Recommandation et analyse de sentiments dans un espace latent textuel

Charles-Emmanuel Dias* — **Vincent Guigue*** — **Patrick Gallinari***

**Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6, F-75005, Paris, France – Mail: Prénom.Nom@lip6.fr*

RÉSUMÉ. *Les systèmes de recommandation permettent d'aider les utilisateurs à identifier les contenus qu'ils seraient susceptibles d'apprécier dans des catalogues en expansion constante. Les méthodes traditionnelles de filtrage collaboratif se focalisent majoritairement sur les notes que les gens laissent en ligne pour établir leurs profils et ignorent les commentaires textuels éventuellement joints. Nous prenons le parti d'exploiter les avis textuels comme source principale d'information et proposons, en plus de prédire les notes, de prédire les remarques des utilisateurs. Nous montrons aussi comment cette approche peut offrir de nouvelles perspectives pour le problème du démarrage à froid. À l'aide d'une technique récente de représentation de textes, nous construisons un espace latent d'avis où chacun de leurs aspects est représenté. Tous les acteurs du système sont représentés sous forme de vecteurs et une métrique ad hoc nous permet d'attaquer différentes problématiques autour de la recommandation.*

ABSTRACT. *Recommender systems were developed to cherry-pick interesting content in an always growing environment to help users discover new products or information that they might like. Traditional collaborative filtering methods mainly focus on the ratings to establish user profiles, ignoring the joint review text. Our work does the opposite. We use the written comments as the primary source of information and propose, in addition to rating forecast, to predict review text. Our approch also enables us to elegantly tackle the cold-start problem. Using state-of-the-art text embedding technique, we build a latent review space where all the components of a review is mapped. The created vectors are used to predict both rating and review text.*

MOTS-CLÉS : *Recommandation, Analyse de sentiments, Apprentissage de représentations textuelles*

KEYWORDS: *Recommender systems, Text embedding, Sentiment classification.*

1. Introduction

Afin d'aider les utilisateurs à accéder à des ressources en ligne toujours plus foisonnantes, des systèmes de recommandation ont été développés pour sélectionner et trier les contenus susceptibles de leur plaire. Ils permettent, par exemple, de composer des newsletters ou de fournir des conseils individualisés sur les produits à acheter sur Amazon, les restaurants où diner grâce à TripAdvisor et même les films à regarder sur Netflix. L'enjeu est de proposer un véritable accès personnalisé à l'information.

Pour atteindre cet objectif, il faut pouvoir fournir des recommandations justes. Mais, il est tout aussi important d'expliquer en quoi celles-ci sont appropriées (Tintarev et Masthoff, 2007). Les algorithmes de recommandation ont franchi un premier cap d'efficacité et de popularité avec le challenge Netflix (Bennett et Lanning, 2007) qui a consacré le filtrage collaboratif (Koren *et al.*, 2009). Ces dernières années, ils sont devenus de plus en plus performants grâce à la prise en compte d'un nombre croissant de facteurs comme le temps (Koren, 2009 ; McAuley et Leskovec, 2013a), les liens sociaux (Guy et Carmel, 2011) ou le texte (McAuley et Leskovec, 2013b). Pourtant, ces améliorations n'ont pas contribué à l'explicitation des propositions faites aux utilisateurs et les méthodes récentes de filtrage collaboratif utilisant la factorisation matricielle sont encore souvent qualifiées de boites noires (Ben-elazar et Koenigstein, 2014). Il reste donc des progrès à faire pour permettre à l'utilisateur d'interpréter les conseils de ces algorithmes.

Dans ce papier, nous avons choisi de travailler sur les revues en ligne. Celles-ci, laissées par chacun sur internet, renferment de précieuses informations sur les avantages et les inconvénients des éléments notés. Leur exploitation peut permettre de connaître les préférences d'un utilisateur de façon plus fine que les notes, qui peuvent être trompeuses (Gauthier *et al.*, 2015). Différents travaux explorent l'utilisation du texte en recommandation, majoritairement avec comme but d'améliorer la qualité prédiction plutôt que celle des explications (McAuley et Leskovec, 2013b ; Ling *et al.*, 2014). (Poussevin *et al.*, 2014) a proposé une nouvelle tâche de prédiction des remarques utilisateurs qui permet justement d'apporter éclaircissement quant aux suggestions faites.

Depuis le lancement de la compétition Netflix en 2006, on observe un consensus sur les données qui servent à la recommandation: des triplets (utilisateur, produit, note). Le texte n'est utilisé qu'à la marge pour affiner l'espace de représentation et donc la recommandation. Nous proposons dans cet article d'évaluer les similarités entre acteurs (utilisateur, produit...) dans un espace latent de mots et de nous baser uniquement sur le texte disponible pour la création des profils latents. Grâce à une technique récente de représentation de texte, nous construisons un espace latent d'avis en ligne entièrement basé sur le texte (Mikolov *et al.*, 2013 ; Le et Mikolov, 2014). Les similarités découlant de cet espace sont ensuite utilisées pour prédire à la fois les notes et les remarques des utilisateurs avec précision. Nous démontrons dans cet article la pertinence de l'espace sémantique comme support de projection pour les utilisateurs et les produits dans le cadre de la tâche de prédiction de note. De plus, notre démarche

apporte une solution naturelle et élégante au problème du démarrage à froid (Contardo *et al.*, 2014). Alors que les systèmes de factorisation matricielle ne peuvent pas modéliser un utilisateur où un produit sans historique, une simple description textuelle nous permet de projeter un nouvel élément dans l'espace latent. Au bout du processus, tous les utilisateurs, produits ou éléments du système de recommandation sont positionnés dans un espace sémantique de mots, de phrases et de revues. Il est donc possible d'expliquer chaque recommandation faite à travers des mots-clés, des phrases et même des revues à la manière de (Poussevin *et al.*, 2014).

