UNIVERSITÉ KASDI MERBAH OUARGLA

Faculté des Nouvelles Technologies de l'Information et de la Communication

Département d'Informatique et des Technologies de l'Information

Mémoire

Master en Informatique

Domaine : Mathématique et Informatique

Filière : Informatique

Spécialité : Administration et Sécurité des Réseaux

Présenté par : Belhachani Fouzia et Rouas Khadîdja

La recommandation des localisations : une nouvelle approche basée sur la fouille incrémentale des règles séquentielles.

Soutenu le : 22/09/2020

Devant le jury composé de :

Dr. Amirat Hanane	Encadreur	UKM Ouargla
Mr. Harouz Abdelhakim	Président	UKM Ouargla
Dr. Korichi Merieme	Examinateur	UKM Ouargla

Remerciement

Tout d'abord, nous remercions Dieu Tout-Puissant pour le courage et la volonté qu'il nous a donnés pour mener à bien cette tâche.

Nous remercions chaleureusement Mme ***Amirat Hanan*** *d'avoir accepté d'être l'encadreur de notre thèse. Nous la remercions pour le soin avec lequel elle a lu ce mémoire, ainsi que pour ses conseils et ses remarques pertinentes.*

Nous remercions en Particulièrement nos familles (nos parents) qui ont su nous soutenir, nous encourager, nous aider et nous supporter tout au long de l'année.

Nous tenons à remercier les membres de notre jury pour avoir bien voulu consacrer une partie de leurs temps à examiner et à évaluer ce travail. Nous remercions tous nos professeurs et nos camarades qui ont veillé sur nous pendant ces cinq (05) années d'études supérieures.

Rouas Khadîdja

Belhachani Fouzia

Dédicace

À nos chers parents pour tous leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de nos études. À toutes nos familles (frères et sœurs), et nos amis pour les soutenir tout au long de notre carrière universitaire. À toute personne qui nous a donné son soutien, encouragement, conseils et orientations pour la réalisation de ce travail

Tables des matières

1 Table des matières

Tables des matières

Tables des matières

Liste des tableaux

Tables des figures

Liste des figures

Résumé

Résumé

Ces dernières années, le développement rapide dans le domaine technologique a stimulé le développement des technologies d'acquisition de localisation et de communication mobile Ce développement a créé de nombreux services de localisation, tels que la recommandation de points d'intérêt (POI). La recommandation des localisations consiste à suggérer à un utilisateur des lieux qu'il pourrait être intéressé à visiter. L'objectif de ce service est d'aider les utilisateurs mobiles à découvrir de nouveaux lieux intéressants (par exemple, des restaurants et des magasins), en déplacement.

Dans la littérature, de nombreux modèles de recommandation des POI ont été proposés. Ces modèles tiennent compte de divers facteurs tels que l'influence géographique, temporelle et sociale. Bien que ces modèles se soient avérés performants, peu d'entre eux prennent en compte le comportement séquentiel de la mobilité humaine.

En outre, plusieurs modèles ont été conçus en tenant compte des données statiques, ignorant ainsi les collectes continues des données d'enregistrement dans les RSBL (Réseaux Sociaux Basés sur la Localisation). Cela a conduit à la conception des systèmes de recommandations qui doivent être construit à partir de zéro pour effectuer des prédictions actualisées, lorsque de nouvelles données d'enregistrement arrivent. La complexité temporelle et spatiale de ces systèmes de recommandation peut donc augmenter considérablement lorsqu'elle est appliquée sur des données dynamiques. Par conséquent, il est nécessaire de concevoir des systèmes de recommandeurs incrémentales capables de traiter efficacement les données dynamiques.

Pour remédier ces limitations, ce mémoire propose une nouvelle approche de recommandation des points d'intérêt, appelée *STS-Rec*. Ce dernier, basé sur l'extraction des règles séquentielles et il prend en compte le comportement séquentiel de la mobilité humaine. *STS-Rec* transforme d'abord les données de mobilité en séquences de localisation. Ensuite, il extrait progressivement des règles de recommandation séquentielles à partir de ces séquences.

Une évaluation expérimentale menée sur un jeu de données d'enregistrement réel à grande échelle montre que le modèle proposé surpasse la version statique (non incrémentale) de système en termes de temps d'exécution et l'espace mémoire occupé.

Mots Clés : extraction incrémentale, règles séquentielles, recommandation de POI, modèle basé sur l'arbre.

Résumé

Abstract

In recent years, the rapid development in the technological field has stimulated the development of location acquisition and mobile communication technologies. This development has created many location services, such as the recommendation of points of interest (POI). POI recommendation consists of suggesting places that a user might be interested in visiting. The purpose of this service is to help mobile users to discover new and interesting places (for example, restaurants and shops), while on the move.

In the literature, many models for recommending POIs have been proposed. These models take into account various factors such as geographic, temporal and social influences. Although these models have proven to be efficient, few of them take into account the sequential behavior of human mobility.

Additionally, several models have been designed with static data in mind, thus ignoring the continuous recording and collection of check-in data in the LBSN (Location Based Social Network). This has led to the design of recommendations that must be formed from scratch to make up-to-date predictions, when new check-in data arrives. The temporal and spatial complexity of these recommenders can therefore increase considerably when applied to dynamic data. Therefore, there is a need to design incremental recommenders capable of handling dynamic data efficiently.

To accommodate these limitations, this thesis proposes a new point of interest recommendation approach, called STS-Rec. The latter is mainly based on the extraction of sequential rules and it takes into account the sequential behavior of human mobility. STS-Rec first transforms mobility data into location sequences. Then, it gradually extracts sequential recommendation rules from these sequences.

An experimental evaluation conducted on large-scale real-life check-in data from Brightkite shows that the proposed model outperforms the static version of the system in terms of execution time and space.

Keywords: incremental mining, sequential rules, POI recommendation, tree-based model.

Résumé

ملخص

في الســـنوات الأخيرة، أدى التطور الســـريع في المجال التكنولوجي إلى تحفيز تطوير تقنيات الاســـتحواذ على المواقع والاتصالات المتنقلة، وقد أدى هذا التطور إلى إنشاء العديد من خدمات تحديد المواقع، مثل التوصية بنقاط الاهتمام (POI) التوصـــية هي اقتراح الأماكن التي قد يهتم المســـتخدم بزيارتها. الغرض من هذه الخدمة هو مســـاعدة مســـتخدمي الهاتف المحمول على اكتشاف أماكن جديدة ومثيرة للاهتمام (على سبيل المثال، المطاعم والمحلات التجارية) أثناء التنقل.

في الدراسات السابقة، تم اقتراح العديد من النماذج للتوصية بالنقاط المهمة. تأخذ هذه النماذج في الاعتبار عوامل مختلفة مثل التأثير الجغرافي والزمني والاجتماعي على التوصية. على الرغم من أن هذه النماذج أثبتت فعاليتها، إلا أن القليل منها يأخذ في الاعتبار السلوك المتسلسل للتنقل البشري.

بالإضافة إلى ذلك، تم تصميم العديد من النماذج مع وضع البيانات الثابتة في الاعتبار، وبالتالي تجاهل التسجيل المستمر وجمع بيانات التسجيل في RSBL وقد أدى ذلك إلى تصميم التوصيات التي يجب تشكيلها من البداية لعمل تنبؤات محدثة، عند وصول بيانات قياسية جديدة. وبالتالي يمكن أن يزيد التعقيد الزمني والمكاني لهؤلاء الموصيين بشكل كبير عند تطبيقهم على البيانات الديناميكية. لذلك، هناك حاجة إلى تصميم توصيات إضافية قادرة على التعامل مع البيانات الديناميكية بكفاءة.

لمعالجة هذه القيود، تقترح هذه الأطروحة نهجًا جديدًا لتوصية نقاط الاهتمام يسمىSTS-Rec ، استنادًا إلى استخراج القواعد المتسلسلة، والتي تأخذ في الاعتبار السلوك المتسلسل للتنقل البشري. يقوم STS-Rec أولاً بتحويل بيانات التنقل إلى تسلسلات الموقع. بعد ذلك، يستخرج تدريجياً قواعد التوصية المتسلسلة من هذه التسلسلات.

أظهر تقييم تجريبي تم إجراؤه على مجموعة، من بيانات التسجيل المباشر على نطاق واسع من Brightkite أن النموذج المقترح يتفوق على نموذجين متسلسلين متقدمين من حيث وقت النفيذ ومساحة الذاكرة.

الكلمات الرئيسيةً: الاستخراج المتزايد، القواعد المتسلسلة، توصية POI، النموذج المستند إلى الشجرة.

Résumé

Introduction générale

Ces dernières années, les progrès de la technologie mobile et de l'utilisation des appareils mobiles ont causé l'émergence d'un grand nombre de réseaux sociaux basés sur la localisation (RSBL). Ces RSBLs, tels que Foursquare et Gowalla, ont attiré des millions d'utilisateurs générant des milliards de points d'intérêt d'enregistrement dans leurs bases de données.

Chaque utilisateur d'un RSBL maintient un profil contenant ses informations (nom, âge, adresse, etc.) et signale ses lieux d'enregistrement dans le RSBL. Un enregistrement effectué par un utilisateur indique un lieu visité, également appelé point d'intérêt (POI Point of Interest) (par exemple, un centre commercial), le temps de visite, les coordonnées géographiques (GPS) etc. L'un des principaux services offerts par les RSBLs est la recommandation de POI, qui consiste à exploiter les données d'enregistrements pour suggérer des points d'intérêt à un utilisateur. Une recommandation efficace porte un profit aux utilisateurs et aux fournisseurs de RSBL. Elle permet aux utilisateurs de profiter des expériences précédentes d'autres utilisateurs (en particulier des amis) pour décider si un emplacement est intéressant et doit être visité. En général, la recommandation de localisation est plus utile pour les utilisateurs qui visitent des zones inconnues. Pour les fournisseurs d'un RSBL, un profit économique est réalisé en annonçant des emplacements aux visiteurs potentiels.

Au cours de la dernière décennie, la recommandation de POI est devenue un sujet de recherche populaire ou de nombreuses approches de recommandations des POIs ont été proposées. La technique la plus utilisée pour la recommandation est le filtrage collaboratif (FC) [3]-[7]. Cette technique exploite les évaluations et les préférences des utilisateurs pour effectuer des recommandations.

