Recommandation sociale et locale basée sur la confiance

Simon Meyffret [1,2], **Lionel Médini** [1,3], **Frédérique Laforest** [1,4]

1. Université de Lyon

2. INSA-Lyon, CNRS, LIRIS, UMR5205, F-69621, France
simon.meyffret@liris.cnrs.fr

3. Université Lyon 1, CNRS, LIRIS, UMR5205, F-69622, France
lionel.medini@liris.cnrs.fr

4. Université de Saint-Etienne, Télécom Saint-Etienne, LT2C, F-42000, France
frederique.laforest@telecom-st-etienne.fr

RÉSUMÉ. *Les systèmes de recommandation sont largement utilisés. Avec les réseaux sociaux, ces systèmes tiennent désormais compte des relations d'amitié ou de confiance entre les utilisateurs. Dans cet article, nous proposons un algorithme de recommandation respectant la confidentialité des données et pouvant être implanté sur des architectures pair à pair. Notre approche n'a besoin d'aucune connaissance globale du réseau, limite l'échange de données aux relations d'amitié et de confiance entre les pairs et évite la diffusion des données privées sur le réseau. Nous avons réalisé une évaluation de différents algorithmes de recommandation, afin de comparer notre proposition aux algorithmes centralisés. Pour cela, nous avons défini des mesures spécifiques de précision des résultats et identifié l'ensemble des connaissances nécessaires par algorithme. Nous avons mené deux séries de tests de simulation qui sont présentés et discutés.*

ABSTRACT. *Recommender Systems are widely used to achieve a constantly growing variety of services. Along with social networks, recommendation systems that take into account friendship or trust between users have emerged. In this article, we propose a recommendation algorithm that respects data privacy constraints in order to be implemented on P2P architecture. It does not need any global knowledge on the network, it limits data exchange to trusted relations and protects private data from being spread over the network.*
We present an evaluation of different recommendation algorithms, to compare our to centralized ones, such as those used in online marketing services. Several concerns are considered: specific accuracy measures are defined, as well as the amount of required knowledge on the overall dataset. We ran two series of simulation tests that are herein presented and discussed.

MOTS-CLÉS : *recommandation basée sur la confiance, P2P, protection des données personnelles.*

KEYWORDS: *trust-based recommender systems, P2P, data privacy protection.*

DOI:10.3166/DN.15.1.33-56

1. Introduction

Internet et le World Wide Web sont aujourd'hui des outils incontournables utilisés dans toutes les activités, qu'elles soient professionnelles, éducatives, associatives, ou personnelles. La multitude des fournisseurs d'information et des sujets traités a entraîné le problème bien connu de surcharge cognitive (Eppler, Mengis, 2004) où l'utilisateur se trouve démuni devant la profusion d'informations disponibles. Les sites gérant de nombreuses références (d'objets ou de sujets) fournissent aujourd'hui non seulement des outils de recherche classique par mots-clés mais également des systèmes de recommandation. Ces systèmes proposent à l'utilisateur une présélection d'objets pouvant l'intéresser. L'objectif est de répondre au mieux à l'utilisateur qui peut alors sélectionner dans cette liste les objets qui lui conviennent, en fonction de ses besoins et de ses préférences.

Afin de fournir une recommandation adaptée, les systèmes de recommandation analysent les profils des utilisateurs et des objets. De façon générale, ils se basent sur les notes déjà fournies par les utilisateurs sur les objets. Ces notes peuvent être fournies explicitement par les utilisateurs ou être calculées par le système en fonction de certaines traces d'utilisation par exemple (articles consultés, nombre de clics, durée...). Afin de prédire les notes manquantes, deux approches sont privilégiées (Balabanović, Shoham, 1997). Les systèmes de recommandation basés sur le contenu (content-based recommender systems) comparent les caractéristiques des objets afin de recommander des objets similaires à ceux déjà appréciés de l'utilisateur. Les approches de filtrage collaboratif (collaborative filtering recommender systems) comparent les utilisateurs en fonction des notes déjà obtenues et agrègent les notes des utilisateurs similaires afin de proposer leurs objets préférés. Des approches hybrides utilisent des algorithmes alliant des comparaisons entre profils d'objets et profils d'utilisateurs. Quoiqu'il en soit, les algorithmes classiques de recommandation requièrent une centralisation des profils des utilisateurs ou des objets, ou du moins d'en avoir une connaissance globale.

Avec la démocratisation d'Internet et le développement des environnements pervasifs, les utilisateurs peuvent demeurer continuellement connectés. Ils peuvent accéder à toutes sortes de ressources, documents, médias, facilement et rapidement, depuis des dispositifs de plus en plus variés. Les smartphones ou tablettes décuplent la connectivité des utilisateurs, qui peuvent dorénavant partager n'importe quelle information n'importe quand, depuis n'importe où. Ces systèmes ubiquitaires ont ainsi augmenté l'étendue des usages de l'informatique dans tous les domaines. Les architectures sous-jacentes à ces nouveaux usages sont toujours réparties, et de plus en plus souvent pair à pair. La centralisation de l'information n'est plus adaptée face à la mobilité croissante des utilisateurs, et à l'émergence des réseaux ad hoc.

Aujourd'hui, grâce à leurs réseaux sociaux, les utilisateurs non seulement consultent de l'information, mais deviennent également plus facilement auteurs en partageant régulièrement des données et des opinions avec leurs amis, leurs familles ou de simples connaissances, membres de leur réseau social. Les systèmes de recommandation ont pris le tournant de ces nouveaux usages, avec l'apparition des systèmes de recom-

mandation basés sur la confiance, ou “trust-based recommender systems”. Ces derniers utilisent les relations de confiance entre utilisateurs afin de prédire les notes à partir des profils “de confiance”. Les algorithmes de recommandation passent ainsi d'une connaissance globale de l'ensemble des utilisateurs à une restriction de l'étendue des informations prises en compte : seules les informations fournies par les amis de confiance d'un utilisateur sont utilisées. Cependant les réseaux de confiance affichent souvent de petits nombres de liens de confiance établis par les utilisateurs, réduisant les capacités de prédiction des algorithmes de recommandation. Pour pallier le manque d'information, les systèmes de recommandation basés sur la confiance propagent la confiance à travers les liens entre utilisateurs afin d'élargir les capacités de prédiction. Cette propagation nous paraît incompatible avec la notion de confiance communément admise par les utilisateurs car elle induit une extension automatisée du réseau de confiance d'un utilisateur ; elle nécessite par ailleurs une connaissance globale sur les utilisateurs.

Avec les réseaux sociaux, l'affichage d'informations personnelles est devenue une pratique courante ; les questions de confidentialité et de protection des données personnelles divisent les membres des réseaux sociaux et autres internautes. Les dernières évolutions sur ce point de Facebook montrent la tendance : la gestion des droits et des autorisations se veut de plus en plus fine, et différencie de plus en plus le cercle des amis du reste des internautes. On voit également naître de nombreuses initiatives qui tentent de passer d'un réseau social centralisé à un réseau social réparti où les informations sont localement maîtrisées par ses membres, sans être divulguées à un système central omniscient. On peut par exemple citer The Social Web du W3C [1] ou encore le projet Freedom Box [2] dont les objectifs sont de permettre une répartition des nœuds du réseau social, chaque membre du réseau étant maître de son nœud physique et logiciel. Un réseau social pair à pair considère que chaque membre du réseau social est associé à un pair et qu'il est lié aux autres pairs par des liens d'amitié et/ou de confiance définis manuellement.

La prise en compte de la confidentialité apporte des contraintes que les méthodes classiques de recommandation ne peuvent satisfaire dans le contexte de réseaux sociaux pair à pair. De plus, la centralisation de l'information n'est plus adaptée face à la mobilité croissante des utilisateurs et à l'émergence des réseaux ad hoc.

Dans cet article, notre objectif est de proposer un système de recommandation sociale adapté aux architectures décentralisées et pair à pair qui prend en considération les problématiques de confidentialité. Le moteur de recommandation utilisé par un utilisateur est donc un logiciel localement hébergé par le pair considéré. Nous voulons montrer qu'une vision partielle mais judicieusement choisie des profils des utilisateurs est suffisante pour offrir une recommandation pertinente. Notre proposition considère une propagation de scores plutôt que de confiance, reposant sur les liens définis dans

1. http://www.w3.org/2005/Incubator/socialweb/
2. http://wiki.debian.org/FreedomBox

les réseaux sociaux. Nous montrons qu'un tel mode de propagation est approprié pour cette recommandation.