Notre contribution est d'utiliser l'information textuelle comme support principal d'un système de recommandation afin de créer des profils d'utilisateurs et de produits plus riches. D'améliorer et, surtout, d'expliquer les recommandations faites.

Ce papier est organisé de la façon suivante: tout d'abord, nous exposons le contexte bibliographique de nos travaux (section 2). Ensuite, nous expliquons comment construire notre espace latent de recommandation et comment nous en servir (section 3). Enfin, nous exposons les résultats obtenus (section 4).

2. État de l'art

Dans cette section, nous décrivons d'abord la tâche de prédiction des notes à travers le filtrage collaboratif et ses principaux modèles. Ensuite, nous présentons les approches de la littérature pour exploiter les représentations textuelles.

2.1. *Filtrage collaboratif*

En recommandation, le filtrage collaboratif est une méthode de prédiction personnalisée qui se base sur l'historique des notes données par les utilisateurs. L'hypothèse sous-jacente est que si plusieurs personnes ont des intérêts communs pour certaines choses alors ces intérêts s'étendent probablement à d'autres produits. En filtrage collaboratif, la factorisation matricielle draine l'attention de la plus grande partie de la communauté scientifique. Nous discutons aussi des techniques à base de voisinage sur lesquelles repose notre exploitation des espaces latents de texte pour la recommandation.

2.1.1. *Factorisation matricielle*

La principale technique utilisée en recommandation est la factorisation matricielle. Cette méthode a fait ses preuves lors du concours Netflix (Bennett et Lanning, 2007) grâce aux travaux de référence de (Koren *et al.*, 2009). Ces modèles extraient des profils latents d'utilisateurs et de produits directement à partir de la matrice de note de façon à ce que la note prédite $\hat{r_{ui}}$ résulte du produit scalaire entre les profils du produit i et de l'utilisateur u.

$$\hat{r_{ui}} \approx q_i^T p_u \quad [1]$$

À la manière des méthodes de segmentation thématiques on estime que chaque dimension latente des profils correspond à un aspect concret. La présence du même aspect dans deux profils provoque une union locale faisant monter la note.

En recommandation, les notes sont fortement biaisées. Certaines personnes ont tendance à toujours mettre de bonnes/mauvaises notes et certains produits ont tendance à être sur/sous notés. Classiquement, ces biais sont modélisés en ajoutant une moyenne globale μ, un biais produit b_i et un biais utilisateur b_u à [1]. De ce fait, une note $\hat{r_{ui}}$ est prédite de cette façon:

$$\hat{r_{ui}} = \mu + b_i + b_u + q_i^T p_u \quad [2]$$

$$\min_{q,p,b} \sum_{(u,i)} (r_{ui} - \mu + b_i + b_u + q_i^T p_u)^2 + \lambda(||p_u||^2 + ||q_u||^2 + b_u^2 + b_i^2) \quad [3]$$

Grâce à cette méthode, les profils latents capturent uniquement la déviation à la moyenne, ce qui permet une estimation moyenne lorsqu'aucune information n'est disponible. De nombreuses variantes ont été proposées pour prendre en compte le temps (Koren, 2009 ; McAuley et Leskovec, 2013a), les liens sociaux (Guy et Carmel, 2011) ou le texte (cf section 2.2).

2.1.2. *Modèle par voisinage*

L'approche par plus proches voisins est souvent considérée comme obsolète depuis l'avènement des techniques de factorisation matricielle (Koren, 2008). Bien qu'incontestablement lourde à mettre en œuvre du point de vue computationnel, elle est toujours d'actualité, du fait de sa précision, sa simplicité et sa popularité. Prédire la note $\hat{r_{ui}}$ d'un utilisateur u sur un produit i en utilisant les k plus proches produits, ou utilisateurs, voisins $\mathcal{N}$ se fait à l'aide d'une somme pondérée des notes des produits, ou utilisateurs, similaires:

$$\hat{r_{ui}} = \frac{\sum_{v \in \mathcal{N}} \alpha_{uv} r_{vi}}{\sum_{v \in \mathcal{N}} \alpha_{uv}}, \; \hat{r_{ui}} = \frac{\sum_{j \in \mathcal{N}} \alpha_{ij} r_{uj}}{\sum_{j \in \mathcal{N}} \alpha_{ij}} \quad [4]$$

Où α_{uv} et α_{ij} représentent respectivement la similarité entre utilisateurs u et v et entre produits i et j. Un certain nombre de mesures de similarités existent comme la corrélation de Pearson qui mesure la tendance qu'ont les utilisateurs de noter de manière identique. Les notes peuvent être normalisées par la moyenne afin de prendre en compte les biais de notation des utilisateurs. (Desrosiers et Karypis, 2011) présente ces méthodes de manière détaillée.

Pour nous, cette stratégie présente l'intérêt de décorréler la mesure de similarité entre i et j (α_{ij}) et la tâche visée, par exemple la prédiction de note. Nous allons exploiter une telle stratégie pour démontrer la richesse des profils textuels dans la tâche de filtrage collaboratif, à la manière de ce qui est fait lorsque les recommandations sont basées sur le contenu.