D'une manière générale, il est souhaitable que les approches de recommandation de POI prennent en compte de nombreux facteurs, notamment les influences géographiques [10], temporelles [12] - [13] et sociales [2] - [11] sur le comportement de mobilité des utilisateurs. Pour cette raison, de nombreuses études ont étendu les approches traditionnelles des FC pour tenir compte de ces facteurs. Le facteur d'influence géographique est basé sur l'hypothèse qu'un utilisateur est plus susceptible de visiter des sites proches que des sites lointains. Le facteur

d'influence temporelle est essentiel pour la recommandation. Il indique qu'une destination utilisateur dépend généralement du temps. Par exemple, un utilisateur peut avoir tendance à se

rendre sur son lieu de travail le matin et à manger dans un restaurant le midi les jours de travail dans la semaine, alors qu'il peut visiter des centres commerciaux dans les week-end. Enfin, le facteur d'influence sociale est basé sur l'idée que les amis partagent souvent des intérêts, des goûts et des préférences communs. En d'autres termes, les utilisateurs peuvent suivre les recommandations de leurs amis et visiter les mêmes endroits. Par exemple, des amis peuvent s'entraîner dans le même gymnase et parfois manger dans le même restaurant.

Motivations

Bien que les systèmes de recommandation des FC traditionnels aient montré leur efficacité, une limitation majeure de ces systèmes est qu'ils ne tiennent pas en compte l'emplacement actuel de l'utilisateur lorsqu'il recommande le prochain emplacement. Ces recommandeurs ignorent donc la nature séquentielle de mobilité humaine dans la recommandation [18]. En d'autres termes, les systèmes de recommandation des FC utilisent les données d'enregistrement, mais ne considèrent pas que les besoins des utilisateurs varient en fonction de leurs déplacements et emplacements récents.

Récemment, la tâche des recommandations successives des POI [26], [28] a été largement étudiée. Les recommandeurs pour cette tâche tiennent en compte des relations séquentielles masquées dans le comportement de mobilité des utilisateurs pour effectuer des recommandations. En particulier, compte tenu de l'emplacement actuel d'un utilisateur et de ses mouvements historiques (si disponibles), un ensemble de suggestions de POI sont générés vers des emplacements que l'utilisateur peut être intéressé à visiter dans le future proche. La plupart de ces systèmes utilisent des modèles basés sur des chaînes de Markov [14] [15] ou des modèles basés sur l'extraction des motifs fréquents séquentiels [16] [17].

Bien que ces modèles fonctionnent bien, leur principal inconvénient est leur sensibilité au changement de l'ordre de visite des localisations. La plus petite déviation dans l'ordre de visite affecte la précision de la recommandation.

Un autre problème avec les approches séquentielles de modèle ou d'exploration de règles séquentielles est qu'elles échouent généralement dans la gestion des données dynamiques. En fait, la plupart des approches sont conçues pour gérer des données statiques. Mais dans la vie réelle, de nouvelles données d'enregistrement arrivent en permanence à un rythme rapide (un flux de données). Ainsi, les modèles découverts dans les données d'enregistrement peuvent

devenir rapidement obsolètes. Étant donné qu'un bon nombre de ces approches ne fournissent pas de mécanisme de mise à jour, elles doivent être appliquées à partir de zéro lorsque de nouvelles données arrivent pour effectuer des recommandations à jour. Pour cette raison, la complexité temporelle et spatiale de ces approches peut augmenter de façon rapide lorsqu'elles sont appliquées à des données dynamiques, ce qui les rend inefficaces. Ceci est particulièrement un problème lorsque les données sont fréquemment mises à jour.

La résolution de ce problème nécessite de concevoir des modèles incrémentaux pour l'exploration motifs séquentiels dans les flux continus des données d'enregistrements. Cependant, l'extraction de motifs séquentiels est un problème récent et difficile pour la recommandation des POI et la communauté d'exploration de données. À notre connaissance, certains algorithmes incrémentaux ont été proposés pour l'exploration de motifs séquentiels et de règles d'association, mais aucun algorithme n'a été proposé pour l'extraction incrémentale des règles séquentielles pouvant être utilisées pour identifier de fortes relations temporelles entre les emplacements d'enregistrement. En outre, la plupart des travaux sur l'extraction incrémentale de motifs se sont concentrés sur la recherche de motifs séquentiels et de règles d'association. Mais ces deux problèmes sont différents de l'exploration séquentielle de règles puisque cette dernière doit non seulement prendre en compte l'ordre séquentiel entre les éléments d'un motif, mais également la fréquence d'occurrence et la confiance du modèle.

Contribution

Pour remédier les limitations citées ci-dessus, nous proposons dans ce mémoire, un système de recommandation basé sur l'exploitation incrémentale des règles séquentielles, appelé ***STS-Rec*.** Ce dernier aborde les principaux inconvénients des approches d'exploration de motifs séquentiels pour effectuer la recommandation de POI. *STS-Rec* découvre efficacement l'évolution du comportement séquentiel périodique de la mobilité humaine en relâchant la contrainte d'ordre stricte sur les séquences de localisation en découvrant les règles POSR (Partialy Ordered Sequentiel Rules).

De plus, *STS-Rec* étend également le concept de motifs séquentiels en utilisant une fenêtre coulissante (une contrainte de taille de fenêtre). Cette contrainte permet d'extraire des motifs de recommandation d'emplacements qui apparaissent dans un nombre maximum d'emplacements consécutifs dans des séquences d'emplacements. Grâce à cette technique, l'ordre d'apparition des emplacements est considéré pour éviter de recommander des

emplacements éloignés.

En outre, nous avons proposé une version incrémentale de l'outil de recommandation *STS-Rec* pour gérer les données dynamiques là où de nouveaux lieux d'enregistrement peuvent arriver à tout moment et où de nouveaux utilisateurs peuvent rejoindre le RSBL. L'exploration progressive des règles de recommandation est bénéfique car elle maintient le système de recommandation à jour et au courant des nouvelles tendances. Ainsi, le système de recommandation peut fournir des recommandations plus pertinentes. L'approche incrémentale met à jour les règles en traitant des lots de nouvelles séquences de localisation. L'approche utilise une structure compacte d'arbre et une structure bitmap pour stocker les règles précédemment extraites. Ensuite, l'arbre est utilisé pour mettre à jour efficacement l'ensemble des règles de recommandation (en évitant d'effectuer des calculs redondants).

En bref, notre mémoire apporte les contributions suivantes :

- Un nouveau système de recommandation appelé STS-Rec est proposé pour découvrir des règles de recommandation qui prennent en compte le facteur d'influence séquentielle.
- L'utilisation de règles séquentielles partiellement ordonnées (POSR) dans la recommandation.
- Un algorithme est présenté appelé *IncrPOSR* pour extraire de manière incrémentale les règles séquentielles. Plus précisément, nous concevons un arbre compact de règles pour stocker les règles séquentielles précédemment trouvées avec des informations pertinentes, afin qu'elles puissent être rapidement mises à jour. Une structure de données bitmap appelée *SequenceIndex* est également adoptée pour accélérer les calculs de fréquence. Elle permet de calculer la fréquence d'une règle mise à jour sans scanner à nouveau les séquences de données d'enregistrement précédentes. D'autres structures ont été également proposées pour améliorer l'efficacité.
- Une étude expérimentale a été menée à l'aide de jeu de données d'enregistrement RSBL réels pour évaluer les performances de notre système de recommandation.

Organisation de chapitres

Les chapitres sont organisés comme suit.

Introduction générale

- Dans le premier chapitre, nous passons en revue sur certains travaux connexes dans le domaine de la recommandation de localisation, puis nous présentons une taxonomie qui classe ces travaux en fonction du facteur d'influence considéré.
- Dans le deuxième chapitre, nous présentons le concept la fouille de donnée. Nous concentrons le plus sur l'extraction des règles d'association que nous avons utilisée dans notre étude.
- Dans le troisième chapitre, nous fournissons une description détaillée de notre approche.
- Dans le quatrième chapitre nous évaluons les performances de notre proposition contre la version non incrémentale de notre système.
- Enfin, nous terminons la mémoire par une conclusion.

Chapitre 1

L'ETAT DE L'ART

Chapitre 1

1 Introduction

Ces dernières années, les réseaux sociaux basés sur emplacement (RSBL) ont attiré un nombre démesuré d'utilisateurs, car ils fournies plusieurs services, l'un de ces services est la recommandation de points d'intérêt (POI ou Points Of Interest). Ce service vise principalement à suggérer aux utilisateurs des lieux qu'ils pourraient être intéressés à visiter dans le future.

Dans la littérature, de nombreux modèles ont été proposés pour soutenir les systèmes de recommandation conçus pour la recommandation d'emplacement. Ces modèles tiennent compte des divers facteurs tels que l'influence séquentielle, géographique, temporelle et sociale sur la recommandation.

Dans ce chapitre, nous allons définir et expliquer quelques notions ainsi que les concepts fondamentaux liés aux systèmes recommandations, nous présentons ensuite les principaux facteurs d'influence sur la recommandation des POIs, puis une petite présentation et formulation de problème de recommandation des POIs seront présentées. Enfin nous allons passer en revue sur les principales études sur la recommandation des POIs en concentrant sur celles utilisant des informations séquentielles.

2 Réseaux Sociaux Basés Sur la localisation (RSBL)

Un réseau social basé sur la localisation (RSBL) est une structure abstraite qui contient diverses relations entre les individus, telles que des amitiés, des intérêts partagés et des connaissances partagées. Le service de réseau social en ligne est une représentation numérique et participative des réseaux sociaux dans le monde réel.

Les services de réseaux sociaux révèlent de réelles connexions sociales aux utilisateurs et améliorent également la croissance en leur permettant de partager et de transmettre des idées, des activités tel que leur position géographique (détecte ou proposé un emplacement pour un utilisateur), des événements, des nouvelles et des intérêts d'une manière beaucoup plus facile.

En outre, l'utilisation rapide des appareils intelligents a conduit à prospérité et la popularité des réseaux sociaux basés sur la localisation (emplacement), tels que Foursquare [1].

Ces réseaux sont considérées une source riche des données d'enregistrement des utilisateurs, y compris leurs enregistrements GPS, les sites visités les commentaires sur ces emplacements ainsi que d'autres données contextuelles, par exemple, Figure 1.1 montre comment les informations d'enregistrements notamment le notamment le nom d'utilisateur, le POI,

l'horodatage d'enregistrement et la position géographique sur la carte nom, sont collectés dans Foursquare.