De telles architectures imposent des contraintes qui rendent les approches classiques de recommandation inadaptées. Premièrement, puisque le moteur de recommandation est situé dans un pair, il ne peut accéder à une base de données centralisée contenant tous les profils. Il doit donc utiliser la propagation des données parmi les pairs dans l'architecture décentralisée. Deuxièmement, le système ne doit utiliser comme information que ce que les utilisateurs acceptent de partager, et respecter leur vie privée. Les relations de confiance sont à la base de ce mécanisme : les utilisateurs acceptent de partager leurs informations avec d'autres utilisateurs de confiance. Finalement, les pairs doivent pouvoir se connecter et se déconnecter du réseau instantanément et de manière impromptue sans perturber la recommandation.

Nous utilisons le réseau social des utilisateurs pour propager les scores des objets. Notre algorithme de recommandation est donc fondé sur les liens sociaux et peut être qualifié de "système de recommandation sociale" (ou *social recommendation system*). Dans notre système, seuls les utilisateurs de confiance communiquent entre eux. La confiance (trust) est explicitement définie par les utilisateurs eux-mêmes au sein de leur réseau social. Les profils utilisateurs sont privés et ne peuvent être partagés qu'avec des utilisateurs de confiance. Notre approche est locale : elle n'échange des informations qu'avec les pairs représentant les utilisateurs de confiance ; elle est donc adaptée aux architectures pair à pair.

Après un état de l'art sur les systèmes de recommandation montrant leur inadéquation aux architectures pair à pair, nous présentons dans cet article différentes versions d'un algorithme de recommandation sociale décentralisé. Une évaluation de notre algorithme est ensuite fournie, elle permet de nous comparer aux systèmes présentés dans l'état de l'art en utilisant des mesures classiques et en insistant sur l'importance des profils atypiques, notamment via une mesure de précision prédictive qui accentue l'importance des notes inhabituelles. Enfin, nous discutons de la précision de cette approche par rapport aux systèmes existants.

2. État de l'art

2.1. Recommandation

Le filtrage collaboratif utilise les profils des utilisateurs afin de calculer un coefficient de similarité entre tous les utilisateurs et d'agréger les notes provenant d'utilisa-

teurs similaires. Deux fonctions d'agrégation sont en général utilisées (Adomavicius, Tuzhilin, 2005) :

$$r_{a,i} = \frac{1}{|A_i|} \times \sum_{a' \in A_i} r_{a',i} \quad (1)$$

$$r_{a,i} = \overline{r_a} + \frac{\sum_{a' \in A_i} sim(a,a') \times (r_{a',i} - \overline{r_{a'}})}{\sum_{a' \in A_i} sim(a,a')} \quad (2)$$

où : $sim(a,a')$ est la similarité entre a et a' ; $r_{a,i}$ est la note (rating) donnée par a sur l'objet i ; $\overline{r_a}$ (respectivement $\overline{r_{a'}}$) est la moyenne de toutes les notes (ratings) données par a (resp. a') ; A_i est l'ensemble des utilisateurs qui ont noté l'objet i. On utilise généralement la différence à la moyenne afin de prendre en compte le comportement général de l'utilisateur : a-t-il tendance a être généreux ou strict dans sa notation par exemple. sim est traditionnellement un coefficient de corrélation de Pearson entre les notes des utilisateurs (Breese *et al.*, 1998).

Nous appelons *GlobalMean* l'algorithme de recommandation (*scorer*) utilisant l'équation 1 pour prédire un score. *GlobalMean* renvoie la moyenne de toutes les notes obtenues par l'objet. Ce *scorer* ne personnalise donc pas la réponse en fonction de l'utilisateur. Cependant il offre la meilleure couverture possible en filtrage collaboratif puisqu'il suffit que l'objet ait au moins une note fournie par n'importe quel utilisateur pour prédire un score.

Nous appelons *GlobalCF* l'approche pure de filtrage collaboratif utilisant l'équation 2 pour prédire un score. Elle utilise tous les profils pour calculer la similarité entre les utilisateurs, ce qui est coûteux en temps de calcul et se fait généralement hors-ligne. Les notes des utilisateurs similaires sont ensuite agrégées. Bien qu'ils soient utilisés avec succès par beaucoup de sites de commerce en ligne, les systèmes utilisant le filtrage collaboratif ne sont pas robustes face aux utilisateurs malveillants (Mehta *et al.*, 2007) ou qui ont des goûts particuliers (Schafer *et al.*, 2007). Ils ont également du mal à faire face à la faible densité des notes existantes (Lee, Brusilovsky, 2009) et plus particulièrement à gérer les *cold start users* : nouveaux utilisateurs venant juste d'arriver dans le système et ayant noté peu d'objets (généralement moins de 5).

De plus, ils nécessitent une connaissance globale sur tous les utilisateurs afin de calculer les similarités. Les données sont agrégées par un système omniscient. Ceci est possible dans un environnement centralisé, mais ne l'est plus si l'on veut préserver la vie privée de l'utilisateur et si l'on utilise une architecture décentralisée, voire pair à pair.

Les systèmes de recommandation basés sur la confiance utilisent les réseaux de confiance entre utilisateurs (O'Donovan, Smyth, 2005 ; Massa, Avesani, 2007a ; Ma *et al.*, 2009). Différentes mesures de confiance peuvent être utilisées. Les mesures globales comme PageRank (Page *et al.*, 1999) ou les réactions sur eBay sont répandues, elles impliquent une unique valeur de confiance globale par utilisateur. À l'opposé, les mesures locales définissent une valeur de confiance entre paires/couples d'utilisateurs. De plus, la confiance peut être subjective ou objective (réputation). Les mesures

locales subjectives sont définies par chaque utilisateur pour indiquer ceux en qui il a confiance. C'est une mesure centrée sur et contrôlée par l'utilisateur, qui assure une prise en compte de la confidentialité. (McKnight, Chervany, 1996 ; Hasan, 2010) proposent plusieurs définitions pour la confiance. Dans ce papier, la confiance (ou trust) est définie comme la croyance qu'a un utilisateur dans l'utilité des informations qu'un autre utilisateur peut lui fournir (Lee, Brusilovsky, 2009).

Les systèmes de recommandation basés sur la confiance résolvent les problèmes présentés ci-dessus en utilisant la confiance à la place de la similarité dans l'équation 2. Les cold start users n'ont plus besoin d'avoir des notes pour recevoir une prédiction, ils ont besoin de faire confiance à d'autre personnes (Massa, Avesani, 2007a ; Pitsilis, Knapskog, 2009). La sécurité est améliorée puisque les utilisateurs connaissent leurs voisins directs.

Cependant ils doivent faire face à des réseaux de confiance clairsemés : dans Epinions (Richardson, Domingos, 2002), un réseau de confiance utilisé pour la recommandation d'objets divers, un utilisateur fait confiance en moyenne à huit utilisateurs. Pour résoudre ce problème, ils propagent la confiance dans le réseau, créant de nouvelles relations entre utilisateurs. La confiance est considérée comme transitive afin de pouvoir propager les liens de confiance (Hang *et al.*, 2009 ; Massa, Avesani, 2007a): si a fait confiance à b et b fait confiance à c, alors un nouveau lien de confiance est créé entre a et c.

MoleTrust fonctionne de cette manière (Massa, Avesani, 2007b). Il prédit les valeurs de confiance possibles entre les utilisateurs en propageant graduellement les liens de confiance existant dans le réseau des utilisateurs, jusqu'à une profondeur k. Si plusieurs liens relient deux utilisateurs, leur moyenne est utilisée. Cette approche nécessite une connaissance locale étendue du réseau de confiance, puisque tous les chemins entre deux utilisateurs distants de k au plus sont connus.

(Golbeck, 2005) propose TidalTrust, qui est similaire à MoleTrust, en prenant en compte les chemins les plus courts via un parcours en largeur modifié dans le réseau de confiance. De la même manière, il nécessite une connaissance locale étendue.

Le clustering dans les réseaux sociaux est aussi utilisé pour améliorer la recommandation (DuBois *et al.*, 2009), mais là encore de nouvelles relations de confiance sont implicitement créées entre utilisateurs.

