	84%	CIU-IT	1.51	1.77	17%	LDF
	3	1.46	2.69	83%	CIU-EDF-ET	1.55	2.78	79%	CIU-LDF	1.51	1.7	12%	EDF-IT
H@50	1	19.4	25.4	30%	CIU-EDF-IT	19.6	25.8	31%	CIU-IT	19.8	21.4	8%	CIU-EDF-IT
	2	19.4	24.9	28%	CIU-LDF-IT	19.6	25.8	31%	CIU-EDF-ET	19.8	21.2	7%	EDF-IT
	3	19.4	24.7	27%	CIU-IT	19.6	25.4	29%	CIU-EDF	19.8	21.2	7%	CIU-IT
M@50	1	2.34	3.87	65%	CIU-EDF	2.5	3.87	54%	CIU-EDF	2.74	3.97	44%	LDF
	2	2.34	3.74	60%	CIU-LDF	2.5	3.86	54%	CIU-LDF-ET	2.74	3.61	31%	CI-LDF-IT
	3	2.34	3.69	57%	CIU-LDF-IT	2.5	3.82	52%	CIU-LDF	2.74	3.51	28%	CIU-LDF-IT

Ciao Dataset

	No	BIP				STG				LSG			
		Basic	Best	Imp.	BIP-Best	Basic	Best	Imp.	STG-Best	Basic	Best	Imp.	LSG-Best
F@100	1	1.11	1.84	66%	CIU-EDF-ET	1.15	1.76	52%	CIU-EDF-ET	1.06	1.18	11%	EDF
	2	1.11	1.74	57%	CIU-EDF-IT	1.15	1.74	51%	CIU-EDF	1.06	1.17	10%	LDF
	3	1.11	1.73	56%	CIU-EDF	1.15	1.73	50%	CIU-EDF-IT	1.06	1.16	9%	CI-EDF-IT
H@100	1	27.9	35.8	27%	CIU-EDF-ET	28.3	35.9	26%	CIU-EDF	27.8	30.5	9%	CI-LDF-IT
	2	27.9	35.0	25%	CIU-EDF	28.3	35.9	26%	CIU-EDF-ET	27.8	30.5	9%	CIU-LDF-IT
	3	27.9	34.8	24%	CIU-ET	28.3	35.8	26%	CIU-EDF-IT	27.8	29.9	7%	EDF
M@100	1	2.39	3.83	60%	CIU-EDF	2.49	3.81	52%	CIU-LDF	2.64	3.94	49%	LDF
	2	2.39	3.75	57%	CIU-LDF-IT	2.49	3.79	52%	CIU-LDF-IT	2.64	3.5	32%	CI-LDF-IT
	3	2.39	3.7	55%	CIU-LDF	2.49	3.76	51%	CIU-EDF	2.64	3.46	31%	CIU-LDF-IT

Table 13: Ciao Dataset - Best recommender graphs for Top-20, -50 and -100. Comparison of the three best recommender graph combinations with the associated basic graph.

Chapitre C

Liste protocolaire

UNIVERSITÉ DE YAOUNDÉ I **Faculté des Sciences** Division de la Programmation et du Suivi des Activités Académiques		THE UNIVERSITY OF YAOUNDE I **Faculty of Science** Division of Programming and Follow-up of Academic Affaires
LISTE DES ENSEIGNANTS PERMANENTS		**LIST OF PERMANENT TEACHING STAFF**

ANNÉE ACADEMIQUE 2018/2019

(Par Département et par Grade)

DATE D'ACTUALISATION 19 Février 2019

ADMINISTRATION

DOYEN : TCHOUANKEU Jean- Claude, *Maitre de Conférences*
VICE-DOYEN / DPSAA : DONGO Etienne, *Professeur*
VICE-DOYEN / DSSE : AJEAGAH Gideon AGHAINDUM, Maître *de Conférences*
VICE-DOYEN / DRC : ABOSSOLO Monique, Maitre de Conférences
Chef Division Administrative et Financière : NDOYE FOE Marie C. F., *Maitre de Conférences*
Chef Division des Affaires Académiques, de la Scolarité et de la Recherche DAASR : MBAZE MEVA'A Luc Léonard, *Professeur*