Un RSBL permet également aux utilisateurs de s'identifier, de créer des amitiés et de partager des informations.

Figure 1.1 : La collecte des informations d'enregistrement dans Foursquare.

3 La recommandation des POIs

La recommandation des POIs consiste à exploiter les données d'enregistrement des utilisateurs pour suggérer un ou plusieurs emplacements aux utilisateurs des RSBLs qu'ils sont susceptibles de les visiter à l'avenir, dans cette section, nous allons tout d'abord définir les termes clés de la recommandation des POI, puis donner la formulation du problème de la recommandation des POI dans les RSBLs.

Dans cette même section, nous allons ainsi l'ensemble de facteurs qui influencent la recommandation. Enfin, nous allons passer en revue les approches de recommandation considérants le facteur séquentielles.

Chapitre 1

4 Préliminaires

Définition 1(Check-in). Une check-in ou enregistrement E_i est le triplet E_i=<u_i, ti, E_{id}> indiquant un utilisateur u_i visitant POI E_{id} à l'instant t_i.

Définition 2 (Lieu ou POI). Un emplacement est une zone géographique, qui a un identifiant unique.

Définition 3 (Séquence d'emplacement, sous-séquence d'emplacement). Une séquence d'emplacement (localisation) ou une séquence de check-in $S = E_i, E_{i+1}, .., E_m$ est une liste ordonnée des emplacements visitées par un utilisateur u_i pendant une période de temps T, à titre d'illustration T, tableau 1.1 montre un échantillon des séquences de Check-in d'un ensemble de trois utilisateurs $U : \{u_1, u_2, u_3\}$, représentant leur activité check-in dans trois jours (j_1, j_2, j3) [39].

Dans cet exemple, la séquence s_1 indique que l'utilisateur u_1, a visité l'emplacement E_0 suivi de E_1 et E_2 dans le jour J_1.

Une séquence d'emplacement $S_1 = E_{x1}, E_{x2,\ldots}, E_{xn}$ est appelée une sous- séquence d'une séquence $S_2 = E_{y1}, E_{y2,\ldots}, E_{ym}$ (notée $S_1 \subseteq S_2$), si et seulement si les entiers $1 \leq a_1 < a_2 < \ldots < a_n \leq m$ tels que $E_{x1}=E_{ya1}$, $E_{x2}=E_{ya2}$, $E_{xn}=E_{yan}$, existent.

Utilisateur	**Jour**	**Check-in**	**Séquence d'emplacement**
U_1	J_1	<U_1 ,8 :00, E_0>< U_1, 10 :00, E_1><U_1 ,14 :00, E_2>	$S_1 : E_0, E_1, E_2$
	J_2	<U_1 ,8:30, E_2>< U_1, 11 :50, E_9><U_1 ,18 :00, E_{14}>	$S_2 : E_2, E_9, E_{14}$
	J_3	<U_1 ,6 :00, E_8>< U_1, 7 :10, E_5><U_1 ,15:10, E_9>	$S_3 : E_8, E_5, E_9$
U_2	J_1	<U_2 ,7 :00, E0>< U_2, 10 :20, E_5><U_2 ,11 :50, E_2>	S_4: $E_0, E_5, E2$
	J_2	<U_2 ,8 :00, E_5>< U_2, 11 :20, E_8><U_2 ,19 :30, E_{12}>	$S_5 : E_5, E_8, E_{12}$
	J_3	<U_2 ,5 :30, E_2>< U_2, 12 :00, E_4><U_2 ,14 :20, E_{22}>	$S_6 : E_2, E_4, E_{22}$
U_3	J_1	<U_2 ,8 :50, E_4>< U_2, 12:20, E_9><U_2 ,18 :30, E_{11}>	$S_5 : E_4, E_9, E_{11}$
	J_2	<U_2 ,09:00, E_2>< U2, 12:20, E_3><U_2 ,14 :30, E_{12}>	$S_5 : E_2, E_3, E_{12}$

Tableau 1. 1: Un échantillon de séquences d'emplacement.

Définition 5 (Fréquence d'un motif). *La fréquence* d'un motif *Pi*, notée comme *Fréquence* (P_i) désigne le nombre de fois qu'un utilisateur a visité les emplacements de P_i dans *LH* sur le nombre total de séquences dans *LH*. P_i est considéré *fréquent* si sa *fréquence* dépasse un paramètre défini par l'utilisateur appelé *minfreq* (*fréquence* minimale) (c.-à-d. *fréquence* (P_i)>= *minfreq*).

Définition 6 (règle de recommandation). Soit *E* l'ensemble de toutes les emplacements dans une zone géographique *Z*.

Une règle de recommandation $R: P_1 \rightarrow P_2$ est une relation entre deux motifs d'emplacements *fréquents* $P_i = \{E_x, E_{x+1}, \dots E_n\}$ et $P_j = \{E_y, E_{y+1}, .., E_m\}$ tels que $P_1 \cap P_2 = \emptyset$ et $P_1, P_2 \neq \emptyset$. P_1 et P_2 sont appelés l'antécédent et le conséquent de *R*, respectivement. La règle *R* est interprétée comme si un utilisateur a visité les emplacements dans P_1, il visitera ensuite ceux de P_2.

5 Les facteurs d'influence

La recommandation des POIs est affectée par plusieurs facteurs notamment le facteur social, le facteur géographique, le facteur temporel ainsi que facteur séquentiel. Dans ce qui suit, nous allons présenter une description détaillée sur chaque facteur.

5.1 L'influence sociale

L'influence sociale est un facteur essentiel dans la recommandation des POIs, ce facteur base principalement sur de l'hypothèse que des amis d'un utilisateur partagent des préférences, ainsi que des intérêts similaires et peuvent visiter par conséquent les mêmes endroits.

5.2 L'influence géographique.

La proximité géographique des points d'intérêt affecte considérablement les comportements d'enregistrement check-in des utilisateurs vis-à-vis des points d'intérêt. Elle dépend de l'hypothèse que l'utilisateur est plus susceptible de visiter des sites proches par rapport les sites distants, par conséquent, l'effet des informations de géolocalisation sur les comportements check-in a été largement utilisé et considéré dans recommandation d'emplacement.

5.3 L’influence temporelle

Le facteur temporel est un facteur important pour la recommandation. Il indique qu’une destination d’utilisateur dépend généralement sur le temps. En effet, l'influence temporelle est d'une importance vitale pour la recommandation des POIs.

Les contraintes physiques sur l'activité check-in conduisent à des schémas spécifiques qui se réfèrent à la relation entre les lieux et les informations de temps.

Les utilisateurs ont la tendance de toujours visiter certain endroits et les quitte dans une heure bien définie (ex : l’utilisateur va au travail ou au restaurant et lier avec temps. L’historique utilisateur repose donc sur des séquences temporelles (séquences d’emplacement temporel) qui intègrent l'heure dans chaque POI.

5.4 L’influence séquentielle

Le mouvement humain présente une régularité séquentielle [18].D’où des motifs séquentiels qui peuvent être extraits des séquences de lieux d'enregistrement des utilisateurs. Par exemple, les gens vont généralement aux cinémas après les restaurants car ils souhaitent se détendre après le dîner. L'influence séquentielle sur l’activité d’enregistrement des utilisateurs a devenue de plus en plus importante dans les recommandations pour améliorer la recommandation des POI, Figure 1.2 présente un exemple de l’''influence séquentielle.

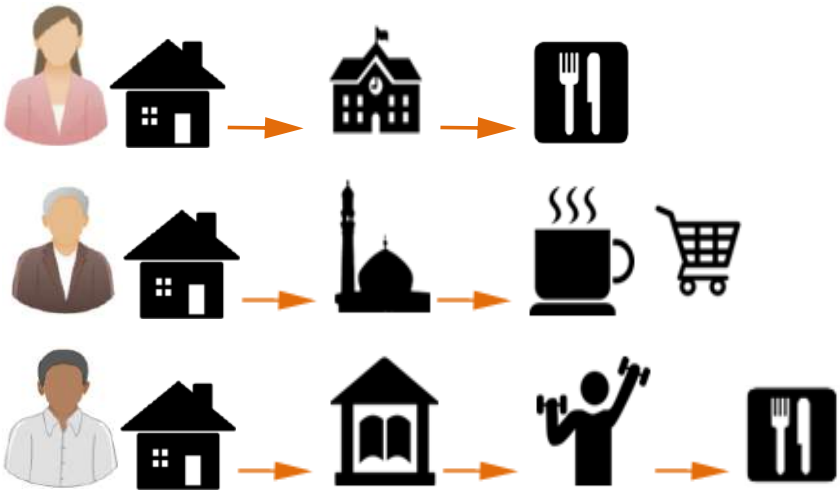


Figure 1.2: Le comportement séquentiel de mouvement de trois utilisateurs.

6 Formulation de problème de recommandation des POIs

Compte tenu des historiques d'emplacement LH d'un ensemble d'utilisateurs *U*, les facteurs d'influence être considérés ainsi que la séquence d'emplacement *S* d'un utilisateur *ui* (contenant sa position actuelle ses positions précédemment visitées.

La recommandation de POIs vise à suggérer à u_i un ou plusieurs emplacement qu'il est susceptible de les visiter en s'appuyant sur les règles de recommandation générées à partir de LH.

7 Modèles de base de recommandation

La recommandation des POIs est une branche des systèmes de recommandation d'où, les techniques conventionnelles des systèmes de recommandations, ex : le modèle de Markov ou le filtrage collaboratif ont été utilisées.

7.1 Modèle de Markov

Le modèle de Markov, également appelé les chaînes de Markov, est un modèle graphique d'un état simple ou une transition. A une fonction de transition de probabilité à chaque étape de temps. Chaque transition est liée à sa probabilité d'être assumée et cette probabilité peut être zéro.

Le modèle de Markov simple base principalement sur l'hypothèse indiquant que l'état précédent de système n'a aucun effet sur l'atteinte de l'état futur. D'où, ce dernier dépend seulement de l'état actuel. En Sens mathématique, le modèle de Markov peut se formuler comme suit : Soit n le nombre des états dans un système donné.