TrustWalker (Jamali, Ester, 2009) est une approche hybride entre celles basées sur la confiance et sur la similarité entre objet (*item-based collaborative filtering*). Il agrège les recommandations des utilisateurs de confiance jusqu'à une profondeur k, sans toucher au réseau de confiance. TrustWalker utilise aussi la similarité entre objets pour retourner les notes des objets similaires. Cela améliore grandement la couverture et traite le problème des réseaux clairsemés, cependant cela nécessite une connaissance globale des notes existantes et n'est pas compatible avec une architecture pair à pair.

Nous pensons que la confiance doit être contrôlée par l'utilisateur et doit être explicitement définie par lui. De plus les approches basées sur la confiance décrites ci-dessus propagent généralement la confiance jusqu'à une profondeur de 6, ce qui, vis-à-vis du *small world effect* (Milgram, 1967), est équivalent à propager dans tout le réseau social. Nous proposons une approche basée sur la confiance mais limitant la propagation dans le réseau à une profondeur de $k = 3$, afin de limiter l'inondation du réseau social. De plus, notre approche ne crée pas de nouveaux liens et laisse à l'utilisateur un contrôle total dans le choix de ses partages d'information.

2.2. Évaluation

(Herlocker *et al.*, 2004) indiquent plusieurs mesures permettant d'évaluer les systèmes de recommandation, incluant des mesures de précision de la prédiction et du classement.

Pour mesurer la précision de la prédiction, on compare la note réelle à la note prédite. La principale mesure est la *Mean Absolute Error* (éq. 3). p_n est la note prédite pour un couple (utilisateur, objet), r_n est la note existante pour ce couple et N est le nombre total de notes. La *Root Mean Square Error* (RMSE) accentue les erreurs larges (éq. 4). Quelle que soit la mesure, plus l'erreur est basse, plus la note prédite est proche de la note réelle et donc meilleure est la recommandation.

$$MAE = \frac{\sum_{n=1}^{N} |p_n - r_n|}{N} \tag{3}$$

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N} (p_n - r_n)^2}{N}} \tag{4}$$

La précision du classement compare le rang prédit p de l'objet avec le rang réel r. La corrélation de Pearson (PCC) mesure la relation linéaire entre les notes des objets. La corrélation des rangs de Spearman (SRCC) mesure la relation entre les rangs des objets. La même formule (Eq. 5) est appliquée dans les deux cas mais Pearson utilise les notes des objets alors que Spearman utilise les rangs des objets (Herlocker *et al.*, 1999). Une grande corrélation indique une meilleure recommandation. Dans l'équation 5, r_n indique la $n$ième note réelle ($n$ième rang pour Spearman), p_n indique la $n$ième note prédite ($n$ième rang pour Spearman), $\bar{r}$ et $\bar{p}$ indiquent respectivement la moyenne des notes réelles et prédites (rang réel et prédits pour Spearman).

$$\rho = \frac{\sum_{n=1}^{N} (r_n - \bar{r})(p_n - \bar{p})}{N \times \sqrt{\sum_{n=1}^{N} (r_n - \bar{r})^2} \times \sqrt{\sum_{n=1}^{N} (p_n - \bar{p})^2}} \tag{5}$$

La couverture est également un bon indicateur de qualité des moteurs de recommandation. Elle indique combien de notes le système a pu prédire, quelle que soit la prédiction. Nous appelons *couverture* le ratio des scores prédits (entre 0 et 1).

Les notes prennent leur valeur dans un ensemble limité de possibilités (*e.g.* $[\![1, 5]\!]$), donc un système de recommandation peut être vu comme un classifieur où les classes sont les notes possibles. La précision, le rappel, la f-mesure, les taux de vrais ou faux positifs sont des mesures utilisées dans ce domaine. Cependant nos classes ne sont pas indépendantes : prédire un 2 pour une note valant 1 est meilleur que prédire un 3.

Dans la recherche d'information, un document est pertinent ou non en fonction d'une requête. La précision et le rappel sont fréquemment utilisés. En recommandation, il n'existe aucune requête, ces mesures sont donc rarement adaptées. Cependant, (Basu *et al.*, 1998) utilise le premier quartile des objets les mieux notés et définit cet ensemble comme l'ensemble pertinent, le reste étant non pertinent. Alors, la précision et le rappel sont calculés avec les scores prédits. Mais comme les notes sont dans un ensemble restreint (en général entre 1 et 5), la plupart des objets ont la même note, il est alors difficile de les départager dans un classement contenant beaucoup d'ex æquo.

(Jamali, Ester, 2009) propose cependant une version adaptée de la f-mesure, en définissant la précision comme étant :

$$Précision = 1 - \frac{RMSE}{4} \tag{6}$$

Cette équation est utilisée avec un jeu de données contenant des valeurs comprises entre 1 et 5 : l'erreur maximale valant 4, la précision définie ci-dessus est assurée d'être entre 0 et 1. La f-mesure est alors définie comme suit :

$$FMesure = \frac{2 * Précision \times Couverture}{Précision + Couverture} \tag{7}$$

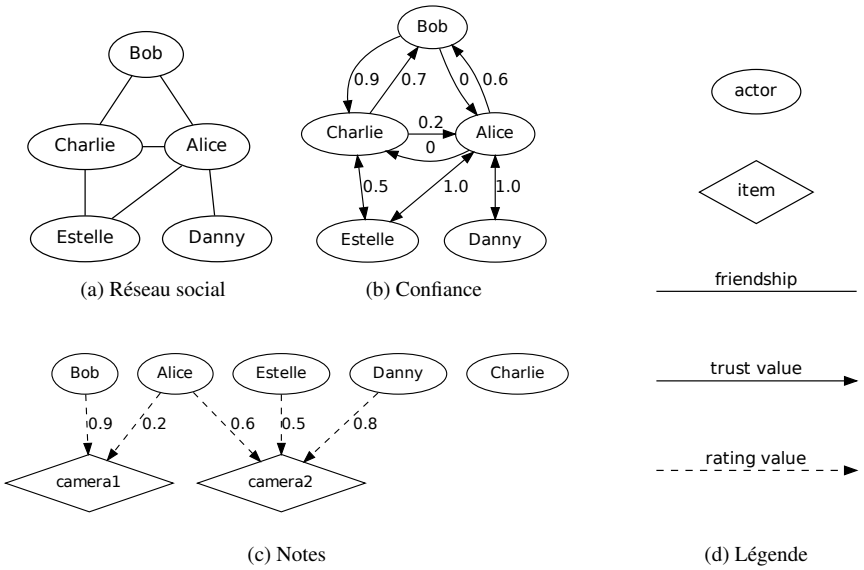
3. Exemple et définitions

Nous appelons ici réseau social tout ensemble d'acteurs et de relations définies entre ces acteurs. Un acteur est une entité sociale (individu, groupe ou entreprise) (Wasserman, Faust, 1994). Dans ce papier, un acteur fait référence à n'importe quelle entité sociale dans le réseau, notamment l'utilisateur du système de recommandation. Nous appelons "amis" deux acteurs directement reliés dans le réseau social.

Nous illustrons cet article avec un exemple simple. Cinq acteurs – Alice, Bob, Charlie, Danny et Estelle – partagent un réseau social contenant des relations entre eux (figure 1a). Les valeurs de confiance sont comprises entre 0 et 1. Elles sont données figure 1b. Une relation de confiance est une relation orientée et valuée d'un acteur vers un de ses amis.

La figure 1c montre les notes des acteurs sur les objets. Nous considérons dans notre exemple deux objets : $camera_1$ and $camera_2$. Les notes sont représentées par une valeur sur le lien entre un acteur et un objet. Les acteurs ne notent évidemment pas les objets de la même manière : Bob adore $camera_1$ alors qu'Alice préfère $camera_2$. Danny et Estelle n'ont noté que $camera_2$ et Charlie n'a rien noté.

Dans notre travail, la recommandation des caméras prend en compte les notes des amis et la confiance entre eux. Pour cela, nos systèmes de recommandation sociale propagent les scores à travers le réseau social afin de prédire le score de Bob pour $camera_2$, ceux de Danny et d'Estelle pour $camera_1$ et ceux de Charlie pour les deux caméras.