1- DÉPARTEMENT DE BIOCHIMIE (BC) (37)

N°	NOMS ET PRÉNOMS	GRADE	OBSERVATIONS
1	FEKAM BOYOM Fabrice	Professeur	En poste
2	MBACHAM FON Wilfried	Professeur	En poste
3	MOUNDIPA FEWOU Paul	Professeur	Chef de Département
4	NINTCHOM PENLAP V. épse BENG	Professeur	En poste
5	OBEN Julius ENYONG	Professeur	En poste
6	ACHU Merci BIH	Maître de Conférences	En poste
7	ATOGHO Barbara Mma	Maître de Conférences	En poste
8	BELINGA née NDOYE FOE M. C. F.	Maître de Conférences	Chef DAF / FS
9	BIGOGA DIAGA Jude	Maître de Conférences	En poste
10	BOUDJEKO Thaddée	Maître de Conférences	En poste
11	EFFA NNOMO Pierre	Maître de Conférences	En poste
12	FOKOU Elie	Maître de Conférences	En poste
13	KANSCI Germain	Maître de Conférences	En poste
14	NANA Louise épouse WAKAM	Maître de Conférences	En poste
15	NGONDI Judith Laure	Maître de Conférences	En poste
16	NGUEFACK Julienne	Maître de Conférences	En poste
17	NJAYOU Frédéric Nico	Maître de Conférences	En poste
18	AKINDEH MBUH NJI	Chargée de Cours	En poste
19	BEBOY EDZENGUELE Sara Nathalie	Chargée de Cours	En poste
20	DAKOLE DABOY Charles	Chargée de Cours	En poste
21	DJOKAM TAMO Rosine	Chargée de Cours	En poste
22	DJUIDJE NGOUNOUE Marcelline	Chargée de Cours	En poste
24	DJUIKWO NKONGA Ruth Viviane	Chargée de Cours	En poste
25	DONGMO LEKAGNE Joseph Blaise	Chargé de Cours	En poste
26	EWANE Cécile Anne	Chargée de Cours	En poste
27	FONKOUA Martin	Chargé de Cours	En poste
28	BEBEE Fadimatou	Chargée de Cours	En poste

29	KOTUE KAPTUE Charles	Chargé de Cours	En poste
30	LUNGA Paul KEILAH	Chargé de Cours	En poste
31	MANANGA Marlyse Joséphine	Chargée de Cours	En poste
32	MBONG ANGIE M. Mary Anne	Chargée de Cours	En poste
33	MOFOR née TEUGWA Clotilde	Chargée de Cours	Inspecteur de Service MINESUP
34	PACHANGOU NSANGOU Sylvain	Chargé de Cours	En poste
35	Palmer MASUMBE NETONGO	Chargé de Cours	En poste
36	TCHANA KOUATCHOUA Angèle	Chargée de Cours	En poste

37	MBOUCHE FANMOE Marceline Joëlle	Assistante	En poste

2- DÉPARTEMENT DE BIOLOGIE ET PHYSIOLOGIE ANIMALES (BPA) (44)

1	BILONG BILONG Charles-Félix	Professeur	Chef de Département
2	DIMO Théophile	Professeur	En Poste
3	DJIETO LORDON Champlain	Professeur	En Poste
4	ESSOMBA née NTSAMA MBALA	Professeur	*VDoyen/FMSB/UYI*
5	FOMENA Abraham	Professeur	En Poste
6	KAMGANG René	Professeur	*C.S. MINRESI*
7	KAMTCHOUING Pierre	Professeur	En poste
8	NJAMEN Dieudonné	Professeur	En poste
9	NJIOKOU Flobert	Professeur	En Poste
10	NOLA Moïse	Professeur	En poste
11	TAN Paul VERNYUY	Professeur	En poste
12	TCHUEM TCHUENTE Louis Albert	Professeur	*Inspecteur de service* *Coord.Progr./MINSANTE*