Pour avant les valeurs $i_1, ..., i_n, i_{n+1}$, La probabilité que l'état X_{n+1} prenne la valeur i_{n+1} sachant que $X_1=i_1, X_1=i_1,..., X_{n-1}=i_{n-1}$ et $X_n=i_n$ ne dépend que de i_{n+1} et de i_n, c'est-à-dire :

$$P(X_{n+1}=i_{n+1} | X_1 = i_1, \ldots\ldots, X_n = i_n) = P(X_{n+1} = i_{n+1} | X_n = i_n).$$

Le modèle de Markov d'ordre *k* suppose que les futurs états de système dépendent uniquement des états les plus récents. Autrement dit, si l'ensemble H_n des différents états de l'utilisateur se compose de $Hn = \{X_1 = i_1, ..., X_n = i_n\}$, alors, pour tous un a∈A *P*

$$(X_{n+1} = i | H_n) = P(X_{n+1} = i | X_{n-k+1} = i_{n-k+1} \ldots X_n = i_n) = P(X_{i+k+1} = i | X_{i+1} = i_{n-k+1} \ldots X_{i+k} = i_n), V$$

$I \in N$. Ce qui indique que le futur est conditionné par le passé est le future conditionné par le présent.

Le modèle de Markov été aussi étendu introduisant la notion des « chaînes de Markov cachées » (CMC) ou, *Hidden Markov model* (HMM). Ce dernier est un modèle de Markov, sauf qu'on ne peut pas observer directement la séquence d'états : les états sont cachés. Chaque état émet des "observations" qui, elles, sont observables. On ne travaille pas donc sur la séquence d'états, mais sur la séquence d'observations générées par les états.

7.2 Le filtrage collaboratif

Le filtrage collaboratif (de l'anglais : collaborative filtering) est une technique utilisée par les systèmes de recommandation, qui fonctionne sur des prédictions automatiques basées sur l'utilisation d'opinions, de notes, de comportements et de goûts exprimés par de nombreux autres utilisateurs afin d'aider l'utilisateur à recommander.

L'hypothèse fondamentale de FC est que si les utilisateurs *X* et *Y* notent n éléments de manière similaire, ou ont des comportements similaires (par exemple, acheter, regarder, écouter), et donc évaluer ou agir sur d'autres éléments de manière similaire, Les techniques FC utilisent une base de données de préférences, pour les articles par les utilisateurs afin de prévoir des sujets ou des produits supplémentaires qu'un nouvel utilisateur pourrait aimer.

Il existe trois techniques de filtrage collaboratif soit FC basé sur la mémoire, FC basé sur le modèle ou FC hybrides.

7.2.1 FC basé sur la mémoire

Les algorithmes FC basés sur la mémoire utilisent la totalité ou un échantillon de la base de données des éléments utilisateur pour générer une prédiction. Chaque utilisateur fait partie d'un groupe de personnes ayant des intérêts similaires. En identifiant les soi-disant voisins d'un nouvel utilisateur (ou utilisateur actif), une prédiction des préférences sur de nouveaux éléments pour lui peut être produite.

Par exemple, la méthode de l'algorithme FC basé sur le voisinage utilise les étapes suivantes : calculer la similitude ou le poids, $w_{i,j}$, qui reflète la distance, la corrélation ou le poids, entre deux utilisateurs ou deux éléments, *I* et *j* ; produire une prédiction pour l'utilisateur actif en prenant la moyenne pondérée de toutes les évaluations de l'utilisateur ou de l'article sur un certain article ou utilisateur, ou en utilisant une moyenne pondérée simple [4].

Lorsque la tâche consiste à générer une recommandation *top N*, nous devons trouver *k* utilisateurs ou éléments les plus similaires (voisins les plus proches) après avoir calculé les

similitudes, puis agréger les voisins pour obtenir les éléments *N* les plus fréquents comme recommandation.

7.2.2 FC basé sur le modèle

Dans cette approche, la conception et le développement de modèles sont développés à l'aide de différents algorithmes d'exploration de données et d'apprentissage automatique pour prédire les classifications des éléments non évalués par les utilisateurs. Il existe de nombreux algorithmes FC basés sur des modèles. Réseaux bayésiens, modèles de regroupement, modèles sémantiques latents tels que la décomposition en valeurs singulières, analyse sémantique latente probabiliste, facteur multiplicatif multiple, allocation latente de Dirichlet et modèles basés sur le processus de décision de Markov.

7.2.3 FC hybride

Ce modèle intègre des applications entre les algorithmes FC basés sur la mémoire et les algorithmes FC. Il s'agit de surmonter les problèmes de perte d'informations dans les recommandations FC surtout résoudre le problème de l'insuffisance des données telles que et d'améliorer les performances de recommandation des POIs.

Les systèmes FC hybrides se combinent avec d'autres technologies de recommandation (généralement avec des systèmes basés sur le contenu) pour faire des prédictions ou des recommandations.

Bien que les systèmes de recommandation traditionnels des FC soient quelque peu efficaces, ils présentent des faiblesses, par exemple, car ils ne tiennent pas compte de l'emplacement actuel de l'utilisateur lors de la recommandation du site suivant. Ces commandants ignorent la nature hiérarchique du comportement de mobilité humaine dans la recommandation.

8 Travaux connexes

Pour étudier les modèles actuels, dans cette section nous passerons en revue les études sur la recommandation POI, puis les recommandations de l'approche utilisant les informations séquentielles, Figure 1.3 présent les différents facteurs d'influence.

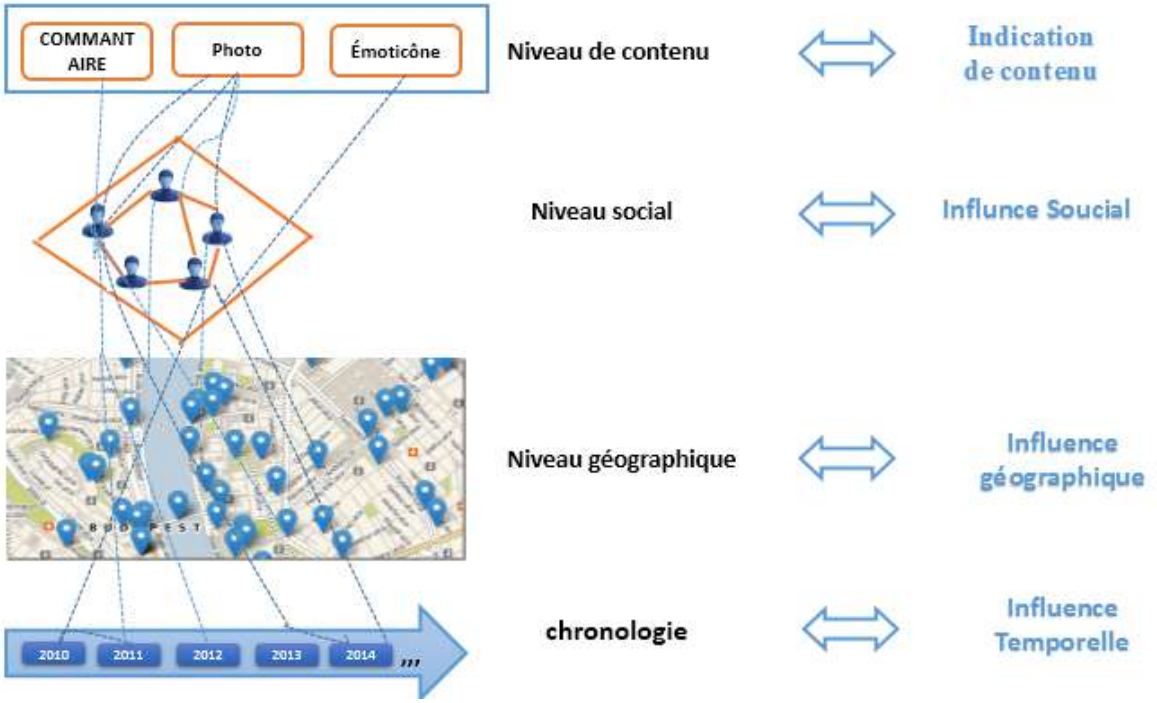


Figure 1.3: Les factures d'influence sur la recommandation dans un RSBL

8.1 Systèmes de recommandation basés sur le filtrage collaboratif

Les systèmes traditionnels de recommandation de POI, qui utilisent des techniques de filtrage collaboratif pour suggérer des POI aux utilisateurs, ont été largement étudiés. Ils sont conçus principalement pour connaître les préférences, le comportement et les intérêts des utilisateurs en monde.

Les approches appliquant le FC basé sur la mémoire prennent en compte les relations d'amitié entre les utilisateurs. Par exemple, Jie et al. [3] ont proposé un système de recommandation qui exploit tout d'abord sur les préférences personnelles de l'utilisateur tirées de son dossier d'emplacement et puis les opinions sociales extraites d'experts locaux qui peuvent partager des intérêts similaires.

Ce système de recommandations peut faciliter les déplacements des personnes, non seulement à proximité de leur lieu de vie, mais aussi vers une nouvelle ville pour eux. En utilisant les informations de catégorie pour le check-in d'un utilisateur, leur système a surmonté le problème de la divergence des données dans la matrice de positionnement d'origine.

Contrairement aux approches FC basées sur la mémoire, les approches FC basées sur un modèle sont plus évolutives et peuvent mieux faire face à des ensembles de données check-in clairsemés.

Dans cette catégorie, des techniques d'apprentissage automatique et de fouille de données sont utilisées telles que la factorisation matricielle [4]–[6] , le clustering [7] etc.

Dans [4], un system de recommandeur basé sur la factorisation matricielle pondérée a été proposé par Lian et al. Pour faire face au problème de perte des données. Ce système intègre des informations géographiques et prend en compte non seulement les facteurs latents de l'utilisateur et du POI mais aussi il faire les augmentés avec les zones d'activité des utilisateurs et les zones d'influence des POI. Cela permet donc à prendre en compte les phénomènes de clustering spatial du comportement de mobilité humaine.

Pour effectuer une recommandation contextuelle précise, Dik et al. [7] ont proposé le modèle CLR (Collaborative Location Recommandation). CLR utilise un algorithme de regroupement dynamique pour regrouper les données de trajectoire en groupes d'utilisateurs similaires, des activités similaires et des emplacements similaires en prenant en charge la mise à jour incrémentale des groupes lorsque de nouvelles données de trajectoire GPS arrivent.