(a) Réseau social (b) Confiance (c) Notes (d) Légende

Figure 1. Exemple de réseau social, de confiance et de notes

Soit A l'ensemble des acteurs et a un acteur. Soit I l'ensemble des objets et i un objet. Dans notre exemple:

- $A = \{Alice, Bob, Charlie, Danny, Estelle\}$
- $I = \{camera_1, camera_2\}$

La relation de confiance entre un acteur a et son ami f est noté $t_{a,f}$ avec $t_{a,f} \in [0, 1]$. Cette relation est seulement définie entre amis. Soit T l'ensemble des triplets de confiance $(a, f, t_{a,f})$ avec $(a, f) \in A^2$ et $t_{a,f} \in [0, 1]$. À chaque couple $(a, f) \in A^2$ peut être associée zéro ou une valeur de confiance $t_{a,f}$. Plus $t_{a,f}$ est élevée, plus l'acteur a a confiance en l'avis de f.

Cette relation n'est ni symétrique ni transitive, notre approche n'infère pas de nouvelles relations de confiance à partir de celles existantes :

$$(a, f, t_{a,f}) \in T \not\Rightarrow (f, a, t_{f,a}) \in T$$

$$(a, b, t_{a,b}) \in T \wedge (b, c, t_{b,c}) \in T \not\Rightarrow (a, c, t_{a,c}) \in T$$

Dans la figure 1b Bob a confiance dans les recommandations de Charlie à une hauteur de 0.9 mais Charlie ne fait confiance à Bob qu'à 0.7. L'amitié n'implique pas

forcément une relation de confiance entre deux acteurs : Alice et Charlie sont amis mais ne partagent qu'une faible confiance dans leurs recommandations respectives.

Soit T_a l'ensemble des amis de confiance de a : $T_a = \{f \in A | \exists (a, f, t_{a,f}) \in T \wedge t_{a,f} > 0\}$. Ce qui donne dans notre exemple :

- $T = \{(Alice, Bob, 0.6), (Bob, Alice, 0), \cdots, (Estelle, Charlie, 0.5)\}$
- $T_{Alice} = \{Bob, Danny, Estelle\}$
- $T_{Estelle} = \{Alice, Charlie\}$

Le graphe de notes entre les acteurs et les objets (figure 1c) est un graphe bipartite. Les notes sont des liens entre les acteurs (première partition) et les objets (seconde partition). De plus chaque acteur est la racine d'un arbre de profondeur 1 qui contient tous les objets notés par cet acteur. La note donnée par l'acteur a envers l'objet i est notée $r_{a,i}$. Soit R l'ensemble des triplets de note $(a, i, r_{a,i})$ avec $a \in A$, $i \in I$ et $r_{a,i} \in [0, 1]$. 0 signifie que l'acteur n'a pas aimé cet objet. 1 signifie qu'il l'a aimé. Pour chaque couple $(a, i) \in A \times I$ peut être associée zéro ou une note $r_{a,i}$.

Soit $A_i = \{a \in A | \exists (a, i, r_{a,i}) \in R\}$ l'ensemble des acteurs qui ont noté l'objet i. Ce qui donne dans notre exemple :

- $R = \{(Bob, camera_1, 0.9), (Alice, camera_1, 0.2), \cdots, (Danny, camera_2, 0.8)\}$
- $A_{camera_1} = \{Alice, Bob\}$
- $A_{camera_2} = \{Alice, Estelle, Danny\}$

4. Social scoring

Dans cet article, nous proposons d'utiliser un calcul basé sur le réseau social pour prédire les notes manquantes entre un acteur et un objet. Nous appelons ces notes manquantes des scores. Différents algorithmes peuvent être utilisés pour prédire des scores à partir des relations existantes dans un réseau social. Nous présentons ici incrémentalement un algorithme de recommandation fondé sur cinq étapes.

Notre approche est basée sur la confiance, puisqu'elle utilise un réseau de confiance déjà construit entre les utilisateurs du système. Notre approche est également une approche sociale, car elle ne modifie pas le réseau social ou de confiance et laisse aux utilisateurs un contrôle total sur leurs relations de confiance.

4.1. Immediate social scoring

Afin de calculer un score pour un acteur en particulier, l'*immediate social scoring* utilise les notes des amis de l'acteur. Aucune donnée n'est transmise entre acteurs s'ils ne sont pas amis. C'est un moyen d'assurer la confidentialité des notes des acteurs.

L'*immediate social score* d'un objet i par un acteur a est sa note si elle existe, ou la combinaison des notes des amis de a pondérée par la confiance sinon. Pour rappel, T_a est l'ensemble des amis f de a avec une confiance strictement positive ($t_{a,f} > 0$). Si l'acteur n'a aucun ami qui a noté cet objet i, alors le score n'est pas calculable ($\perp$).

$$score_1 : A \times I \to [0,1] \cup \{\perp\}$$
$$(a,i) \mapsto score_1(a,i)$$
$$score_1(a,i) = \begin{cases} r_{a,i} & \text{si } \exists r_{a,i} \\ \frac{\sum_{f \in T_a \cap A_i} t_{a,f} \times r_{f,i}}{\sum_{f \in T_a \cap A_i} t_{a,f}} & \text{si } \nexists r_{a,i} \wedge T_a \cap A_i \neq \emptyset \\ \perp & \text{sinon} \end{cases} \quad (8)$$

Nous pouvons reprendre l'exemple défini section 3 figure 1 et prédire les scores de Charlie (noté C) pour les deux caméras (notées 1 et 2). D'après la formule ci-dessus, nous demandons à ses trois amis Alice (A), Bob (B) et Estelle (E) leurs scores respectifs. Estelle n'ayant pas noté $camera_1$ et Bob n'ayant pas noté $camera_2$, nous avons $T_C \cap A_1 = \{A, B\}$ et $T_C \cap A_2 = \{A, E\}$, ce qui entraîne :

$$score_1(C,1) = \frac{t_{C,A} \times r_{A,1} + t_{C,B} \times r_{B,1}}{t_{C,A} + t_{C,B}} = \frac{0.2 \times 0.2 + 0.7 \times 0.9}{0.2 + 0.7} = 0.74$$
$$score_1(C,2) = \frac{t_{C,A} \times r_{A,2} + t_{C,E} \times r_{E,2}}{t_{C,A} + t_{C,E}} = \frac{0.2 \times 0.6 + 0.5 \times 0.5}{0.2 + 0.5} = 0.53$$

4.2. $k-$Depth social scoring

Cet algorithme étend le précédent en utilisant des amis d'amis jusqu'à une profondeur k au lieu du voisinage immédiat. Ceci simule une transitivité jusqu'à une profondeur k tout en restant anonyme et sécurisé. $score_k$ renvoie la note de l'acteur si elle existe. Sinon, il demande à tous ses amis de renvoyer leurs notes ou de prédire leurs scores, en demandant à leurs amis, etc., jusqu'à une profondeur k.

Pour faciliter les définitions suivantes, nous introduisons $\mathcal{A}^k_{a,l,i}$ comme étant l'ensemble des acteurs f différents de l et amis de a où $score_k(f,a,i)$ est défini :

$$\mathcal{A}^k_{a,l,i} = \{f \in T_a | f \neq l \wedge score_k(f,a,i) \neq \perp\}$$
$$\mathcal{A}^0_{a,l,i} = A_i \cap T_a, \quad \forall l \in A$$

$$score_k : A^2 \times I \to [0,1] \cup \{\perp\}$$
$$(a,l,i) \mapsto score_k(a,l,i)$$
$$score_k(a,l,i) = \begin{cases} r_{a,i} & \text{si } \exists r_{a,i} \\ \frac{\sum_{f \in \mathcal{A}^{k-1}_{a,l,i}} t_{a,f} \times score_{k-1}(f,a,i)}{\sum_{f \in \mathcal{A}^{k-1}_{a,l,i}} t_{a,f}} & \text{si } \nexists r_{a,i} \wedge \mathcal{A}^{k-1}_{a,l,i} \neq \emptyset \\ \perp & \text{sinon} \end{cases} \quad (9)$$

Afin de limiter les cycles dans le calcul de score, nous avons introduit un paramètre indiquant le dernier nœud l de la requête : si a reçoit une demande de score depuis un ami l et si $t_{a,l} > 0$, a ne redemandera pas à l un score pour cet objet. Alors, a devient le dernier nœud de la requête pour ses propres amis, qui ne savent pas que l est l'initiateur de la requête. Cependant, si f est un ami commun à a et l, il pourrait demander à l un score. Mais comme k décroît à chaque saut et que nous limitons sa valeur initiale à cause du *small world effect* (Milgram, 1967), k reste faible (2 ou 3 au maximum). Si l'on veut utiliser $k > 3$, alors un système de cache dans chaque nœud devrait renvoyer $\perp$ si un duplicata de requête arrive plusieurs fois [3]. Les cycles deviennent marginaux et la confidentialité reste préservée puisqu'aucune information n'est partagée entre deux nœuds qui ne sont pas reliés dans le réseau social. Nous posons $score_k(a, i) = score_k(a, a, i)$ pour l'initiateur de la requête.