13	AJEAGAH Gideon AGHAINDUM	Maître de Conférences	*VICE-DOYEN / DSSE*
14	DZEUFIET DJOMENI Paul Désiré	Maître de Conférences	En poste
15	FOTO MENBOHAN Samuel	Maître de Conférences	En poste
20	JATSA BOUKENG Hermine épse MEGAPTCHE	Maître de Conférences	En Poste
16	KEKEUNOU Sévilor	Maître de Conférences	En poste
17	MEGNEKOU Rosette	Maître de Conférences	En poste
18	MONY Ruth épse NTONE	Maître de Conférences	En Poste
19	NGUEGUIM TSOFACK Florence	Maître de Conférences	En poste
21	TOMBI Jeannette	Maître de Conférences	En poste
22	ZEBAZE TOGOUET Serge Hubert	Maître de Conférences	En poste

23	ALENE Désirée Chantal	Chargée de Cours	En poste
24	ATSAMO Albert Donatien	Chargé de Cours	En poste
25	BELLET EDIMO Oscar Roger	Chargé de Cours	En poste
26	BILANDA Danielle Claude	Chargée de Cours	En poste
27	DJIOGUE Séfirin	Chargée de Cours	En poste
28	DONFACK Mireille	Chargée de Cours	En poste
29	GOUNOUE KAMKUMO Raceline	Chargée de Cours	En poste
30	KANDEDA KAVAYE Antoine	Chargé de Cours	En poste
31	LEKEUFACK FOLEFACK Guy B.	Chargé de Cours	En poste

32	MAHOB Raymond Joseph	Chargé de Cours	En poste
33	MBENOUN MASSE Paul Serge	Chargé de Cours	En poste
34	MOUNGANG LucianeMarlyse	Chargée de Cours	En poste
35	MVEYO NDANKEU Yves Patrick	Chargé de Cours	En poste
36	NGOUATEU KENFACK Omer Bébé	Chargé de Cours	En poste
37	NGUEMBOK	Chargé de Cours	En poste
38	NJUA Clarisse Yafi	Chargée de Cours	Chef Div. UBA
39	NOAH EWOTI Olive Vivien	Chargée de Cours	En poste
40	TADU Zephyrin	Chargé de Cours	En poste
41	YEDE	Chargé de Cours	En poste

43	ETEME ENAMA Serge	Assistant	En poste
44	KOGA MANG DOBARA	Assistant	En poste

3- DÉPARTEMENT DE BIOLOGIE ET PHYSIOLOGIE VÉGÉTALES (BPV) (27)

1	AMBANG Zachée	Professeur	Chef Division/UYII
2	BELL Joseph Martin	Professeur	En poste
3	MOSSEBO Dominique Claude	Professeur	En poste
4	YOUMBI Emmanuel	Professeur	Chef de Département
5	ZAPFACK Louis	Professeur	En poste

6	ANGONI Hyacinthe	Maître de Conférences	En poste
7	BIYE Elvire Hortense	Maître de Conférences	En poste
8	DJOCGOUE Pierre François	Maître de Conférences	En poste
9	KENGNE NOUMSI Ives Magloire	Maître de Conférences	En poste
10	MALA Armand William	Maître de Conférences	En poste
11	MBARGA BINDZI Marie Alain	Maître de Conférences	CT/UDs
12	MBOLO Marie	Maître de Conférences	En poste
13	NDONGO BEKOLO	Maître de Conférences	*CE / MINRESI*
14	NGONKEU MAGAPTCHE Eddy L.	Maître de Conférences	En poste
15	TSOATA Esaïe	Maître de Conférences	En poste