2.2. *Filtrage collaboratif avec données textuelles*

Plusieurs méthodes de filtrage collaboratif utilisent déjà le texte afin d'améliorer leurs précisions. (McAuley et Leskovec, 2013b ; Ling *et al.*, 2014) alignent les représentations latentes de sujets extraits par *Latent Dirichlet Allocation* (Blei *et al.*, 2003) avec les profils des utilisateurs et des produits issus de l'analyse des triplets (utilisateur, produit, note). (Poussevin *et al.*, 2014) construit des profils textuels basés sur des sacs de mots et les incorpore dans la règle de prédiction en ajoutant un terme dans [2].

Bien que notre travail présente certaines similarités avec ceux précédemment cités, il ne s'agit pas d'une extension des travaux de (Koren *et al.*, 2009). Au contraire, nous cherchons à mettre en œuvre un algorithme de recommandation par le texte avec comme enjeu majeur l'explication en plus de la performance. Nos travaux sont philosophiquement plus proches de ceux de (Ganu *et al.*, 2009) qui cherche à démontrer l'importance du texte pour la recommandation.

3. Espace latent d'avis en ligne

Notre but est de construire un espace latent où chaque utilisateur, produit, note et mot est représenté par un vecteur afin d'encoder l'ensemble des informations disponibles dans un avis (figure 1). De plus, il doit exister une mesure de similarité entre chacune de ces représentations, que des utilisateurs qui aiment les mêmes choses soit plus proches que ceux qui aiment des choses différentes.

Pour obtenir ces propriétés, nous nous appuyons sur l'approche *paragraph2vec* (Le et Mikolov, 2014) qui est une extension de *word2vec* (Mikolov *et al.*, 2013) aux documents. Nous modifions celui-ci en un algorithme que nous appelons *Conceptual Skip-Gram* (CSG). Le texte devient le dénominateur commun des acteurs du système (utilisateur, produit, note). La modification du *Skip-Gram* doit permettre la projection puis la comparaison des différents concepts, même lorsqu'ils ne sont pas homogènes.

3.1. *Conceptual Skip-Gram*

Notre modèle dérive donc de *paragraph2vec* de (Le et Mikolov, 2014). Plus exactement, nous utilisons un modèle appelé *Distributed Bag-of-Word* (DBOW), extension du modèle *Skip-Gram* (SG). En effet, en considérant les utilisateurs, les produits et les notes comme des éléments conceptuels caractérisés par un ensemble de mots les représentant il est possible de les introduire dans l'algorithme précédemment cité comme des paragraphes: le but est de trouver la meilleure représentation pour chaque concept, celle qui permette au mieux de prédire ses mots (figure 2).

Nous utilisons la technique dite du *negative sampling* de (Mikolov *et al.*, 2013) décrite en détail par (Goldberg et Levy, 2014). Cette méthode est une simplification de la *Noise-Contrasive Estimation* (Gutmann et Hyvärinen, 2010) qui postule qu'un

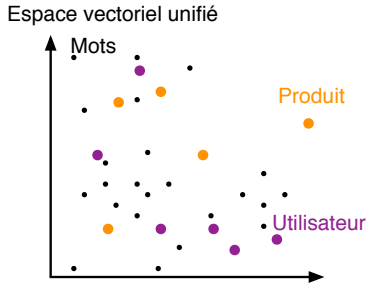

Figure 1. *Représentation de l'espace latent d'avis en ligne. Une fois les concepts hétérogènes projetés dans le même espace, différentes tâches deviennent traitables.*

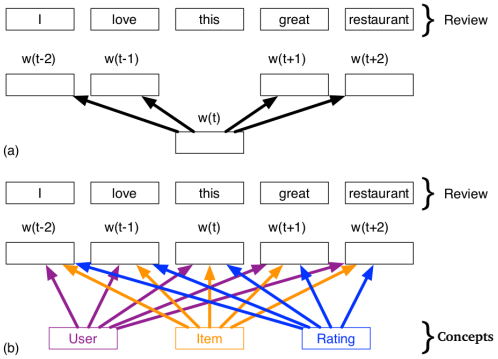

Figure 2. *(a) Skip-Gram classique sur du texte. (b) Conceptual Skip-Gram qui projette les concepts (utilisateur, produit, note) dans l'espace des mots.*

bon modèle doit être capable de différencier les données réelles du bruit grâce à la régression logistique.

Les représentations latentes de chaque mot et de chaque "concept" des avis (utilisateur, produit, note) sont apprises ensemble avec deux objectifs séparés, mais similaires: les mots cooccurrents localement dans les avis (à l'intérieur d'une fenêtre) doivent être proches les uns des autres (et les autres doivent être plus loin) ; chaque concept doit être proche de ses mots (et éloigné de ceux des autres). Dans la pratique, les positions des mots sont apprises en premier, les concepts sont ensuite positionnés par rapport aux mots. Autrement dit, avec $\sigma(x) = \frac{1}{1+e^{-x}}$, les fonctions objectifs sont les suivantes:

$$\underset{v}{\operatorname{argmax}} \left(\log \sigma(v_a'^{\intercal}.v_w) + \sum_{i=1}^{k} \mathbb{E}_{v_b \sim P_n(\mathbf{w})} \log \sigma(-v_b'^{\intercal}.v_w) \right) \quad [5]$$

$$\underset{v}{\operatorname{argmax}} \left(\log \sigma(v_a'^{\intercal}.v_c) + \sum_{i=1}^{k} \mathbb{E}_{v_b \sim P_n(\mathbf{w})} \log \sigma(-v_b'^{\intercal}.v_w) \right) \quad [6]$$

Où v_w et v_c sont les mots (resp. concepts) et leurs mots associés v_a (cooccurrences ou contextes). Les v_b sont le bruit, tirées selon une loi unigramme, élevée à la puissance 3/4[1]. Contrairement à (Le et Mikolov, 2014) qui utilise un softmax hiérarchique pour l'apprentissage, nous utilisons seulement le negative sampling comme décrit ci-dessus.