Le *$k-$depth social scoring* ne partage des données qu'entre amis immédiats. Les scores sont agrégés avant d'être propagés, ainsi le demandeur ne sait pas quels acteurs ont contribué au calcul d'un score.

Reprenons l'exemple illustrant la section précédente. Nous devons d'abord prédire $score_1$ pour les couples (Estelle, $camera_1$) et (Bob, $camera_2$) afin d'introduire leurs valeurs dans le calcul du *k-depth social scoring*. $T_E \cap A_1 = \{A\}$ et $T_B \cap A_2 = \emptyset$, en effet $T_B \cap A_2$ est l'ensemble des amis *de confiance* de Bob ayant noté $camera_2$, Alice est la seule amie de Bob ayant noté $camera_2$ mais la confiance de Bob vers Alice est nulle. Nous pouvons déjà anticiper que la propagation dans le réseau social n'entraînera aucun changement concernant le calcul du score de Charlie pour $camera_2$, puisque Bob va de nouveau retourner $\perp$.

$$
\begin{aligned}
score_1(E,1) &= \frac{t_{E,A} \times r_{A,1}}{t_{E,A}} = r_{A,1} = 0.2 \\
score_2(C,1) &= \frac{t_{C,A} \times r_{A,1} + t_{C,B} \times r_{B,1} + t_{C,E} \times score_1(E,1)}{t_{C,A} + t_{C,B} + t_{C,E}} \\
&= \frac{0.2 \times 0.2 + 0.7 \times 0.9 + 0.5 \times 0.2}{0.2 + 0.7 + 0.5} \\
&= 0.55 \\
score_2(C,2) &= score_1(C,2) = 0.53
\end{aligned}
$$

Dans cet exemple, Danny n'intervient pas dans les calculs de score de Charlie. En effet, Danny a pour seul lien Alice qui a noté les deux caméras. Elle n'a donc

3. Inclure un identifiant unique dans les requêtes n'est toutefois pas un problème trivial dans notre système. Nous ne voulons pas dévoiler l'origine de la requête et ne pouvons donc pas utiliser l'identifiant du pair dans le réseau. Assurer un identifiant unique dans une architecture pair à pair devient alors délicat. Cependant nous pourrions imaginer une approche basée sur un hashage d'identifiant, un timestamp et/ou une valeur aléatoire, sans toutefois empêcher les risques de collision.

pas besoin de demander le score de Danny pour retourner une valeur. Ceci permet d'alléger le réseau en minimisant les valeurs transmises.

Notre exemple étant très simpliste, les calculs de score sont identiques dans les sections suivantes. L'exemple ci-dessus ne sera donc pas repris.

4.3. Correlative social scoring

Puisque les relations de confiance sont manuellement définies dans le réseau social par les acteurs, ce sont des valeurs subjectives. Nous avons introduit un coefficient de corrélation entre acteurs afin d'affiner la confiance. Contrairement aux approches traditionnelles, le *correlative social scorer* ne calcule cette corrélation qu'entre amis directs. Le coefficient de confiance est donc modulé par un coefficient de similarité, qui est un classique coefficient de corrélation de Pearson, (*c.f.* section 2 éq. 5), noté ρ. Pour calculer $\rho_{a,f}$ entre deux amis a et f, seuls les objets notés par les deux amis sont utilisés. Si ρ n'est pas calculable (les deux amis n'ont aucun objet en commun par exemple), le coefficient vaut 0.5. Le *correlative social scoring* prend en compte la confiance et la similarité pour le calcul de score, ainsi les amis aux goûts similaires sont promus dans le processus de recommandation.

$$\begin{aligned} score_k^\rho &: A^2 \times I \to [0,1] \cup \{\bot\} \\ &(a,l,i) \mapsto score_k^\rho(a,l,i) \end{aligned}$$

$$score_k^\rho(a,l,i) = \begin{cases} r_{a,i} & \text{si } \exists r_{a,i} \\ \dfrac{\sum_{f \in \mathcal{A}_{a,l,i}^{k-1}} t_{a,f} \times \rho_{a,f} \times score_{k-1}^\rho(f,a,i)}{\sum_{f \in \mathcal{A}_{a,l,i}^{k-1}} t_{a,f} \times \rho_{a,f}} & \text{si } \nexists r_{a,i} \wedge \mathcal{A}_{a,l,i}^{k-1} \neq \emptyset \\ \bot & \text{sinon} \end{cases} \tag{10}$$

4.4. Relative social scoring

Les *scorers* précédents calculent des scores absolus, *i.e.* indépendants de la tendance de notation des acteurs. Le *relative social scorer* calcule des scores relatifs, *i.e.* la différence pour chaque acteur entre un score et sa moyenne de notes. Les scores relatifs des amis sont ensuite agrégés et ajoutés à la moyenne des notes de l'acteur local.

Cette approche a deux principaux avantages. Premièrement, elle ne transmet aucun score absolu, mais uniquement des scores relatifs. Cela signifie moins d'information et donc plus de confidentialité. Deuxièmement, elle prend en compte les "tendances générales" de notation des acteurs, par exemple sévères (notes faibles) ou généreuses (notes élevées). Dans le cas où l'acteur n'a aucune note, nous utilisons 0.5 comme moyenne par défaut.

$$\delta score_k : A^2 \times I \to [-1,1] \cup \{\perp\}$$
$$(a,l,i) \mapsto \delta score_k(a,l,i)$$

$$\delta score_k(a,l,i) = \begin{cases} r_{a,i} - \overline{r_a} & \text{si } \exists r_{a,i} \\ \dfrac{\sum_{f \in \mathcal{A}_{a,l,i}^{k-1}} t_{a,f} \times \delta score_{k-1}(f,a,i)}{\sum_{f \in \mathcal{A}_{a,l,i}^{k-1}} t_{a,f}} & \text{si } \nexists r_{a,i} \wedge \mathcal{A}_{a,l,i}^{k-1} \neq \emptyset \\ \perp & \text{sinon} \end{cases} \tag{11}$$

Chaque acteur ajoute sa moyenne au score relatif reçu afin de calculer le score absolu final :

$$\Delta score_k(a,i) = \delta score_k(a,a,i) + \overline{r_a} \tag{12}$$

Cependant, le fait de passer d'un score absolu à un score relatif modifie les bornes des valeurs retournées par les pairs :

$$\begin{aligned} r_{a,i} \in [0,1] &\Rightarrow \overline{r_a} \in [0,1] \Rightarrow r_{a,i} - \overline{r_a} \in [-1,1] \\ &\Rightarrow \delta score_k \in [-1,1] \\ &\Rightarrow \Delta score_k \in [-1,2] \end{aligned}$$

Si $\Delta score_k(a,i)$ est négatif, 0 est retourné ; s'il est supérieur à 1, 1 est retourné.

Ce *scorer* partage moins d'information que les précédents. Les pairs ne partagent qu'une différence entre leurs notes et leurs moyennes. Ils indiquent qu'ils aiment un objet (ou pas) en renvoyant un score relatif positif (ou négatif).

4.5. CorrRelative social scoring

Le *CorrRelScorer* combine le *relative* et le *correlative social scoring* [4]:

4. Puisque ρ est calculé avec $r_{a,i} - \overline{r_a}$, il ne contredit pas l'approche du *relative social scoring*

$$\delta score_k^\rho : A^2 \times I \to [-1,1] \cup \{\perp\}$$
$$(a,l,i) \mapsto \delta score_k^\rho(a,l,i)$$

$$\delta score_k^\rho(a,l,i) = \begin{cases} r_{a,i} - \overline{r_a} & \text{si } \exists r_{a,i} \\ \dfrac{\sum_{f \in \mathcal{A}_{a,l,i}^{k-1}} t_{a,f} \times \rho_{a,f} \times \delta score_{k-1}^\rho(f,a,i)}{\sum_{f \in \mathcal{A}_{a,l,i}^{k-1}} t_{a,f} \times \rho_{a,f}} & \text{si } \nexists r_{a,i} \wedge \mathcal{A}_{a,l,i}^{k-1} \neq \emptyset \\ \perp & \text{sinon} \end{cases} \tag{13}$$

Puis :

$$\Delta score_k^\rho(a,i) = \delta score_k^\rho(a,a,i) + \overline{r_a} \tag{14}$$

De la même façon qu'en 4.4, si $\Delta score_k^\rho(a,i)$ est négatif, 0 est retourné ; s'il est supérieur à 1, 1 est retourné.