16	GOMANDJE Christelle	Chargée de Cours	En poste
17	MAFFO MAFFO Nicole Liliane	Chargé de Cours	En poste
18	MAHBOU SOMO TOUKAM. Gabriel	Chargé de Cours	En poste
19	NGALLE Hermine BILLE	Chargée de Cours	En poste
20	NGOUO Lucas Vincent	Chargé de Cours	En poste
22	NOUKEU KOUAKAM Armelle	Chargé de Cours	En poste
23	ONANA JEAN MICHEL	Chargé de Cours	En poste
24	NSOM ZAMO Annie Claude épse PIAL	Chargée de Cours	*Expert national/UNESCO*
25	TONFACK Libert Brice	Chargé de Cours	En poste

26	DJEUANI Astride Carole	Assistante	En poste
27	NNANGA MEBENGA Ruth Laure	Assistante	En poste

4- DÉPARTEMENT DE CHIMIE INORGANIQUE (CI) (32)

1	AGWARA ONDOH Moïse	Professeur	*Vice RecteurUniv ,Bamenda*
2	ELIMBI Antoine	Professeur	En poste
3	Florence UFI CHINJE épouse MELO	Professeur	*Recteur Univ.Ngaoundere*
4	GHOGOMU Paul MINGO	Professeur	*Ministre Chargé deMiss.PR*
5	NANSEU Njiki Charles Péguy	Professeur	En poste
6	NDIFON Peter TEKE	Professeur	*CT MINRESI/Chef de Departement*
7	NDIKONTAR Maurice KOR	Professeur	*Vice-Doyen Univ. Bamenda*
8	NENWA Justin	Professeur	En poste
9	NGAMENI Emmanuel	Professeur	*DOYEN FS UDs*

10	BABALE née DJAM DOUDOU	Maître de Conférences	*Chargée Mission P.R.*
11	DJOUFAC WOUMFO Emmanuel	Maître de Conférences	En poste
12	KAMGANG YOUBI Georges	Maître de Conférences	En poste
13	KEMMEGNE MBOUGUEM Jean C.	Maître de Conférences	En poste
14	KONG SAKEO	Maître de Conférences	*En poste*
16	NGOMO Horace MANGA	Maître de Conférences	*Vice Chancelor/UB*
17	NJIOMOU C. épse DJANGANG	Maître de Conférences	En poste
18	NJOYA Dayirou	Maître de Conférences	En poste
19	YOUNANG Elie	Maître de Conférences	En poste

20	ACAYANKA Elie	Chargé de Cours	En poste
21	BELIBI BELIBI Placide Désiré	Chargé de Cours	CS/ ENS Bertoua
22	CHEUMANI YONA Arnaud M.	Chargé de Cours	En poste
23	EMADACK Alphonse	Chargé de Cours	En poste
24	KENNE DEDZO GUSTAVE	Chargé de Cours	En poste
24	KOUOTOU DAOUDA	Chargé de Cours	En poste
25	MAKON Thomas Beauregard	Chargé de Cours	En poste
26	MBEY Jean Aime	Chargé de Cours	En poste
27	NCHIMI NONO KATIA	Chargé de Cours	En poste
28	NDI NSAMI Julius	Chargé de Cours	En poste
29	NEBA nee NDOSIRI Bridget NDOYE	Chargée de Cours	Inspecteur de Service MINFEM
30	NYAMEN Linda Dyorisse	Chargée de Cours	En poste
31	PABOUDAM GBAMBIE A.	Chargée de Cours	En poste
32	TCHAKOUTE KOUAMO Hervé	Chargé de Cours	En poste

5- DÉPARTEMENT DE CHIMIE ORGANIQUE (CO) (32)

1	DONGO Etienne	Professeur	Vice-Doyen / PSAA
2	GHOGOMU TIH Robert Ralph	Professeur	Dir. IBAF/UDS
3	NGOUELA Silvère Augustin	Professeur	En poste
4	NKENGFACK Augustin Ephreïm	Professeur	Chef de Département
5	NYASSE Barthélemy	Professeur	*Directeur/UN*
6	PEGNYEMB Dieudonné Emmanuel	Professeur	*Directeur/ MINESUP*