1. La puissance 3/4 permet de lisser la distribution unigramme et de faire ressortir les mots rares (Levy *et al.*, 2015)

Après entraînement, on obtient un espace vectoriel en n dimension où les concepts et les mots similaires sont proches. La nombre de représentations obtenues est de $|mots| + |concepts|$.

3.2. *Exploitation de l'espace latent d'avis*

À l'instar de word2vec, le cosinus s'utilise comme mesure de similarité afin de trouver les éléments proches. Par convenance, nous basculons cette mesure sur l'intervalle $[0, 1]$ afin de l'utiliser comme pondération dans [4].

$$\alpha_{uv} = \frac{\frac{<u,v>}{\|u\| \times \|v\|} + 1}{2} \in [0, 1] \qquad [7]$$

En considérant chaque utilisateur, produit et note comme concepts grâce aux mots qui les représentent, on obtient un espace où chaque composante des avis en ligne est représentée (figure 1). Il est également possible de projeter individuellement chaque avis et chaque phrase, mais le nombre de représentations deviendrait trop important. Les phrases sont donc projetées dans l'espace latent en moyennant les représentations de leurs mots, comme suggéré par (Kageback *et al.*, 2014).

Prédiction de notes

La mesure de similarité [7] rend possible le tri et la sélection des produits, des utilisateurs ou même des phrases et mots voisins à un autre produit, utilisateur ou même mot. C'est à l'aide de cette mesure que nous effectuons nos recommandations. Plus précisément, pour prédire la note $\hat{r_{ui}}$ d'un utilisateur u sur un nouveau produit i on utilise les notes déjà données sur ce même produit i par les autres utilisateurs $v \in \mathcal{N}, r_{uv}$, pondérées par leur similarité à l'utilisateur u, α_{uv}. En pratique, on calcule seulement cette moyenne sur le k plus proche voisinage $\mathcal{N}$. Par ailleurs, les notes données par chaque utilisateur sont normalisées par la moyenne de chaque utilisateur μ_u pour prendre en compte leur biais de notation. Symétriquement, la même opération est possible avec les notes déjà données par l'utilisateur u sur des produits similaires au produit i. De ce fait, on obtient deux modèles de prédiction [8] qui découlent de [4] avec les α issus de [7]:

$$\hat{r_{ui}} = \mu_u + \frac{\sum_{v \in \mathcal{N}} \alpha_{uv}(r_{vi} - \mu_v)}{\sum_{v \in \mathcal{N}} \alpha_{uv}},\ \hat{r_{ui}} = \mu_i + \frac{\sum_{j \in \mathcal{N}} \alpha_{ij}(r_{uj} - \mu_j)}{\sum_{j \in \mathcal{N}} \alpha_{ij}} \qquad [8]$$

La prédiction est donc obtenue en moyennant soit:

– les avis du même utilisateur sur des produits proches,

– les avis des autres utilisateurs sur le produit cible.

Dataset	#Users	#Items	#Reviews	#Appren.	#Valid. + #Test
Ratebeer	29073	110283	2844778	2274819	569959
BeerAdvocate	33368	66032	1585241	1268174	317067
Movies	1223167	164446	2985563	3676787	920365
Music	1133669	420466	3229487	2583532	645955
Yelp	366402	60784	1566139	1253144	312995

Tableau 1. *Détails des corpus d'évaluation*

Prédiction de revues

La similarité [7] étant basée sur le texte écrit par les utilisateurs, il est également légitime de l'utiliser pour extraire du texte des avis existant afin de le présenter en accompagnement de la recommandation. Deux possibilités sont étudiées:

– présenter l'avis complet le plus proche,

– créer un résumé personnalisé par extraction de phrases selon la méthode de (Kageback *et al.*, 2014).

Dans les deux cas, nous avons envisagé deux options:

– prendre les avis du même utilisateur pour d'autres produits (intérêt limité mais grande proximité de style d'écriture),

– proposer les avis des autres utilisateurs sur le produit cible.

4. Expériences

Nous utilisons notre espace latent pour différentes tâches. D'abord sur l'épreuve classique de prédiction de notes. Puis sur la prédiction des remarques textuelles, tâche originale introduite par (Poussevin *et al.*, 2014). Enfin, nous montrons les capacités d'extraction de mots-clés et d'analyse de sentiment de notre espace latent textuel.

Nous sélectionnons différents corpus d'avis en ligne afin d'évaluer nos systèmes de recommandation (tableau 1). Pour évaluer la prédiction de notes, nous utilisons l'erreur moyenne au carré (MSE), métrique standard de cette application. Nos résultats sont comparés avec ceux publiés dans (McAuley et Leskovec, 2013b). La prédiction de remarques étant une application sans méthode d'évaluation habituelle, nous gardons celle utilisée par (Poussevin *et al.*, 2014), afin de comparer nos résultats.