Dans la section suivante, nous évaluons ce *scorer* et le comparons à des algorithmes de recommandation classiques. Nos expériences ont souligné que le *relative social scorer* n'apporte pas ou peu de précision avec notre jeu de données, cependant il apporte une plus grande confidentialité dans le partage des notes. Le *correlative social scoring* est quant à lui le *scorer* offrant la plus grande précision. C'est pourquoi nous avons choisi d'évaluer la combinaison de ces deux algorithmes: le *CorrRelScorer*.

5. Évaluation

Pour évaluer notre approche, nous avons implémenté le *CorrRelative Social Scoring* (section 4.5 éq 13 et 14) ainsi que deux autres algorithmes couramment utilisés en filtrage collaboratif : *GlobalCF* et *GlobalMean*. *GlobalCF* utilise un coefficient de corrélation de Pearson pour calculer la similarité entre deux acteurs (section 2 éq.2). *GlobalMean* renvoie la moyenne de l'objet quel que soit l'utilisateur (section 2 éq.1). Ce *scorer* n'est donc pas personnalisé. Le jeu de données est présenté en section 5.1. Nous avons utilisé différentes mesures pour comparer les algorithmes, tel qu'expliqué en section 5.3. Nos campagnes d'évaluation sont détaillées en section 5.2.

5.1. Jeu de données Epinions

(Massa, Bhattacharjee, 2004) proposent un jeu de données dérivé du site web Epinions [5]. Ce jeu de données contient des objets notés par des acteurs et des liens de confiance, orientés et valués, entre acteurs. Il existe peu de jeux de données de cette nature disponibles, la plupart des réseaux sociaux ne contiennent pas de notes et les jeux de données classiques de recommandation n'incluent aucun réseau social. Ce jeu

5. www.epinions.com

de données est anonymisé : il ne contient aucune information sur les acteurs ou les produits, empêchant toute approche basée sur le contenu. Les notes sont des entiers de 1 (n'aime pas) à 5 (aime), avec une moyenne de 4 et une déviation standard de 1.2. Un acteur peut faire confiance à un autre acteur (trust = 1) ou pas (trust = 0). En moyenne, un acteur fait confiance à dix acteurs. La plupart des acteurs ne sont reliés qu'à peu d'autres acteurs, dans un sens ou dans l'autre.

La densité des notes est très faible avec seulement 0.01 % de notes. La distribution des notes est montrée figure 1. On y voit que la distribution n'est pas uniforme.

Tableau 1. Distribution des notes dans Epinions

Note	1	2	3	4	5	total
Effectif	43 228	50 678	75 525	194 340	301 053	664 824
Pourcentage	6.5 %	7.6 %	11.4 %	29.2 %	45.3 %	100 %
Cumulé	6.5 %	14.1 %	25.5 %	54.7 %	100 %	-

Nous avons ensuite retiré du jeu de données tous les acteurs n'ayant pas au moins une note et une relation de confiance, soient 11 858 acteurs et 75 109 notes. En effet *GlobalCF* nécessite au moins une note pour fonctionner et notre approche nécessite au moins une relation de confiance.

Notre jeu de test est donc constitué de 28 305 acteurs et 589 715 notes. Un tiers des acteurs sont des cold start users, c'est-à-dire des acteurs avec quatre notes ou moins (Massa, Avesani, 2007b).

5.2. Campagnes d'évaluation

La première méthode d'évaluation consiste à séparer le jeu de données en deux jeux : les données d'entraînement (ou de contexte) et les données d'évaluation. Les données d'entraînement sont utilisées par les algorithmes pour prédire les notes contenues dans le jeu d'évaluation. La différence entre le score prédit et la note réelle, contenue dans le jeu d'évaluation, est un indicateur de la qualité de prédiction de l'algorithme. Nous avons aléatoirement séparé le jeu de données en parts de tailles différentes : 20 %-80 %, 50 %-50 %, 80 %-20 % et 90 %-10 %. Le premier groupe est le jeu d'entraînement, le second le jeu d'évaluation.

Nous avons aussi défini un jeu d'entraînement contenant "100 %" des notes. Cette deuxième méthode d'évaluation, couramment appelée *leave one out*, consiste à donner à l'algorithme toutes les notes sauf une afin de lui faire prédire la note manquante avec un maximum de contexte, ainsi de suite pour chaque note. Il n'y a aucune sélection aléatoire, cette méthode d'évaluation est entièrement reproductible.

Notre évaluation comporte donc cinq jeux d'entraînement contenant respectivement 20, 50, 80, 90 et 100 % des notes.

5.3. Mesures

Dans la section 2, nous avons sélectionné et décrit des mesures pertinentes pour la recommandation: des mesures de précision de la prédiction (MAE et RMSE) et du rang (PCC et SRCC) et une mesure de couverture. En plus de ces mesures, nous ajoutons deux mesures de précision de la prédiction qui pondèrent l'erreur : *Weighted Absolute Error* (WAE) et *Root Weighted Square Error* (RWSE), respectivement similaires aux MAE et RMSE.

$$WAE = \frac{\sum_{n=1}^{N} \left(|p_n - r_n| \times |r_n - \bar{r}| \right)}{N} \tag{15}$$

$$RWSE = \sqrt{\frac{\sum_{n=1}^{N} \left((p_n - r_n) \times (r_n - \bar{r}) \right)^2}{N}} \tag{16}$$

Ces mesures permettent de pondérer l'erreur commise par le système par un coefficient prenant en compte l'originalité de la note. Plus la note est originale, c'est-à-dire plus la note est éloignée de la moyenne des notes de l'objet, plus l'erreur sera pénalisée. Cela permet d'accentuer les erreurs commises sur les notes les plus difficiles à prédire, *i.e.* celles éloignées de la moyenne. RWSE pénalise notamment beaucoup plus les grandes erreurs commises sur des notes éloignées de la moyenne. Nous utiliserons cette mesure d'erreur pour calculer la précision définie en éq. 6 et ainsi calculer la f-mesure (*c.f.* éq. 7).

6. Résultats

6.1. Connaissance du système

Nous résumons dans le tableau 2 la connaissance des *scorers* évalués sur l'état du système. Les *GlobalCF* et *GlobalMean* ont une connaissance globale du système : toutes les notes de l'objet sont utilisées pour calculer la moyenne d'un objet et toutes les notes sont utilisées pour calculer les similarités entre acteurs. Le *CorrRelative Social Scorer* n'a qu'une connaissance partielle du système : il ne connaît que les scores relatifs des voisins immédiats dans le réseau social.

Tableau 2. Algorithmes évalués

Scorer	Définition	Connaissance
GlobalMean	éq. 1	toutes les notes de l'objet
GlobalCF	éq. 2	toutes les notes
CorrRelScorer$_1$	éq. 14, avec $k = 1$	local trust et Δratings jusqu'à une profondeur 1
CorrRelScorer$_2$	éq. 14, avec $k = 2$	local trust et Δratings jusqu'à une profondeur 2
CorrRelScorer$_3$	éq. 14, avec $k = 3$	local trust et Δratings jusqu'à une profondeur 3

Moins le *scorer* a de connaissances sur le système, mieux il peut prendre en compte la confidentialité des données. *GlobalCF* a besoin d'une connaissance globale sur le système pour calculer les similarités entre utilisateurs. *GlobalMean* n'utilise qu'une vision partielle, dans le sens où seules les notes d'un objet en particulier sont utilisées pour en calculer la moyenne, cependant aucune distinction n'est faite entre les utilisateurs, donc *GlobalMean* a potentiellement une connaissance globale sur le système.

Notre *scorer* est donc le seul à offrir à l'utilisateur la possibilité de sélectionner les amis avec qui il veut partager ses informations. Ceci permet d'établir une politique de confidentialité orientée utilisateur.

6.2. Couverture

La figure 2 montre la couverture des cinq *scorers* en fonction de la taille du jeu d'entraînement, de 20 % jusqu'à 100 % du jeu de données total.