8.2 Prise en compte des facteurs d'influence

8.2.1 Influence géographique

L'influence géographique est un facteur important qui distingue la recommandation des POI à la recommandation traditionnelle (Filtrage collaboratif), car le comportement check-in dépend largement des caractéristiques géographiques des lieux. Pour capturer l'influence géographique dans le POI recommandation, trois modèles représentatifs ont été proposés 1) le modèle de distribution de la loi de puissance, 2) le modèle de distribution gaussien, et 3) le modèle d'estimation de la densité du noyau. Dans [8], Ye et al ont utilisé un modèle de distribution de la loi de puissance pour considérer l'influence géographique. Le modèle de distribution de la loi de puissance a été observé dans la mobilité humaine comme les activités de retrait dans les distributeurs automatiques de billets et les déplacements dans différentes villes.

Le deuxième type de modélisation de l'influence géographique est une série de méthodes basées sur la distribution gaussienne. Cho et al. [9] ont observé que les utilisateurs dans les RSBL agissent toujours autour de certains centres d'activité.

De plus, Cheng et al. [10] ont proposé un modèle gaussien multicentrique pour capturer l'influence géographique. Étant donné l'ensemble multicentrique *Cu*, la probabilité de visiter le point d'intérêt POI_l par l'utilisateur *u* est définie par un modèle d'estimation de la densité du noyau (Kernel Density Estimation (KDE)).

Afin d'exploiter l'influence géographique personnelle, Zhang et al. [9] Il a fait valoir que l'influence géographique sur chaque utilisateur individuel devrait être spécifiquement attribuée par le biais de sa visite à un endroit spécifique plutôt que de la modélisation par une distribution commune, par exemple, la distribution de la loi d'autorité [8] et MGM [10].

8.2.2 Influence sociale

Ce facteur d'influence est basé sur l'idée que les amis partagent souvent des intérêts, des goûts et des préférences communs. Le facteur sociale a été largement pris en compte par les services de recommandation des POI [2],[11].

Ye et al. [2] ont proposé une nouvelle approche nommé FFC (Friend-based Collaborative Filtering) ou un modèle FC basé sur les amis. Le modèle FFC contraint le filtrage collaboratif basé sur l'utilisateur pour trouver les meilleurs utilisateurs similaires parmi les amis plutôt que tous les utilisateurs des RSBL.

8.2.3 Influence temporelle

Les comportements check-in des utilisateurs dans les RSBL présentent un modèle périodique non-uniforme. Cette observation inspire les recherches exploitant ce modèle périodique pour la recommandation de POI [5], [9], [12], [13].

La fonction de non-uniformité illustre la variance des préférences de check-in d'un utilisateur à différentes heures de la journée, à différents mois de l'année ou à différents jours de la semaine [12]. Les auteurs dans [12] a présenté le model TLR (Temporal Location Recommandation). TLR est conçu pour recommander des emplacements à un utilisateur en tirant parti des modèles temporels. Ils ont évalué le rendement de ses recommandations en fonction des tendances quotidiennes, tandis que sa capacité de recommandation ne se limite pas à un modèle temporel particulier.

En prenant différentes définitions de l'état temporel dans TLR, de nombreux autres modèles temporels ont être utilisés pour la recommandation TLR, à condition qu'ils contiennent les propriétés de non-uniformité. Par exemple, nous pourrions définir l'état temporel comme

t= [1, T], avec T=7 pour les modèles hebdomadaires (jour de la semaine), T=2 pour les modèles semaine/fin de semaine, et T=12 pour les modèles mensuels (mois de l'année), etc.

8.2.4 Indication du contenu

Dans un RSBL, les utilisateurs peuvent créer du contenu qui peut être une expression de sentiment personnel, une évaluation des points d'intérêt en plaçant une image sur les points importants ou autres.

Bien que le contenu n'accompagne pas chaque enregistrement check-in, le contenu disponible peut être utilisé pour améliorer la recommandation de POI [14]. Par exemple, un contenu comme les commentaires des utilisateurs sur un emplacement fournissent des informations supplémentaires sur les conseils partagés et peuvent donner une compréhension approfondie du comportement check-in des utilisateurs.

Yang et al.[14] ont utilisé des techniques d'analyse émoticône basée sur le texte pour extraire son moral , puis les convertir comme mesure de l'estimation des préférences de check-in, les conseils bruts dans les RSBL sont collectés et analysés à l'aide de techniques de traitement du langage naturel, y compris la détection de la langue, le fractionnement des phrases, traitées par la segmentation des expressions .

Ensuite, chaque commentaire reçoit un degré de sentiment. Selon le sentiment estimé, un score de préférence d'un utilisateur à un POI est généré.

8.2.5 Influence séquentielle

Après avoir étudié les techniques de filtrage collaboratif standard et leur extension considérant les influences temporelle, sociale et géographique, l'inconvénient majeur de ces études est leur ignorance de la nature séquentielle de la mobilité des humains dans la recommandation. Ces études ne tiennent pas en compte les liens séquentiels entre les points d'enregistrement.

En fait, l'influence séquentielle consiste à considérer les relations séquentielles cachées dans le comportement de mobilité des utilisateurs pour faire des recommandations, Autrement dit, nous devons baser sur l'emplacement actuel de l'utilisateur et les mouvements historiques afin de prédire l'ensemble de POI que l'utilisateur peut être intéressé à visiter dans l'avenir.

Dans la littérature, de nombreuses études récentes ont considéré l'effet séquentiel pour l'amélioration de la recommandation. En particulier, deux modèles représentatifs ont été

proposés notamment les chaines de Markov [14]-[15] et la fouille de données des motifs séquentiels [16]-[18].

8.2.5.1 Le modèle de Markov

Le modèle de Markov est un modèle qui considère une séquence de mobilité de taille *k* ou *k* est l'ordre de chaine de Markov .Cette séquence contient l'emplacement actuelle d'un utilisateur pour recommander de nouveaux endroit (l'emplacement future dépend de son état précèdent), par exemple, Zhang et al. [19] ont conçu un système basé sur les chaines de Markov qui tient compte de l'influence séquentielle, géographique et sociale pour la recommandation de POIs

Le modèle de Markov à l'avantage d'être un modèle simple et facile de mettre en œuvre, cependant une limitation majeure de ce modèle est le paramètre *k* qui doit être défini.

De plus, la prise en compte de plusieurs emplacements pour effectuer des prédictions peut améliorer les performances en découvrant les emplacements visités par plusieurs utilisateurs partageant un comportement de mobilité commun. Toutefois, cela peut également augmenter considérablement le temps requis pour effectuer la recommandation.

8.2.5.2 La fouille de données des motifs séquentiels

Ces modèles visent à extraire les motifs séquentiels dans l'historique de mobilité des utilisateurs (Un motif séquentiel est dit fréquent, s'il apparaît dans plus de *minfreq* fois dans les données d'emplacement, où *minfreq* est un seuil défini par l'utilisateur).

Ces motifs extraits sont utilisées par la suite pour prédire les POIs qui peuvent intéresser l'utilisateur à être visiter dans l'avenir

Dans la littérature, les chercheurs dans [20] ont proposé un system de recommandation qui extrait les motifs de check-in séquentiels avec des intervalles temporels entre les POIs. Une autre approche, nommée LORE a été proposée dans [21], LORE utilise l'extraction des motifs fréquents et les chaînes de Markov pour la recommandation à la fois.

Malgré les systèmes de recommandation basés sur la fouille des motifs séquentiels ont montré leur efficacité pour la recommandation ; ces systèmes ont une limitation majeure qui exige que les emplacements dans les motifs soient ordonnés. Pour cela, une petite variation de l'ordre des localisations dans une séquence affecte la recommandation et peut conduire à des recommandations complétement différentes.

Tableau 1.2 présente une brève comparaison sur les modèles de recommandation qui ont été examinés dans ce chapitre. Les modèles sont comparé selon trois principaux critères : (1) le type de modèle, (2) le facteur d'influence considéré, et (3) le type de technique utilisée.

Auteur	Modèle	Technique utilisé	Facteur d'influence
Cheng et al. [15] (2013)	Filtrage collaboratif(FC)	Un modèle gaussien multicentrique (MGM)	Géographique
Ye et al. [8] (2011)	Filtrage collaboratif(FC)	Modèle de distribution de la loi de puissance	Géographique
Zhang et al. [9] (2010)	Filtrage collaboratif(FC)	Modèle d'estimation de la densité du noyau (KDE).	Géographique
Ye et al. [2] (2010)	Filtrage collaboratif(FC)	Modèle basé sur la mémoire	Social
Wang et al. [10] (2016)	Filtrage collaboratif(FC)	Une approche basée sur les amis	Social
Eunjoon et al. [7] (2007)	Filtrage collaboratif(FC)	Modèle périodique	Temporelle
Yang et al.[14] (2013)	Filtrage collaboratif(FC)	Analyse émoticône basée sur le texte	Indication du contenu
Zhang et al. [19] (2012)	Chaines de Markov	Chaines de Markov	Séquentiel

Tableau 1. 2 **:** Comparaison des modèles de recommandations de POI.

Conclusion

Dans ce chapitre, on a présenté les concepts et les notions de bases liés à la recommandation de POI dans les réseaux sociaux basés sur la localisation. On a abordé aussi les facteurs d'influences qui peuvent affecter la recommandation .On a présenté en bref les modèle et les travaux les plus importants le domaine de recommandation des POIs.

Chapitre 2

La fouille de données (Data mining)

Chapitre 2

1. Introduction

Durant les dernières années, une croissance importante des moyens de génération et de collection des données a été remarquée. Ceci est principalement dû à l'évolution de la technologie des supports de stockage. Du fait de l'informatisation rapide des entreprises, des administrations, du commerce, des télécommunications, la quantité de données disponibles augmente très rapidement. Cependant, l'analyse et l'exploitation de ces données restent très difficiles. Cela créé un besoin d'acquisition de nouvelles techniques et méthodes intelligentes de gestion qui permettent d'extraire des données, des informations utiles appelées connaissances. C'est ainsi qu'on a commencé à parler de la découverte de connaissances à partir de données Knolwedge data Discovery (KDD) ou encore de ***Data Mining ou de fouille de données***.

Dans ce chapitre, on va faire un tour d'horizon sur le concept du fouille de données (Data mining), son historique, ses définitions de bases, ainsi que des tâches et techniques. Parmi les techniques citées dans ce chapitre, la technique d'extraire des règles d'association est la plus adaptée à notre étude de cas qui consiste à extraire les règles séquentielles qui considère l'influence séquentielle dans la recommandation de POIs.