4.1. *Prétraitement des avis et paramètres d'apprentissage*

Afin d'obtenir l'ensemble d'apprentissage, il convient d'effectuer différents prétraitements sur les avis extraits. D'abord, tous les duplicatas d'avis sont retirés[2]. L'ensemble des marques de ponctuations à l'exception des points sont retirées. Les mots apparaissant moins de 10 000 fois sont supprimés, de ce fait chaque base d'avis contient entre 1500 et 3000 mots distincts. Les mots restants sont de véritables pivots du langage qui vont jouer le rôle des aspects latents des techniques de factorisation matricielle. Les avis sont enfin subdivisés en phrases reliées à leurs concepts à l'aide des points restants. Un extrait de phrases après prétraitement est présenté figure 3.

```
r_1.0  drinkable but not great
r_2.0  Not much else going on here
r_3.0  Spicy lactic dry taste
r_4.0  Taste quite bitter super clove esters big yeast
r_5.0  The best beer in the world
```

Figure 3. *Phrases après prétraitement (associées à leurs notes) extraites du corpus Ratebeer.*

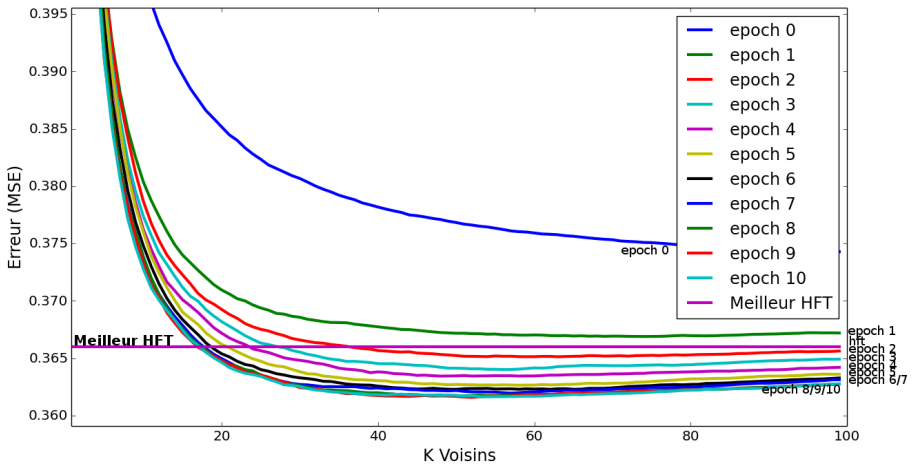

Figure 4. *Erreur en MSE en validation en fonction de la taille du voisinage $\mathcal{N}$ à chaque itération du modèle sur les données Beeradvocate. Le modèle converge en 5 à 8 itérations et le nombre de voisins optimaux à considérer varie de 9 à 33 selon les jeux de données.*

Nous suivons le protocole de (McAuley et Leskovec, 2013b) ; 80% des avis sont destinés à l'apprentissage et 20% divisés à parts égales pour la validation et l'éva-

2. Dans les corpus, certains avis apparaissent plusieurs fois pour différentes raisons. Il est important d'avoir peu de doublons afin d'obtenir des résultats non biaisés.

luation. Nous supprimons les avis où aucun historique d'utilisateur ou produit n'est disponible.

Nous utilisons une version modifiée de *Word2Vec* pour apprendre les vecteurs des utilisateurs, des produits et des notes. Les hyper paramètres sont les suivants: les vecteurs ont une taille de 200, le pas de gradient est 0.050 et le paramètre d'échantillonnage est de 10^{-5}. Nous itérons jusqu'à 10 fois sur le corpus d'apprentissage. Dans la pratique, la convergence est atteinte entre 5 et 8 itérations. Ces paramètres sont ceux proposés par (Mikolov *et al.*, 2013). Ils se sont révélés les mieux adaptés à notre problème. Enfin, le nombre k de voisins est choisi grâce à une phase de validation. Selon les jeux de données, le nombre de voisins permettant de minimiser la MSE varie de 9 à 33. La figure 4 montre l'évolution de la MSE en fonction du nombre d'itérations et du nombre de voisins. L'apprentissage est très rapide: peu d'itérations suffisent pour converger. Par contre l'inférence est plus chère, il est nécessaire de considérer un grand voisinage pour que la méthode fonctionne correctement.

Dataset	Moyenne	Fact. Mat	HFT*	CSG-kNN		k*
				util.	prod.	
Ratebeer	0.701	0.306	0.301	0.336	**0.286**	23
Beeradv.	0.521	0.371	0.366	0.382	**0.357**	29
Movies	1.678	**1.118**	1.119	1.39	1.304	33
Music	1.261	**0.957**	0.969	-	1.201	26
Yelp	1.890	1.49	-	1.591	**1.407**	27

Tableau 2. *Précision de la prédiction de note en MSE. La référence HFT* (meilleur modèle "Hidden Factor & Topic model") est reporté de (McAuley et Leskovec, 2013b). CSG-knn: Meilleur k-voisins dans l'espace d'avis utilisant k* voisins*

4.2. *Prédiction de Notes*

Pour prédire une note $\hat{r_{ui}}$ associée à un couple (utilisateur u, produit i), nous utilisons la méthode de filtrage collaboratif par voisinage, décrite par (Desrosiers et Karypis, 2011). Deux techniques classiques ont été évaluées: La prédiction grâce au voisinage utilisateur et celle grâce au voisinage produit [8]. Il s'agit d'agréger les notes des utilisateurs similaires ou les notes précédemment mises sur des produits proches.

Les résultats obtenus en prédiction des notes sont présentés dans le tableau 2. Les meilleurs résultats obtenus à l'aide de nos modèles de plus proches voisins (CSG-$k$nn utilisateur ou produit) sont comparés à un modèle qui prédit systématiquement la note moyenne du corpus d'apprentissage, un modèle de factorisation matriciel classique [3] et aux modèles HFT présentés par (McAuley et Leskovec, 2013b). Les résultats de HFT sur le corpus Yelp ne sont pas reportés, car le corpus utilisé est différent.