7	WANDJI Jean	Professeur	En poste

8	Alex de Théodore ATCHADE	Maître de Conférences	*DEPE/ Rectorat/UYI*
9	EYONG Kenneth OBEN	Maître de Conférences	*Chef Service DPER*
10	FOLEFOC Gabriel NGOSONG	Maître de Conférences	*En poste*
11	KEUMEDJIO Félix	Maître de Conférences	En poste
12	KEUMOGNE Marguerite	Maître de Conférences	En poste
13	KOUAM Jacques	Maître de Conférences	En poste
14	MBAZOA née DJAMA Céline	Maître de Conférences	En poste
15	MKOUNGA Pierre	Maître de Conférences	En poste
16	NGO MBING Joséphine	Maître de Conférences	Sous/Direct. MINERESI
17	NOUNGOUE TCHAMO Diderot	Maître de Conférences	En poste
18	TABOPDA KUATE Turibio	Maître de Conférences	En poste
19	TCHOUANKEU Jean-Claude	Maître de Conférences	*Doyen /FS/ UYI*
20	TIH née NGO BILONG E. Anastasie	Maître de Conférences	En poste
21	YANKEP Emmanuel	Maître de Conférences	En poste

22	AMBASSA Pantaléon	Chargé de Cours	En poste
23	FOTSO WABO Ghislain	Chargé de Cours	En poste
24	KAMTO Eutrophe Le Doux	Chargé de Cours	En poste
25	MVOT AKAK CARINE	Chargé de Cours	En poste
26	NGOMO Orléans	Chargée de Cours	En poste
27	NGONO BIKOBO Dominique Serge	Chargé de Cours	En poste
28	NOTE LOUGBOT Olivier Placide	Chargé de Cours	Chef Service/MINESUP
29	OUAHOUO WACHE Blandine M.	Chargée de Cours	En poste
30	TAGATSING FOTSING Maurice	Chargé de Cours	En poste
31	ZONDENDEGOUMBA Ernestine	Chargée de Cours	En poste

32	NGNINTEDO Dominique	Assistant	En poste

6- DÉPARTEMENT D'INFORMATIQUE (IN) (26)

1	ATSA ETOUNDI Roger	Professeur	*Chef Div.MINESUP*
2	FOUDA NDJODO Marcel Laurent	Professeur	*Chef Dpt ENS/Chef IGA.MINESUP*
3	TCHUENTE Maurice	Professeur	*PCA UYII*

4	NDOUNDAM Réné	Maître de Conférences	En poste

5	AMINOU Halidou	Chargé de Cours	En poste
6	DJAM Xaviera YOUHEP KIMBI	Chargé de Cours	En Poste
7	KOUOKAM KOUOKAM E. A.	Chargé de Cours	En poste
8	MELATAGIA YONTA Paulin	Chargé de Cours	En poste
9	MOTO MPONG Serge Alain	Chargé de Cours	En poste
10	TAPAMO Hyppolite	Chargé de Cours	En poste
11	ABESSOLO ALO'O Gislain	Chargé de Cours	En poste
12	KAMGUEU Patrick Olivier	Chargé de Cours	En poste
13	MONTHE DJIADEU Valery M.	Chargé de Cours	En poste
14	OLLE OLLE Daniel Claude Delort	Chargé de Cours	C/D Enset. Ebolawa

15	TINDO Gilbert	Chargé de Cours	En poste
16	TSOPZE Norbert	Chargé de Cours	En poste
17	WAKU KOUAMOU Jules	Chargé de Cours	En poste