2. Historique

Au début des années 1960, le terme la fouille de données est apparue et avait une signification insultante à cette époque.

La fouille de données travaille sur des méthodes d'identification des données, en plus de la force croissante des nouvelles technologies, ce qui contribuait grandement à augmenter les ensembles de données, la manipulation et la capacité de stockage.

La fouille de donnés est un processus d'application de méthodes visant à découvrir des tendances cachées. Au fil des années, plusieurs méthodes et techniques ont émergé comme les réseaux de neurones de Mac Culloch et Pitts en 1941 [29], et les arbres de décision en 1943 [30]. A partir de 1984 ces technologies ont été améliorées pour qu'elles puissent exploiter et découvrir des modèles de plus en plus précis.

De nos jour, la fouille de donnés se présente comme un outil essentiel dans les processus décisionnels. Elle combine un ensemble de techniques statistiques qui doivent être utilisées en fonction des problèmes descriptifs ou de la prise de décision.

3. Définition de la fouille de donnée (Data mining)

La fouille de donnée est un ensemble des techniques et méthodes destinées à l'exploration et l'analyse de grandes bases de données informatiques en vue de détecter dans ces données des règles, des associations, des structures pour en extraire l'essentiel de l'information utile dont l'objectif est l'aide à la décision [24]. La fouille de donnée a d'autres définitions, nous citons quelques-unes :

1- La fouille de donnée ou data mining est l'analyse de grands ensembles de données observationnelles pour découvrir des nouvelles relations entre elles et de les reformuler afin de les rendre plus utilisables de la part de leurs propriétaires [22].

2- La fouille de donnée est l'extraction de connaissances à partir de grandes quantités de données. C'est un domaine relativement récent qui se situe à l'intersection des statistiques, de l'apprentissage automatique et des bases de données [23].

En bref, la fouille de donnée est l'art d'extraire des informations (ou même des connaissances) à partir des données.

4. Les étapes du processus de la fouille de données

Figure 2.1 récapitule les différentes phases de l'extraction de connaissance ainsi que les enchaînements possibles entre elles. Ce processus comprend des étapes de 1) définition du problème (définition du domaine, but de l'utilisateur final), 2) préparation des données (sélection, préparation, transformation), 3) fouille de données (sélection des outils de data mining appropriés, recherche des patrons ou motif) et 4) évaluation des résultats pour aboutir aux nouvelles connaissances.

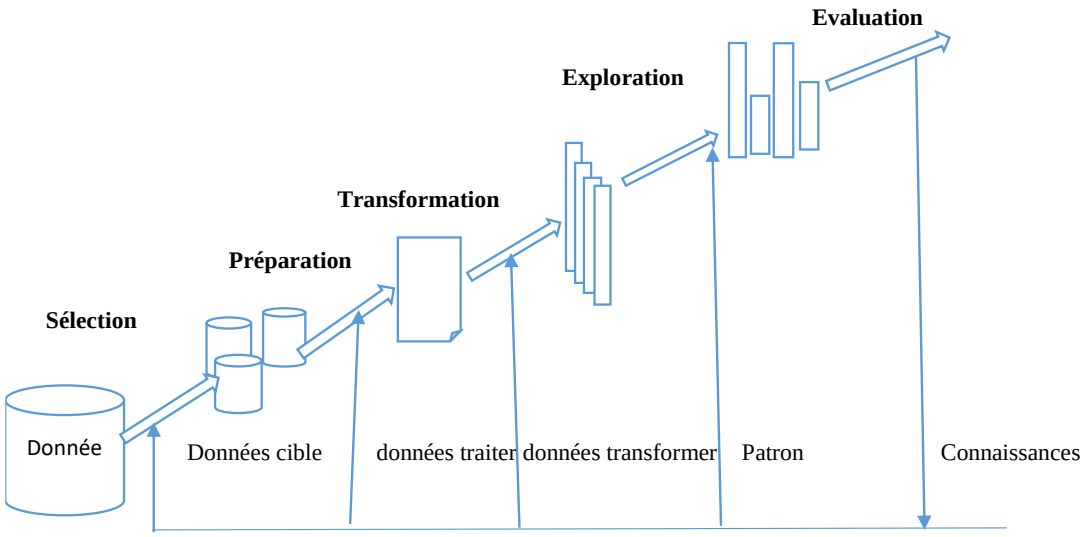


Figure 2. 1 : Le processus de la fouille de données.

5. Principales tâches de la fouille de données

Les tâches de la fouille de données peuvent être largement classées en deux types en fonction de ce qu'une tâche spécifique tente d'accomplir. Ces deux catégories sont les tâches descriptives et les tâches prédictives.

Les tâches de fouille de données descriptives caractérisent les propriétés générales des données, tandis que les tâches d'exploration de données prédictives font une inférence sur l'ensemble de données disponible pour prédire le comportement d'un nouvel ensemble de données.

Chapitre 2

Il existe plusieurs tâches de la fouille de données telles que la classification, la prédiction, l'estimation, l'association, la prévision, la segmentation et la description. Toutes ces tâches sont soit des tâches de fouille de données prédictives ou des tâches d'exploration de données descriptives [31].

5.1 Les tâches de fouille de données prédictives

5.1.1 La prédiction

Cette tâche est une tâche de prédire les valeurs potentielles de données manquantes ou futures prédit. Une prédiction implique l'élaboration d'un modèle basé sur les données disponibles et ce modèle est utilisé pour prévoir les valeurs futures pour un nouvel ensemble de données d'importance. Par exemple, prévoir localisation suivant d'un utilisateur.

5.1.2 La classification

La classification est la tâche la plus commune du la fouille de donnée. La classification dérive un modèle pour déterminer la classe d'un objet en fonction de ses attributs. Dans la tâche de classification, une collection d'enregistrements sera disponible ou chaque enregistrement est associé avec un ensemble d'attributs.

L'un de ces attributs sera l'attribut de classe. L'objectif de la tâche de classification sera donc à associée un attribut de classe à un nouvel ensemble d'enregistrements aussi précisément que possible dont l'objectif final est de trouver un modèle dérivé qui décrit et distingue les classes de données.

5.1.3 L'estimation

Bien que la classification traite des résultats discrets tels que *Oui* ou *Non*, l'estimation traite des résultats évalués en continu. Si certaines données d'entrée sont disponibles, une estimation peut être utilisée pour trouver une variable continue inconnue comme le revenu ou la taille. Avec la tâche d'estimation, on veut trouver une valeur plausible ou une plage de valeurs plausibles pour les paramètres inconnus d'un système.

5.2 Tâches d'exploration de données descriptives

5.2.1 Segmentation ou clustering

La segmentation ou le clustering est le regroupement d'enregistrements ou des observations en classes d'objets similaires. Un cluster est une collection d'enregistrements similaires l'un à l'autre, et différents à ceux existants sur les autres clusters. La différence entre le clustering et

la classification est que dans le clustering il n'y a pas de variables sortantes. La tâche de clustering ne classifie pas, n'estime pas, ne prévoit pas la valeur d'une variable sortantes.

5.2.2 La description

Le but du la fouille de donnée est simplement de décrire ce qui se passe sur une base de données compliquée en expliquant les relations existantes dans les données pour comprendre le mieux les items, les processus présents sur cette base. Une bonne description d'un comportement implique souvent une bonne explication de celui-ci. Les techniques convenables à la description sont les règles d'association et les arbres de décisions

5.2.3 L'association

La recherche de règles d'association est la tâche la plus intéressante du la fouille de données. C'est également celle qui est la plus répandue dans le monde des affaires, notamment en marketing pour l'analyse du panier de ménagé.

La recherche de règles d'association cherche à découvrir les règles de quantification ou de relation entre deux ou plusieurs attributs. Les règles d'association sont de la forme «Si antécédent, puis conséquente », avec une mesure confiance associée à la règle. La recherche de règles d'associations dans une grande base de données permet de découvrir des règles cachées utiles pour la prise de décision [32].

6 Techniques du la fouille de données

Pour effectuer les tâches du fouille de données il existe plusieurs techniques issus afin de faire apparaître des corrélations cachées dans des gisements de données pour construire des modèles à partir de ces données et ça en fonction des besoins de l'utilisateur (selon les tâches à effectuer). Dans ce chapitre, nous allons présenter les techniques du la fouille de données les plus connues. Chacune de ces techniques regroupe une multitude d'algorithmes pour construire le modèle auquel elle est associée.

6.1 Les techniques supervisées

Dans la classification supervisée (Appelée aussi prédictive), le but est d'utiliser les données disponibles pour créer un modèle décrivant une variable particulière prise comme but en termes de ces données.

6.1.1 Les arbres de décision

Les arbres de décision est un outil puissant utilisé beaucoup plus pour la classification que pour la prédiction. Ces arbres permettent de distinguer les différentes classes et de leur associer à une ou plusieurs règles.

Les arbres de décision sont des outils d'aide à la décision qui permettent selon des variables discriminantes de répartir une population d'individus en groupes homogènes en fonction d'un objectif connu. Les arbres de décisions permettent à partir des données connues sur le problème de donner des prédictions par réduction, niveau par niveau, du domaine des solutions. Chaque nœud interne d'un arbre de décision permet de répartir les éléments à classifier de façon homogène entre ses différents fils en portant sur une variable discriminante de ces éléments. Les branches qui représentent les liaisons entre un nœud et ses fils sont les valeurs discriminantes. Dans la littérature, plusieurs algorithmes d'induction des arbres de décision ont été proposés telle que l'algorithme CHAID développé par KASS et ID3 [27], C4.5 développé par QUINLAN [28].

A titre d'illustration, Figure 2.2, présent un exemple d'un arbre de décision qui détermine si on va jouer au tennis ou non. En commençant par le nœud racine, si les perspectives sont nuageuses alors nous devrions certainement jouer au tennis. S'il pleut, on ne devrait pas jouer tennis que si le vent est élevé. Et s'il fait beau alors on devrait jouer au tennis au cas où l'humidité serait normale.