En règle générale nos performances sont proches de celles de (McAuley et Leskovec, 2013b) et même parfois meilleures. Notre modèle est particulièrement performant

sur les corpus des sites d'amateur de bières comme le montre la figure 4. Le vocabulaire utilisé y est spécifique et les revues particulièrement riches comparées aux revues type Amazon. Il est important de comprendre que contrairement aux autres modèles, l'espace latent n'est pas optimisé sur un critère de prédiction de note. Obtenir des résultats proches de l'état de l'art, voire meilleurs, souligne la pertinence de ces profils textuels pour la recommandation.

Sur Movies et Music, à l'inverse, le texte agit comme un bruit par rapport à la factorisation matricielle simple: les résultats du modèle HFT étaient légèrement impactés, mais notre plus grande dépendance au texte nous expose plus durement au phénomène.

4.3. *Problème du démarrage à froid*

Les techniques de factorisation matricielle se basent sur l'historique de notation de l'utilisateur et ne sont efficaces qu'à partir d'un certain nombre de voisins (> 20). Elles ne permettent donc aucune recommandation à un nouvel utilisateur ou pour un nouveau produit, c'est le problème du démarrage à froid. Avec notre méthode, il est possible d'utiliser des textes tiers relatifs aux nouveaux utilisateurs ou aux nouveaux produits (description, blogs, tweet...) et de les utiliser pour projeter les nouveaux éléments dans notre espace textuel.

Comme expliqué en section 3.2, la note prédite pour un couple (utilisateur, produit) dépend soit des notes données par les utilisateurs proches au produit cible, soit des notes données par l'utilisateur visé à des produits proches de la cible. Une fois le nouvel élément (utilisateur ou produit) positionné dans l'espace, et en utilisant l'approche adapté au nouvel entrant, notre système est immédiatement opérationnel sans devoir attendre la constitution d'un historique de notation. La seule différence avec [8] est que l'on ne normalise pas les notes par les moyennes utilisateurs et produit (ces données étant indisponibles dans le cas du démarrage à froid).

Cette stratégie est difficile à évaluer sur les corpus de revues où les utilisateurs et les produits ont été anonymisés. Nous proposons le protocole suivant:

1) deux ensembles de nouveaux utilisateurs et produits sont constitués,

2) dans ces ensembles, nous ne considérons que le texte (comme s'il avait été acquis depuis une source tierce),

3) nous apprenons des profils pour les nouveaux éléments et testons la prédiction de notes sur les données de test.

Ce protocole est certes biaisé, les données servant à constituer les profils étant exactement de la même nature que les données de test, mais il permet d'avoir une première idée de l'apport de notre stratégie.

Les résultats sont présentés dans le tableau 3. Il y a deux cas de figure à prendre en compte. L'arrivée de nouveaux utilisateurs et l'arrivée de nouveaux produits. On ob-

Dataset	Moy.	Nouvel Utilisateur		Nouveau Produit	
		Moy. Prod.	CSG-kNN	Moy. Util.	CSG-kNN
Ratebeer	0.701	0.341	**0.333**	0.599	**0.371**
Beeradv.	0.518	0.397	**0.386**	0.490	**0.419**

Tableau 3. *Précision de la prédiction de notes en MSE avec [4] pour simuler le démarrage à froid. Moy, Prod, Util: Prédis respectivement la note moyenne du corpus d'apprentissage,globale, par produit, par utilisateur.*

serve que systématiquement la prédiction grâce à l'espace latent est meilleure qu'en utilisant les moyennes produits ou utilisateurs (techniques la plus utilisée pour *démarrer* le processus lorsque le moteur de recommandation repose sur la factorisation matricielle).

Cette approche élégante au problème du démarrage à froid repose entièrement sur l'hypothèse d'un espace latent textuel sous-jacent au processus de recommandation. En production, il suffit d'imaginer un processus d'enregistrement des utilisateurs demandant une adresse Twitter, Facebook, un lien vers un site personnel ou une partie d'un historique de navigation web.

4.4. *Prédiction de remarques*

Nous suivons la proposition de (Poussevin *et al.*, 2014) sur la prédiction de remarques qu'un utilisateur u écrirait sur un produit i. Cette tâche peut à la fois être vue comme un enrichissement de la recommandation, une explication, mais également comme un résumé personnalisé des qualités et défauts des produits. Nous sommes convaincus que montrer des informations textuelles relatives aux caractéristiques produits et au ressenti utilisateur ne peut qu'être bénéfique pour expliciter la recommandation faite. La génération pure de texte restant une tâche complexe, nous adoptons une méthode d'extraction de résumés et nous focalisons sur deux techniques. Premièrement, nous sélectionnons un avis complet du produit i comme résumé. Sinon, nous filtrons n phrases parmi celles sur le produit i. (n est sélectionné en fonction de la moyenne de phrases qu'écrit l'utilisateur u.)