18	BAYEM Jacques Narcisse	Assistant	En poste
19	DOMGA KOMGUEM Rodrigue	Assistant	En poste
20	EBELE Serge	Assistant	En poste
21	HAMZA Adamou	Assistant	En poste
22	JIOMEKONG AZANZI Fidel	Assistant	En poste
23	KAMDEM KENGNE Christiane	Assistante	En poste
24	MAKEMBE. S . Oswald	Assistant	En poste
25	MEYEMDOU Nadège Sylvianne	Assistante	En poste
26	NKONDOCK. MI. BAHANACK.N.	Assistant	En poste

7- DÉPARTEMENT DE MATHÉMATIQUES (MA) (28)

1	BITJONG NDOMBOL	Professeur	*En poste*
2	DOSSA COSSY Marcel	Professeur	En poste

3	AYISSI Raoult Domingo	Maître de Conférences	Chef de Département
4	EMVUDU WONO Yves S.	Maître de Conférences	*CD Info/ Chef division MINESUP*
5	NKUIMI JUGNIA Célestin	Maître de Conférences	En poste
6	NOUNDJEU Pierre	Maître de Conférences	En poste
7	TCHAPNDA NJABO Sophonie B.	Maître de Conférences	Directeur/AIMS Rwanda

8	AGHOUKENG JIOFACK Jean Gérard	Chargé de Cours	Chef Cellule MINPLAMAT
9	CHENDJOU Gilbert	Chargé de Cours	En poste
10	DJIADEU NGAHA Michel	Chargé de Cours	En poste
11	DOUANLA YONTA Herman	Chargé de Cours	En poste
12	FOMEKONG Christophe	Chargé de Cours	En poste
13	KIANPI Maurice	Chargé de Cours	En poste
14	KIKI Maxime Armand	Chargé de Cours	En poste
15	MBAKOP Guy Merlin	Chargé de Cours	En poste
16	MBANG Joseph	Chargé de Cours	En poste
17	MBEHOU Mohamed	Chargé de Cours	En poste
18	MBELE BIDIMA Martin Ledoux	Chargé de Cours	En poste
19	MENGUE MENGUE David Joe	Chargé de Cours	En poste
20	NGUEFACK Bernard	Chargé de Cours	En poste
21	NIMPA PEFOUNKEU Romain	Chargée de Cours	En poste
22	POLA DOUNDOU Emmanuel	Chargé de Cours	En poste
23	TAKAM SOH Patrice	Chargé de Cours	En poste
24	TCHANGANG Roger Duclos	Chargé de Cours	En poste
25	TCHOUNDJA Edgar Landry	Chargé de Cours	En poste
26	TETSADJIO TCHILEPECK M. E.	Chargée de Cours	En poste
27	TIAYA TSAGUE N. Anne-Marie	Chargée de Cours	En poste
28	MBIAKOP Hilaire George	Assistant	En poste

8- DÉPARTEMENT DE MICROBIOLOGIE (MIB) (12)			
1	ESSIA NGANG Jean Justin	Professeur	DRV/IMPM
2	ETOA François Xavier	Professeur	Chef de Département/FS/UYI Recteur Université de Douala
3	BOYOMO ONANA	Maître de Conférences	En poste
4	NWAGA Dieudonné M.	Maître de Conférences	En poste
5	NYEGUE Maximilienne Ascension	Maître de Conférences	En poste
6	RIWOM Sara Honorine	Maître de Conférences	En poste
7	SADO KAMDEM Sylvain Leroy	Maître de Conférences	En poste
8	ASSAM ASSAM Jean Paul	Chargé de Cours	En poste
9	BODA Maurice	Chargé de Cours	En poste
10	BOUGNOM Blaise Pascal	Chargé de Cours	En poste
11	ESSONO OBOUGOU Germain G.	Chargé de Cours	En poste
12	NJIKI BIKOÏ Jacky	Chargée de Cours	En poste
13	TCHIKOUA Roger	Chargé de Cours	En poste