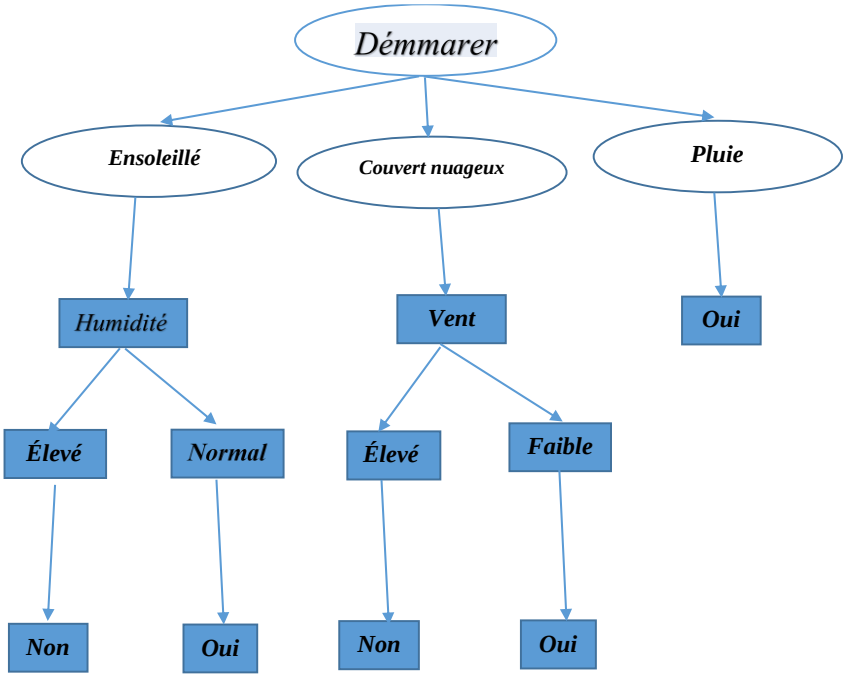


Figure 2. 2*:* Les arbres de décisions.

6.1.2 Les réseaux de neurones

Un réseau de neurones est un modèle de calcul dont le fonctionnement vise à simuler le fonctionnement des neurones biologiques. Un réseau neuronal est l'association, en un graphe plus ou moins complexe, d'objets élémentaires, les neurones formels. Les principaux réseaux se distinguent par 1) l'organisation du graphe (en couches, complets, etc.), c'est-à-dire leur architecture, 2) son niveau de complexité (le nombre de neurones, présence ou non de boucles de rétroaction dans le réseau), et 3) par le type des neurones (leurs fonctions de transition ou d'activation), Figure 2.3.

Il existe deux types de réseaux :1) Réseaux à apprentissage supervisé où la réponse est connue à l'avance et 2) Réseaux à apprentissage non supervisé où le résultat n'est pas connu à l'avance.

Ces outils sont généralement utilisés pour la classification, l'estimation, la prédiction et la segmentation. Ceux-ci obtiennent de bonnes performances, en particulier, pour la reconnaissance de formes. Donc, ils sont bien adaptés pour des problèmes comprenant des variables continues éventuellement bruitées. Le principal inconvénient est qu'un réseau est défini par une architecture et un grand ensemble de paramètres réels (les coefficients synaptiques) ainsi qu'un, faible pouvoir explicatif [33].

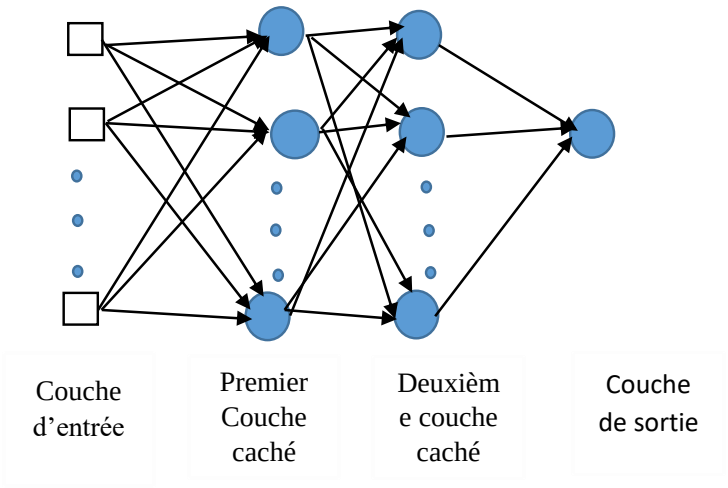


Figure 2. 3: Les réseaux de neurones.

6.2 Les techniques non supervisées

Dans les techniques non supervisées (Appelées aussi descriptives), ces modèles ne précisent pas de valeur pour la cible mais se concentrent sur des relations entre données.

L'analyse peut être également effectuée de manière exploratoire. Le but est donc de regrouper dans un même groupe (ou cluster) les objets considérés comme similaires, pour constituer les classes.

6.2.1 Clustering

Le clustering consiste à segmenter un groupe diversifié en un certain nombre de sous-groupes ou clusters. Les groupes d'objets sont formées de sorte que les objets au sein d'un groupe présentent une grande similitude les uns par rapport aux autres, mais sont très différents des objets d'autres groupes. Le clustering est couramment utilisé pour rechercher des regroupements uniques dans un ensemble de données.

Le facteur de distinction entre le clustering et la classification est que dans le clustering, il n'y a pas de classes prédéfinies et pas d'exemples. Les objets sont regroupés en fonction de l'auto-similitude.

On distingue trois grandes familles de clustering notamment : 1) Clustering par partition, 2) Clustering hiérarchiques, et 3) Clustering par densité.

6.2.2 Les règles d'association

Les règles d'association sont une des méthodes de fouille de données les plus répandus, Elles sont de la forme "Si $action_A$ ou condition alors $action_B$". Cela signifie que l'occurrence d'un acte est le résultat d'une autre action ou condition (par exemple $A \rightarrow B$). Elles peuvent aussi se situer dans le temps : "Si $action_A$ ou condition à l'instant t_1 alors $action_B$ à l'instant t_2», (ex : $< t_1 > A \rightarrow < t_2 > B$), c'est les règles d'association séquentielles.

Ces règles sont intuitivement faciles à interpréter car elles montrent comment des éléments se situent les uns par rapport aux autres. Le but principal de cette technique est donc descriptif. Dans la mesure où les résultats peuvent être situés dans le temps, cette technique peut être considérée comme prédictive.

Cependant, il faut noter que cette méthode, si elle peut produire des règles intéressantes, elle peut aussi produire des règles triviales ou inutiles. La recherche de règles d'association est une méthode non supervisée car on ne dispose en entrée que de la description. Puisque la technique des règles d'association sera dans notre travail dans ce mémoire, une description détaillée des notions de bases de cette technique est présentée dans ce qui suit [26].

6.2.2.1 Préliminaires

Définition 1 (Item). Un item est un objet, élément ou un article d'une base de données, qu'ils doivent extraire pour construire la séquence [39].

Définition 2 (Séquence). Une séquence est un ensemble des items lies entre eux comme les séquences d'ADN, séquence de localisation, etc.…, Une séquence des éléments $S = I_i, I_{i+1}, .., I_m$ est une liste ordonnée des éléments pendant une période de temps T [39].

Définition 3 (Sous-séquence). Une séquence $S_1 = I_{x1}, I_{x2},\ldots, I_{xn}$ est dite être une sous-séquence d'une séquence $S_2 = I_{y1}, I_{y2},\ldots, I_{ym}$ (notée $S_1 \subseteq S_2$), si et seulement s'il existe des entiers $1 \leq a_1 < a_2 < \ldots < a_n \leq m$ tels que $I_{x1} = I_{ya1}, I_{x2} = I_{ya2}, I_{xn} = I_{yan}$ [39].

Définition 4 (Base de données de séquence). Une base de données des séquences SD est un ensemble de séquences $S = \{S_1, S_2,\ldots, S_m\}$ d'un ensemble d'items $I = \{I_1, I_2,\ldots, I_n\}$ apparaissant dans ces séquences, où chaque séquence se voit attribuer un SID unique (ID de séquence). Une séquence est une liste ordonnée d'items (ensembles d'éléments) $S_x = I_1, I_2,\ldots I_p$ telle que $I_1, I_2,\ldots I_p \subseteq I$. Par exemple, Tableau 2.1 décrit une base de

données de séquences contenant quatre séquences ayant respectivement les sids seq_1, seq_2, seq_3 et seq_4 [39].

Dans cet exemple, chaque lettre représente un item. Les lettres entre crochets représentent un ensemble des items. Par exemple, la séquence seq_1 signifie que les items I_1 et I_2 se sont produits en même temps et ont été successivement suivis de I_3, I_4, I_5 et I_6.

SID	***Séquences***
seq_1	$\{I_1, I_2\}, \{I_3\}, \{I_4\}, \{I_5\}, \{I_6\}$
seq_2	$\{I_1, I_4\}, \{I_3\}, \{I_2\}, \{I_1, I_2, I_6, I_4\}$
seq_3	$\{I_1\}, \{I_2\}, \{I_4\}, \{I_6\}$
seq_4	$\{I_2\}, \{I_4, I_5, I_7\}$

Tableau 2. 1*:* Quelque séquence de base de données.

6.2.2.2 Extraction des motifs fréquents

Définition 5 (Un motif fréquent). Un motif (ou itemset) est un sous ensemble d'attributs. Le support d'un motif est la proportion d'individus associés à ce sous ensemble de motif. Un motif est fréquent si son support est supérieur à un seuil minimal fixé *minsup*. Formellement cela se définit comme suit soient $I = \{I_1,..., I_m\}$ un ensemble de m items, et $B = \{seq_1,..., seq_n\}$ une base de données de n séquences. Chaque séquence est composée d'un sous-ensemble d'items $I' \in I$. Le sous-ensemble I' de taille k est appelé un *k-motif*. Un motif est dite *fréquent* si son support (nombre de fois qu'il apparait dans la base de transition) est supérieur à un seuil minimale fixé *minsup* [39].

6.2.2.3 La génération des motifs fréquents

Plusieurs algorithmes traitent le problème de la recherche des motifs par exemple *Apriori* [34], *AprioriTID* [25].

L'algorithme *Apriori* est un algorithme itératif de recherche des itemsets fréquents par niveaux. Pour chaque k-itération, 1-l'algorithme génère un ensemble d'itemsets candidats de taille k, puis scanne de la base de transactions pour supprimer les candidats non fréquents.

L'ensemble des *k-itemsets* fréquents générés est utilisé à l'itération $k + 1$ pour générer les candidats de taille $k + 1$.Apriori base principalement sur l'hypothèse que si un itemset de longueur k est non fréquent alors tous ses sur-ensembles (super-set) le sont également.