Les résultats obtenus sont évalués à l'aide des unigrammes et des bigrammes en utilisant les mesures ROUGE-1 et ROUGE-2 (Lin, 2004). Ces métriques montrent à quel point le texte prédit est égal au texte original. Elles peuvent être vues comme des mesures de rappel sur les n-grammes. N'ayant pas de score de référence, nous en calculons deux: une borne inférieure qui correspond à une sélection d'avis et de phrases faite au hasard. Et une borne supérieure étant un oracle choisissant systématiquement l'avis ou les phrases maximisant le score ROUGE-n possible.

avis complet								
Dataset	Rouge-1				Rouge-2			
	Rand	Prod.	Util.	Oracle	Rand	Prod.	Util.	Oracle
Ratebeer	0.243	0.271	0.351	0.568	0.037	0.052	0.117	0.278
Beeradv.	0.280	0.304	0.366	0.507	0.045	0.063	0.114	0.207
Music	0.255	0.407	0.403	0.545	0.043	0.212	0.215	0.255
Movies	0.253	0.345	0.351	0.565	0.040	0.146	0.156	0.208
Yelp	0.249	0.290	0.284	0.512	0.037	0.067	0.072	0.154
n phrases								
Ratebeer	0.157	0.239	0.217	0.591	0.011	0.043	0.046	0.288
Beeradv	0.122	0.225	0.244	0.606	0.010	0.046	0.057	0.249
Music	0.167	0.291	0.220	0.551	0.017	0.026	0.041	0.282
Movies	0.173	0.100	0.253	0.396	0.014	0.012	0.048	0.142
Yelp	0.150	0.246	0.279	0.550	0.011	0.041	0.050	0.208

Tableau 4. *Évaluation de la prédiction de remarques. Prod.: sélection des avis/-phrases dans les données produit (les avis des autres utilisateurs). Util.: sélection dans les données de l'utilisateur (ses propres avis sur des produits similaires)*

Vérité terrain

Note: 5.0
Commentaire: "Old school. traditional mom n'pop quality and perfection. The best fish and chips you ll ever enjoy and equally superb fried shrimp. A great out of the way non corporate vestige of Americana. You will love it."

Prédiction:

Note: 4.70
Avis complet: "Good fish sandwich"
Choix d'n phrases: "The staff is extremely friendly. On top of being extremely large portions it was incredibly affordable. Most of girls are good one is very slow one is amazing. The fish was very good but the Reuben was to die for. Both dishes were massive and could very easily be shared between two people."

Figure 5. *Exemple de prédiction de note et de remarques sur le corpus Yelp*

Le tableau 4 présente les résultats. En moyenne, le score ROUGE-1 est élevé, nous prédisons donc beaucoup de mots réellement utilisés dans l'avis original (de 10% à 40%). Néanmoins, le score élevé obtenu en utilisant le hasard montre que de nombreux mots doivent être présents dans tous les commentaires. Quoi qu'il en soit, nous sommes systématiquement meilleurs que cette borne inférieure. En comparant nos résultats avec ceux de (Poussevin *et al.*, 2014) sur les données Ratebeer, on peut remarquer que notre modèle est systématiquement meilleur sur l'extraction d'avis complet (+23% en ROUGE-1 et +246 % en ROUGE-2). Par contre sur la sélection de n phrases

les résultats sont plus contrastés: notre modèle a un score inférieur en ROUGE-1 (-30%), mais bien supérieur en ROUGE-2 (+ 150%). Une fois encore, il est important de noter que notre espace latent n'a pas été construit avec comme but d'optimiser ce score ROUGE-n.

Pour rappel, le but de la prédiction de remarque est d'être capable de présenter à l'utilisateur un avis complet ou un ensemble de phrases qui sont le plus en rapport avec ses intérêts. L'exemple présenté figure 5 illustre parfaitement cet objectif. Notre système de recommandation est capable de montrer à l'utilisateur différentes informations sur l'établissement, qui, comparativement à ses remarques, sont utiles. Comme le fait que leur sandwich au poisson est bon.

4.5. *Extraction de mots-clés et analyse de sentiment*

Notre espace étant textuel par nature, chaque concept est relié à différents mots par la similarité [7]. Il est donc possible d'extraire des mots-clés relatifs à un utilisateur, un produit ou une note. L'évaluation quantitative de cette tâche étant compliqué nous illustrons la possibilité d'extraction de mots-clés à l'aide de nuages associés aux notes 1/2/3/5 étoiles (figure 6) et à plusieurs bières (figure 7).

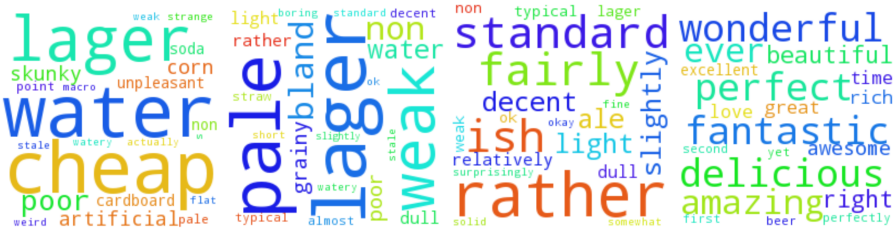

Figure 6. *Nuage de mots-clés associés aux notes (1/2/3/5 ⋆) du corpus Ratebeer*

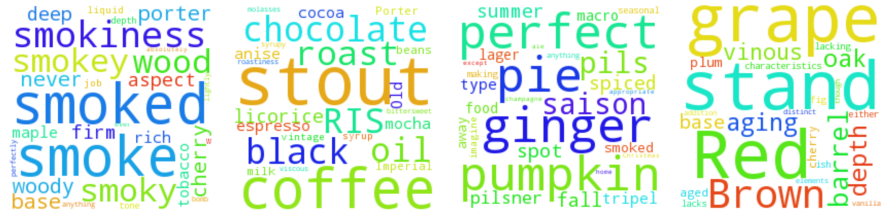

Figure 7. *Nuage de mots-clés associés à différentes bières du corpus Beeradvocate*

Ils démontrent la capacité d'extraire des mots relatifs aux concepts appris avec une certaine granularité.