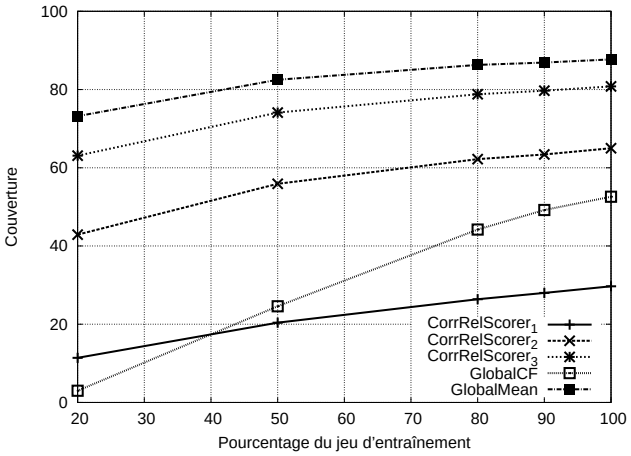
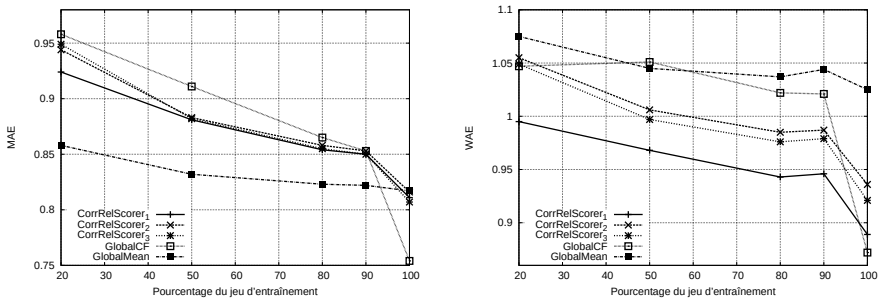


Figure 2. Couverture des prédictions en fonction du pourcentage du jeu d'entraînement

Comme attendu, *GlobalMean* a une très bonne couverture, la meilleure dans notre évaluation, (*c.f.* section 2). *GlobalCF* a beaucoup de difficultés à offrir une recommandation complète, puisqu'il lui faut un jeu d'entraînement complet (100 %) pour atteindre les 50 % de recommandation. La couverture du *CorrRelScorer* est directement proportionnelle à la profondeur de propagation k dans le réseau social : plus k est grand, meilleure est la couverture. Avec seulement une propagation jusqu'à une profondeur de $k = 3$, la couverture atteint presque celle de *GlobalMean*, à 7 % de différence en moyenne. Propager à une profondeur de $k = 2$ suffit pour prédire bien plus de notes que *GlobalCF*, dont plus de la moitié des notes avec un jeu d'entraînement de seulement 50 % des notes. La propagation des scores est donc rapidement efficace pour augmenter la couverture de la recommandation.

6.3. Précision de la prédiction

Les figures 3a et 3b montrent respectivement la MAE et la WAE des cinq *scorers* en fonction de la taille du jeu d'entraînement, de 20 % jusqu'à 100 % du jeu de données total.



(a) MAE (b) WAE

Figure 3. Erreur des prédictions en fonction du pourcentage du jeu d'entraînement

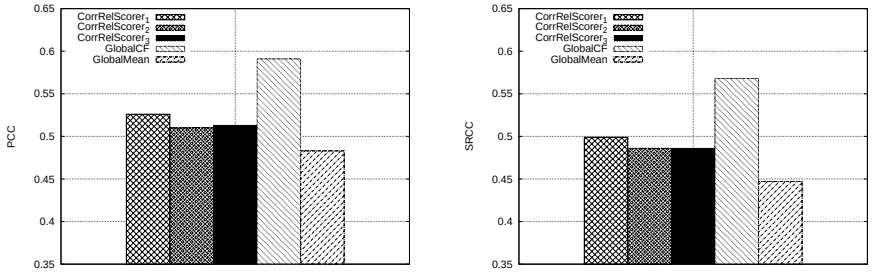
L'erreur devant être minimisée, on voit dans ces deux graphes que notre algorithme social, le CorrRelScorer, est presque toujours plus précis que celui de l'approche pure de filtrage collaboratif *GlobalCF*. Ce dernier a besoin d'un jeu d'entraînement maximal pour être efficace. Ainsi nous voyons qu'avec le jeu d'entraînement de 100 % il devient plus précis que tous les autres algorithmes (quelle que soit la mesure).

La figure 3a montre clairement que l'approche basique qui consiste à renvoyer la moyenne est peu influencée par la taille du jeu d'entraînement. Ayant accès à toutes les notes, elle n'est pas gênée par la faible densité du jeu de données. Par contre augmenter la densité n'augmente pas vraiment sa précision, contrairement aux autres approches. Cependant, *GlobalMean* est pertinente dans notre évaluation, avec une MAE faible en général. Ceci peut s'expliquer par le fait que les notes sont en général très regroupées (une variance par objet faible) et donc renvoyer une moyenne globale de l'objet devient pertinent. Il suffit de prendre en compte ce facteur, grâce à la WAE, pour se rendre compte que *GlobalMean* perd en précision pour les notes originales, éloignées de la majorité des notes. En effet, la figure 3b montre clairement que *GlobalMean* devient alors le moins précis (en moyenne) des *scorers*.

6.4. Précision du classement

Les figures 4a et 4b montrent respectivement le PCC et le SRCC des cinq *scorers* pour un jeu d'entraînement de 100 % des notes.

La corrélation devant être maximisée, on voit dans ces deux graphes que l'approche pure de filtrage collaboratif reprend le dessus en termes de précision. Si le



(a) PCC (b) SRCC

Figure 4. Corrélation des prédictions pour un jeu d'entraînement de 100 %

GlobalCF offre une couverture relativement faible, il offre toutefois une prédiction pertinente, puisque le rang en sortie des prédictions est fortement corrélé avec le rang des notes à prédire. Les deux figures soulignent également le manque de personnalisation des prédictions de *GlobalMean*, avec la corrélation la plus faible. Ceci s'explique notamment par le fait que *GlobalMean* renvoyant une moyenne, les notes vont très vite se retrouver confinées dans un intervalle de valeurs faibles (autour de la moyenne globale du jeu de données) et les rangs vont vite s'inverser entre deux objets ayant une moyenne proche.

6.5. F-Mesure

Afin de fournir une évaluation des algorithmes prenant en compte à la fois la précision et la couverture, la figure 5 montre la f-mesure définie section 2 (éq 7) et adaptée section 5.3 pour un jeu d'entraînement de 100 %.

La f-mesure tient compte de la couverture et de la précision. *GlobalMean* étant l'algorithme ayant la plus grande couverture, il fournit un des meilleurs résultats. Cependant, il est pénalisé par son manque de personnalisation, souligné par la figure 3b. Notre *CorrRelScorer*, avec une propagation de $k = 3$ dans le réseau social combiné à un calcul de similarité entre amis, propose la f-mesure la plus haute. Ceci indique qu'il offre un compromis entre la couverture et la précision. Bien que couvrant moins de prédictions que *GlobalMean*, il offre une précision plus élevée. *GlobalCF*, qui fournit la meilleure prédiction pour le jeu d'entraînement de 100 %, est quant à lui pénalisé par sa faible couverture.

7. Conclusion

Dans ce papier, nous avons présenté différentes versions d'un algorithme de systèmes de recommandation, appelées *scorers*, pour un système de recommandation

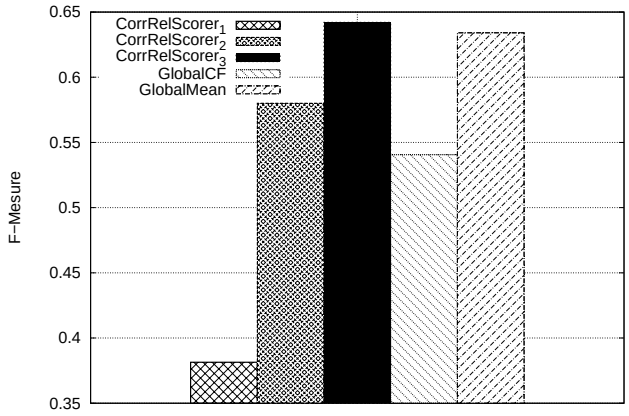


Figure 5. F-Mesure pour un jeu d'entraînement de 100 %

basé sur la confiance. Notre algorithme prend en compte la confidentialité des données et est implémentable sur des architectures décentralisées, notamment en pair à pair. La plupart des algorithmes de recommandation existant dans la littérature utilisent une connaissance globale sur le système, c'est-à-dire les profils complets de tous les utilisateurs. Ces algorithmes ne sont donc pas applicables sur des architectures décentralisées. Les systèmes de recommandation basés sur la confiance tirent parti d'un réseau social déjà construit. Cependant les approches existantes ne tiennent pas compte d'une quelconque confidentialité des données car la confiance est propagée dans le réseau et les profils sont partagés entre utilisateurs qui ne se font pas directement confiance.