9. DEPARTEMENT DE PYSIQUE(PHY) (40)			
1	BEN- BOLIE Germain Hubert	Professeur	En poste
2	ESSIMBI ZOBO Bernard	Professeur	En poste
3	KOFANE Timoléon Crépin	Professeur	En poste
4	NDJAKA Jean Marie Bienvenu	Professeur	Chef de Département
5	NJANDJOCK NOUCK Philippe	Professeur	*Sous Directeur/ MINRESI*
6	NJOMO Donatien	Professeur	En poste
7	PEMHA Elkana	Professeur	En poste
8	TABOD Charles TABOD	Professeur	Doyen Univ/Bda
9	TCHAWOUA Clément	Professeur	En poste
10	WOAFO Paul	Professeur	En poste
	BIYA MOTTO Frédéric	Maître de Conférences	DG/HYDRO Mekin
14	BODOBertrand	Maître de Conférences	En poste
12	DJUIDJE KENMOE épouse ALOYEM	Maître de Conférences	En poste
15	EKOBENA FOUDA Henri Paul	Maître de Conférences	*Chef Division. UN*
16	EYEBE FOUDA Jean sire	Maître de Conférences	En poste
17	FEWO Serge Ibraïd	Maître de Conférences	En poste
18	HONA Jacques	Maître de Conférences	En poste
19	MBANE BIOUELE César	Maître de Conférences	En poste
20	NANA ENGO Serge Guy	Maître de Conférences	Director/Students/Affairs. UB
21	NANA NBENDJO Blaise	Maître de Conférences	En poste
22	NOUAYOU Robert	Maître de Conférences	En poste
23	SAIDOU	Maître de Conférences	Sous Directeur/Minresi

24	SIEWE SIEWE Martin	Maître de Conférences	En poste
25	SIMO Elie	Maître de Conférences	En poste
26	VONDOU Derbetini Appolinaire	Maître de Conférences	En poste
27	WAKATA née BEYA Annie	Maître de Conférences	*Sous Directeur/ MINESUP*
28	ZEKENG Serge Sylvain	Maître de Conférences	En poste

29	ABDOURAHIMI	Chargé de Cours	En poste
30	EDONGUE HERVAIS	Chargé de Cours	En poste
31	ENYEGUE A NYAM épse BELINGA	Chargée de Cours	En poste
32	FOUEDJIO David	Chargé de Cours	Chef Cell. MINADER
33	MBINACK Clément	Chargé de Cours	En poste
34	MBONO SAMBA Yves Christian U.	Chargé de Cours	En poste
35	MELI'I Joelle Larissa	Chargée de Cours	*En poste*
36	MVOGO ALAIN	Chargé de Cours	*En poste*
37	NDOP Joseph	Chargé de Cours	En poste
38	OBOUNOU Marcel	Chargé de Cours	DA/Univ Inter Etat/Sangmalima
39	WOULACHE Rosalie Laure	Chargée de Cours	En poste

40	CHAMANI Roméo	Assistant	En poste

10- DÉPARTEMENT DE SCIENCES DE LA TERRE (ST) (43)

1	BITOM Dieudonné	Professeur	*Doyen / FASA / UDs*
2	FOUATEU Rose épse YONGUE	Professeur	En poste
3	KAMGANG Pierre	Professeur	En poste
4	MEDJO EKO Robert	Professeur	*Conseiller Technique/UYII*
5	NDJIGUI Paul Désiré	Professeur	Chef de Département
6	NKOUMBOU Charles	Professeur	En poste
7	NZENTI Jean-Paul	Professeur	En poste