5. Conclusion

La recommandation a pour objectif principal la personnalisation de l'accès à l'information via la construction de profils produit et utilisateur. Les techniques récentes de filtrage collaboratif par factorisation matricielle issue du concours Netflix se sont principalement focalisées sur la performance, laissant les explications liées aux suggestions au second plan.

La principale contribution de cet article est de mettre en avant le texte des avis et de s'en servir comme principale information pour la recommandation et son explication. Grâce à un espace latent textuel encodant les similarités entre utilisateurs et produits et en utilisant une technique de filtrage collaboratif basé sur cette même similarité, nous montrons qu'il est possible d'obtenir des résultats proches, voire meilleurs, que l'état de l'art sur la tâche de prédiction de note. Nous démontrons également les qualités de notre approche dans le cas d'un démarrage à froid en proposant une méthode élégante pour pallier ce problème. De plus, nous montrons qu'il est possible de prédire les remarques d'un utilisateur sur un produit avec une certaine efficacité.

Finalement, nos expériences démontrent qu'une similarité textuelle peut être utilisée pour concevoir un système de recommandation non seulement efficace en prédiction de notes mais plus large en terme d'usage. Cette nouvelle approche apporte à la fois une solution élégante en amont, pour le démarrage à froid, et des propositions intéressantes en aval avec la génération d'un texte à montrer à l'utilisateur en complément de la note.

6. Remerciements

Ce travail a été réalisé en partie avec le soutien du projet FUI ITER-RH (Investissement d'Avenir).

7. Bibliographie

Ben-elazar S., Koenigstein N., « A Hybrid Explanations Framework for Collaborative Filtering Recommender Systems », *RecSys 2014*, 2014.

Bennett J., Lanning S., « The Netflix Prize », *KDD Cup and Workshop*p. 3-6, 2007.

Blei D. M., Ng A. Y., Jordan M. I., Lafferty J., « Latent Dirichlet Allocation », *Journal of Machine Learning Research*, vol. 3, p. 2003, 2003.

Contardo G., Denoyer L., Artieres T., « Representation Learning for cold-start recommendation », , n° 1, p. 1-10, 2014.

Desrosiers C., Karypis G., « A comprehensive survey of neighborhood-based recommendation methods », *Recommender Systems Handbook*, vol. 69, n° 11, p. 107-144, 2011.

Ganu G., Elhadad N., Marian A., « Beyond the Stars : Improving Rating Predictions using Review Text Content », *WebDB*, vol. 9, p. 1-6, 2009.

Gauthier L.-A., Piwowarski B., Gallinari P., « Polarité des jugements et des interactions pour le filtrage collaboratif et la prédiction de liens sociaux », *CORIA*, 2015.

Goldberg Y., Levy O., « word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method », *arXiv preprint arXiv:1402.3722*, nº 2, p. 1-5, 2014.

Gutmann M., Hyvärinen A., « Noise-contrastive estimation: A new estimation principle for unnormalized statistical models », *International Conference on Artificial Intelligence and Statistics*, p. 1-8, 2010.

Guy I., Carmel D., « Social Recommender Systems », *Proceedings of the 20th international conference companion on World wide web*, Acm, p. 283-284, 2011.

Kageback M., Mogren O., Tahmasebi N., Dubhashi D., « Extractive Summarization using Continuous Vector Space Models », *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@EACL 2014*p. 31-39, 2014.

Koren Y., « Factorization meets the neighborhood: a multifaceted collaborative filtering model », *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*p. 426-434, 2008.

Koren Y., « Collaborative filtering with temporal dynamics », *Proc. of KDD '09*, ACM Press, p. 447-456, 2009.

Koren Y., Bell R., Volinsky C., « Matrix Factorization Techniques for Recommender Systems », *Computer*, vol. 42, p. 42-49, 2009.

Le Q., Mikolov T., « Distributed Representations of Sentences and Documents », *International Conference on Machine Learning - ICML 2014*, vol. 32, p. 1188–1196, 2014.

Levy O., Goldberg Y., Dagan I., « Improving Distributional Similarity with Lessons Learned from Word Embeddings », *Transactions of the Association for Computational Linguistics*, vol. 3, p. 211-225, 2015.

Lin C. Y., « Rouge: A package for automatic evaluation of summaries », *Proceedings of the workshop on text summarization branches out (WAS 2004)*p. 25-26, 2004.

Ling G., Lyu M. R., King I., « Ratings Meet Reviews , a Combined Approach to Recommend », p. 105-112, 2014.

McAuley J., Leskovec J., « From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews », *In Proceedings of the 22nd international conference on World Wide Web*p. 897-908, 2013a.

McAuley J., Leskovec J., « Hidden factors and hidden topics: understanding rating dimensions with review text », *Proceedings of the 7th ACM conference on Recommender systems - RecSys '13*p. 165-172, 2013b.

Mikolov T., Chen K., Corrado G., Dean J., « Distributed Representations of Words and Phrases and their Compositionality arXiv : 1310 . 4546v1 [cs . CL] 16 Oct 2013 », *arXiv preprint arXiv:1310.4546*p. 1-9, 2013.

Poussevin M., Guigue V., Gallinari P., « Extended Recommendation Framework: Generating the Text of a User Review as a Personalized Summary », 2014.

Tintarev N., Masthoff J., « A survey of explanations in recommender systems », *Proceedings - International Conference on Data Engineering*p. 801-810, 2007.