Nous avons défini des algorithmes qui peuvent être distribués sur des pairs et prenant en compte la confidentialité des données des utilisateurs. Ils utilisent une vision limitée du réseau social : le nœud où est implémenté l'algorithme ainsi que ses voisins directs. Ils ne propagent pas la confiance et ne s'appuient que sur les relations de confiance définies par les utilisateurs pour propager les scores. De plus, nos *relative scorers* ne propagent pas des scores complets, mais seulement une différence entre un score et la moyenne de l'utilisateur.

Nous présentons une évaluation comparative de notre approche avec deux algorithmes classiquement utilisés en filtrage collaboratif : *GlobalMean* et *GlobalCF*. Notre évaluation contient plusieurs mesures : précision de prédiction, précision du classement et couverture. Ces trois catégories de mesures montrent des aspects différents auxquels un système de recommandation doit répondre. Nous utilisons la f-mesure afin de combiner la couverture avec la précision de la prédiction. Nous avons également introduit deux nouvelles mesures de précision, la *Weighted Average Error* et la *Root Weighted Square Error*, fondées sur la MAE et la RMSE, qui pondèrent l'er-

reur en fonction de l'originalité de la note, autrement dit en fonction de l'éloignement de la note par rapport à la moyenne de l'objet. Ces mesures sont notamment utiles pour faire ressortir les algorithmes qui ont du mal à prédire des notes personnalisées.

Notre évaluation montre qu'utiliser les relations directes dans le réseau social ($k = 1$) offre une très bonne prédiction mais entraîne une couverture trop faible. Pour améliorer cette couverture, nous proposons de propager les scores dans le réseau social en utilisant uniquement les pairs de confiance et en limitant la profondeur de propagation pour ne pas inonder le réseau. Les résultats montrent que propager à une profondeur de $k = 2$ augmente significativement la couverture : entre deux et trois fois plus de notes sont prédites. Ils montrent également que cette propagation entraîne une légère perte de précision. Le gain en couverture est notamment visible grâce à la f-mesure, qui montre l'intérêt de la propagation dans le réseau social.

Dans la campagne d'évaluation à partir d'un jeu d'entraînement complet (100 % des notes), l'algorithme qui obtient la f-mesure la plus élevée est notre *CorrRelScorer* avec une propagation de $k = 3$. Bien qu'il ait une plus faible couverture que *GlobalMean*, et une plus faible précision que *GlobalCF*, c'est celui offrant le meilleur compromis entre couverture (81 % des notes sont prédites) et précision ($MAE = 0.807$).

Dans la littérature, on trouve des approches qui parcourent l'intégralité du réseau social, comme par exemple PageRank (Page *et al.*, 1999). Leur couverture est certes maximisée, mais le coût de leur utilisation notamment dans des architectures pair à pair est prohibitif. Elles utilisent également des techniques qui vont à l'encontre de notre objectif de confidentialité des données.

Nous avons implémenté une version décentralisée de nos algorithmes sur le simulateur P2P PeerSim (Montresor, Jelasity, 2009). Nos travaux futurs consisteront à évaluer et améliorer la prise en compte de l'architecture décentralisée dans la recommandation. Nous allons observer la charge du réseau induite par notre approche et simuler les contraintes des architectures P2P (déconnections, dynamicité, timeout...). Afin d'éviter toute surcharge du réseau, nous expérimenterons des heuristiques pour sélectionner parmi les amis disponibles ceux qui seront le plus à même de fournir une recommandation pertinente.

Nous avons vu que nos *scorers* offrent une bonne recommandation mais pourraient profiter des approches basées sur le contenu pour améliorer la couverture. Ces approches tirent parti des caractéristiques intrinsèques des objets à recommander. Elles ne nécessitent aucune connaissance sur les autres utilisateurs et sont donc compatibles avec notre objectif de confidentialité des données.

Bibliographie

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 and data engineering*, vol. 17, n° 6, p. 734–749.

Balabanović M., Shoham Y. (1997). Fab: content-based, collaborative recommendation. *Communications of the ACM*, vol. 40, n° 3, p. 66–72.

Basu C., Hirsh H., Cohen W. (1998). Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the national conference on artificial intelligence*, p. 714–720. John Wiley & sons ltd.

Breese J., Heckerman D., Kadie C., Others. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th conference on uncertainty in artificial intelligence*, p. 43–52. Madison: Morgan Kaufmann.

DuBois T., Golbeck J., Kleint J., Srinivasan A. (2009). Improving recommendation accuracy by clustering social networks with trust. In *ACM recsys'09 workshop on recommender systems & the social web.* Citeseer.

Eppler M., Mengis J. (2004, novembre). The Concept of Information Overload: A Review of Literature from Organization Science, Accounting, Marketing, MIS, and Related Disciplines. *The Information Society*, vol. 20, n° 5, p. 325–344.

Golbeck J. (2005). *Computing and applying trust in web-based social networks*. Thèse de doctorat non publiée, University of Maryland at College Park.

Hang C., Wang Y., Singh M. (2009). Operators for propagating trust and their evaluation in social networks. In *Proceedings of the 8th international conference on autonomous agents and multiagent systems-volume 2*, p. 1025–1032. International Foundation for Autonomous Agents and Multiagent Systems.

Hasan O. (2010). *Privacy Preserving Reputation Systems for Decentralized Environments*. Thèse de doctorat non publiée, Institut National des Sciences Appliquées de Lyon.

Herlocker J., Konstan J., 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*, p. 230–237. ACM Press.

Herlocker J., Konstan J., Terveen L., Riedl J. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, vol. 22, n° 1, p. 5–53.

Jamali M., Ester M. (2009). 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*, p. 397–406. ACM Press.

Lee D., Brusilovsky P. (2009). Does Trust Influence Information Similarity? In *Proceedings of workshop on recommender systems & the social web, the 3rd ACM international conference on recommender systems*, p. 3–6. Citeseer.

Ma H., King I., Lyu M. (2009). Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on research and development in information retrieval*, p. 203–210. New York, USA, ACM Press.

Massa P., Avesani P. (2007a). Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on recommender systems*, p. 17–24. New York, USA, ACM Press.

Massa P., Avesani P. (2007b). Trust metrics on controversial users: balancing between tyranny of the majority and echo chambers. *International Journal on Semantic Web and Information Systems*, vol. 3, n° 1, p. 39–64.

Massa P., Bhattacharjee B. (2004). Using trust in recommender systems: an experimental analysis. *Trust Management*, vol. 2995/2004, p. 221–235.

McKnight D., Chervany N. (1996). *The meanings of trust*. Rapport technique n° 612, vol. 55455. University of Minnesota, Carlson School of Management.

Mehta B., Hofmann T., Nejdl W. (2007). Robust collaborative filtering. In *Proceedings of the 2007 ACM conference on recommender systems - recsys '07*, p. 49. New York, USA, ACM Press.

Milgram S. (1967). The small world problem. *Psychology today*, vol. 2, n° 1, p. 60–67.

Montresor A., Jelasity M. (2009, septembre). PeerSim: A scalable P2P simulator. In *Proc. of the 9th int. conference on peer-to-peer (p2p'09)*, p. 99–100. IEEE.

O'Donovan J., Smyth B. (2005). Trust in recommender systems. In *Proceedings of the 10th international conference on intelligent user interfaces*, p. 167–174. New York, USA, ACM Press.

Page L., Brin S., Motwani R., Winograd T. (1999). *The PageRank Citation Ranking: Bringing Order to the Web*. Rapport technique. Stanford Digital Library Technologies Project.

Pitsilis G., Knapskog S. (2009). Social Trust as a solution to address sparsity-inherent problems of Recommender systems. *Recommender Systems & the Social Web*, vol. 826, n° October, p. 33–40.

Richardson M., Domingos P. (2002). Mining knowledge-sharing sites for viral marketing. In *Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining*, p. 61–70. New York, USA, ACM Press.

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

Wasserman S., Faust K. (1994). *Social network analysis: Methods and applications*. Cambridge Univ Pr.