8	ABOSSOLO née ANGUE Monique	Maître de Conférences	*Vice-Doyen / DRC*
9	GHOGOMU Richard TANWI	Maître de Conférences	CD/UMa
10	MOUNDI Amidou	Maître de Conférences	*CT/ MINIMDT*
11	NDAM NGOUPAYOU Jules-Remy	Maître de Conférences	En poste
12	NGOS III Simon	Maître de Conférences	DAAC/Uma
13	NJILAH Isaac KONFOR	Maître de Conférences	En poste
14	ONANA Vincent Laurent	Maître de Conférences	En poste
15	BISSO Dieudonné	Maître de Conférences	*Directeur/Projet Barrage Memve'ele*
16	EKOMANE Emile	Maître de Conférences	*En poste*
17	GANNO Sylvestre	Maître de Conférences	En poste
18	NYECK Bruno	Maître de Conférences	En poste
19	TCHOUANKOUE Jean-Pierre	Maître de Conférences	En poste
20	TEMDJIM Robert	Maître de Conférences	En poste
21	YENE ATANGANA Joseph Q.	Maître de Conférences	*Chef Div. /MINTP*
22	ZO'O ZAME Philémon	Maître de Conférences	*DG/ART*

23	ANABA ONANA Achille Basile	Chargé de Cours	*En poste*

24	BEKOA Etienne	Chargé de Cours	*En poste*
25	ELISE SABABA	Chargé de Cours	En poste
26	ESSONO Jean	Chargé de Cours	*En poste*
27	EYONG JOHN TAKEM	Chargé de Cours	En poste
28	FUH Calistus Gentry	Chargé de Cours	*Sec. D'Etat/MINMIDT*
29	LAMILEN BILLA Daniel	Chargé de Cours	En poste
30	MBESSE CECILE OLIVE	Chargée de Cours	En poste
31	MBIDA YEM	Chargé de Cours	*En poste*
32	METANG Victor	Chargé de Cours	En poste
33	MINYEM Dieudonné-Lucien	Chargé de Cours	*CD/Uma*
34	MOUAFO Lucas	Chargé de Cours	En poste
35	NGO BELNOUN Rose Noël	Chargée de Cours	En poste
37	NGO BIDJECK Louise Marie	Chargée de Cours	En poste
38	NGUEUTCHOUA Gabriel	Chargé de Cours	CEA/MINRESI
39	NOMO NEGUE Emmanuel	Chargé de Cours	En poste
36	NTSAMA ATANGANA Jacqueline	Chargé de Cours	En poste
40	TCHAKOUNTE J. épse NOUMBEM	Chargée de Cours	*Chef.cell / MINRESI*
41	TCHAPTCHET TCHATO De P.	Chargé de Cours	En poste
42	TEHNA Nathanaël	Chargé de Cours	En poste
43	TEMGA Jean Pierre	Chargé de Cours	En poste

Répartition chiffrée des Enseignants de la Faculté des Sciences de l'Université de Yaoundé I

NOMBRE D'ENSEIGNANTS					
DÉPARTEMENT	**Professeurs**	**Maîtres de Conférences**	**Chargés de Cours**	**Assistants**	**Total**
BCH	5 (1)	12 (6)	19 (11)	1 (1)	**37 (19)**
BPA	12 (1)	10 (5)	20 (7)	2 (0)	**44 (13)**
BPV	5 (0)	10 (2)	9 (4)	2 (2)	**26 (9)**
CI	9 (1)	9 (2)	14 (3)	0 (0)	**32 (6)**
CO	7 (0)	14 (4)	10 (4)	1 (0)	**32 (8)**
IN	3 (0)	1 (0)	13 (1)	9 (2)	**26 (3)**
MAT	2 (0)	4 (1)	19 (1)	2 (0)	**27 (2)**
MIB	2 (0)	5 (2)	5 (1)	0 (0)	**12 (3)**
PHY	10 (0)	17 (2)	11 (3)	1 (0)	**39 (5)**
ST	7 (1)	15 (1)	21 (5)	1 (0)	**43 (7)**
Total	**62 (4)**	**97 (25)**	**141 (39)**	**19 (5)**	**318 (75)**

Soit un total de **318 (**75**)** dont :

- Professeurs **62 (4)**
- Maîtres de Conférences **97 (**25**)**
- Chargés de Cours **141 (39)**
- Assistants **19** (5**)**

( ) = Nombre de Femmes