Il est important de noter ici que l'extraction des motifs fréquents en fouilles de données génère une quantité énorme de motif et requiert par conséquent la mise en place d'un post-traitement efficace afin de cible les plus utiles.

6.2.2.4 Génération des règles d'association

La génération des règles d'association s'effectue à partir des itemsets fréquents générés précédemment. En général, la génération des règles d'association est réalisée de manière directe, sans accéder au contexte d'extraction, et le coût de cette phase en temps d'exécution est donc faible par rapport au coût de l'extraction des itemsets fréquents.

7 . La fouille de données des règles séquentielles

La fouille de données des règles séquentielles est une technique de fouille de données bien connue pour découvrir des corrélations entre des événements dans deux motifs dans l'antécédents et le conséquent d'une règle selon une relation d'ordre.

En générale, la fouille de donnée des règles séquentielle vise principalement à extraire les motifs fréquents séquentiels puis utilisé ces motifs pour générer des règles d'association.

7.2 Exploration des motifs séquentiels (Sequential patterns mining)

L'exploration de motifs séquentiels (*fréquents*) consiste à trouver toutes les sous-séquences fréquentes comme modèles séquentiels dans une base de données de séquences. Elle est, pour objectif de découvrir des motifs inter transactions comme par exemple un ensemble d'items suivi par un autre item dans un ensemble ordonné de transactions. Formellement, un motif *p* est dit *fréquent* si son support (support(p)) est supérieur de seuil de support minimum (*minSup*).

Définition (Motif séquentiel et motif séquentiel fréquent). Un motif séquentiel est une séquence qui est une sous-séquence d'une ou de plusieurs séquences d'une base de données de séquences BDS. Formellement, une séquence $S_a = (A_1, A_2, \ldots A_e)$ est dite être une sous-séquence d'une séquence $S_b = (B_1, B_2, \ldots B_f)$ si et seulement s'il existe des entiers $1 \leq x_1 < x_2 \ldots < x_e$ tels que $A_1 \subseteq B_{x1}, A2 \subseteq B_{x2}, \ldots A_e \subseteq B_f$. Par exemple, considérons la base de données de Tableau 2.1 comme BDS (la base de données de séquences). La séquence ($\{I_2\}$, $\{I_5\}$) est un motif séquentiel se produisant dans S_1, S_2, S_3 et S_4. Un autre exemple est ($\{I_2\}$, $\{I_5\}$, $\{I_4\}$). Il s'agit d'un motif séquentiel se produisant dans les séquences S_1 et S_3. Une règle séquentielle standard $S_a \rightarrow S_b$ une relation séquentielle entre deux motifs S_a et S_b. Un motif séquentiel *P* est dit fréquent si son support *(support(p))* est supérieur de seuil de support minimum *(minSup)* [39].

Définition (Règle séquentielle). Une règle séquentielle R=X→Y est une relation entre deux motifs séquentiels tels que X et Y ne sont pas ensemble vide et intersection de X et Y est un ensemble vide. {X, Y} != Ø et {X∩Y} =Ø.

7.2.2 Le support d'une règle séquentielle

Le support d'une règle séquentielle R : X→Y s'exprime par le nombre de transactions qui contiennent les items de X et les items de Y (sids (X→Y)) divisé par le nombre total des transactions de la base des transactions (|S|). Le support d'une règle d'association est défini comme suit :

$$Sup\ (X \rightarrow Y) = |sids(X \rightarrow Y)| / |S|.$$

7.2.3 La confiance d'une règle séquentielle

La confiance d'une règle séquentielle s'exprime par le nombre de transactions qui contiennent la relation d'union entre la transaction X et la transaction Y divisé par le nombre des transactions qui contiennent la transaction X. Pour une règle R : X→Y, la confiance d'une règle d'association est définie comme suit :

$$Conf(X \rightarrow Y) = |sids(X \cup Y)| / |sids(X)|.$$

Ou X ∪Y représente l'ensemble union contenant les éléments de la transaction X et les éléments de la transaction Y.

Définition (Une règle séquentielle valide). Soit R : X→Y une règle séquentielle. R est dite valide si R est une règle fréquent (son support *Sup*(R)) est supérieure de seuil de support minimum *(minsup)* (*Sup*(R)>*minsup)* et sa confiance est supérieure de seuil de confiance minimum *(minConf) (Conf(R)> minConf).*

Par exemple, Tableau 2.2 montre quelques règles valides trouvées dans la base de séquences du Tableau 1 pour *minsup* = 0,5 et *minconf* = 0,5. Dans cette exemple, la règle $\{I_1, I_2, I_3\} \Rightarrow \{I_4\}$ a un support de | sids (X→Y) | / | S | = 2/4 = 0,5, et une confiance de | sids (X⇒Y) | / | sids (X)| = 2/2 = 1. Étant donné que ces valeurs ne sont respectivement pas inférieures à *minsup* et *minconf*, la règle est considérée comme valide.

ID	Règle	Support	Confiance
R_1	$\{I_1, I_2, I_3\} \rightarrow \{I_4\}$	0.50	1.00
R_2	$\{I_1\} \rightarrow \{I_3, I_4, I_5\}$	0.50	0.66
R_3	$\{I_1, I_2\} \rightarrow \{I_4, I_5\}$	0.75	1.00
R_4	$\{I_2\} \rightarrow \{I_4, I_5\}$	0.75	0.75
R_5	$\{I_1\} \rightarrow \{I_4, I_5\}$	0.75	1.00

Tableau 2. 2: Quelques règles séquentielles.

8 Conclusion

La fouille de données est le résultat de la combinaison de nombreux facteurs technologiques et économiques. Le développement des capacités de stockage et les vitesses de transmission des réseaux ont conduit les utilisateurs à accumuler de plus en plus de données.

La fouille de données répond au besoin d'exploitation pour bénéficier de ces données collectées. D'une façon générale, la fouille de données est l'art d'extraire des connaissances à partir de données qui peuvent être stockées dans base de données qui est distribuées ou sur Internet. Ce chapitre a fait un tour d'horizon sur le concept et ces techniques. Nous avons concentrée le plus sur la fouille de données de règles d'association utilisé dans notre études pour la prédiction de localisations dans les RSBLs.

Chapitre 3

Le système STS-Rec basé sur la fouille incrémentale des règles séquentielles.

Chapitre 3

1. Introduction

Ce chapitre présent notre système, appelé STS-Rec, pour de recommandation des POI basée sur l'exploration des règles séquentielles. STS-Rec prend en compte non seulement le comportement séquentiel de la mobilité humaine mais aussi l'évolution système en explorant progressivement des règles de recommandation séquentielles tout en préservant la validité de système. Cette extraction incrémentale des règles de recommandation est principalement proposée pour gérer des données dynamiques où de nouveaux utilisateurs peuvent à tout moment rejoindre le RSBL.

L'approche incrémentale de STS-Rec adopte un motif basé sur une structure d'arbre qui utilise aussi une structure bitmap pour stocker les motifs de mobilité. Par la suite, des structures sont utilisées avec d'autre structure de données pour mettre à jour efficacement l'ensemble des règles de recommandation (en évitant d'effectuer des calculs redondants).

De plus et contrairement à la recommandation séquentielle basée sur la fouille de données séquentielle, STS-Rec découvre efficacement des motifs séquentiels indiquant l'évolution du comportement périodique de la mobilité humaine en relâchant la contrainte stricte sur les l'ordre des items dans les séquences de localisation en découvrant les POSR (ou Partially Ordred Sequential Rules ou règles séquentielles partiellement ordonnées).

En outre, et en utilisant d'une fenêtre glissante (contrainte de fenêtre), STS-Rec peut effectuer des recommandations à court terme pour suggérer où les gens iront ensuite plutôt que plus tard. Par conséquent, cette contrainte extrait les motifs fréquents de localisation qui apparaît dans un nombre maximal de localisations consécutives dans les séquences de localisations.

Dans ce chapitre, nous allons présenter le système STS-Rec en commençant par la motivation suivi par l'explique d'exploration des règles de recommandation, en plus la description de l'architecture de système en détaillant ses différents composants.

2. Motivations

L'inconvénient majeur de la recommandation en utilisant des règles séquentielles standard (RSS) est que les règles générées sont trop précise. L'ordre imposé dans les emplacements est trop strict où chaque emplacement doit apparaître exactement dans le même ordre dans une séquence d'emplacement pour qu'une règle corresponde à cette séquence. En d'autres termes, la recommandation utilisant RSS exige que l'utilisateur doive visiter l'ensemble des emplacements dans exactement le même ordre que celui des emplacements dans la règle.

Par conséquence, la recommandation échoue si l'utilisateur visite des emplacements dans un ordre légèrement différent de celui dans les règles de recommandation par exemple, les touristes ont généralement tendance à visiter les mêmes lieux touristiques célèbres d'une ville, mais l'ordre séquentiel peut être légèrement différent selon les touristes.

En outre l'ordre strict dans les règles séquentielles standard RSS peut être également affecté considérablement la façon dont la fréquence des schémas de localisation est calculée. Pour déterminer si un motif est fréquent, l'ordre dans une séquence doit être exactement apparié. Tout changement dans l'ordre entraîne un nouveau motif et, par conséquent, deux motifs qui se composent du même ensemble d'emplacements sont considérés comme deux motifs différents et utilisés d'une manière différente. Par conséquent, le nombre de motifs fréquents peut diminuer, ce qui peut nuire à la capacité de formuler des recommandations (c.-à-d. la portée des recommandations).

Et pour tenir compte des limites susmentionnées des motifs séquentiels, nous présentons, dans les sections suivantes un système de recommandation basé sur des règles séquentielles, appelé « *STS-Rec* ». Ce dernier traite les principaux inconvénients des approches d'exploration de motifs séquentiels et a la capacité de tenir compte de l'influence séquentielle pour effectuer la recommandation des POIs. STS-Rec découvre efficacement l'évolution du comportement séquentiel périodique de la mobilité humaine en relâchant la contrainte d'ordre stricte sur les séquences de localisation en découvrant POSR (Règles séquentielles partiellement ordonnées). Ce type de règles permet une relation d'ordre entre l'antécédent et le conséquent d'une règle (l'antécédent doit apparaître avant le conséquent) tout en relâchant la contrainte d'ordre entre les emplacements dans chaque partie de la règle.

De plus, STS-Rec étend le concept de motifs séquentiels avec l'utilisation d'une fenêtre coulissante (une contrainte fenêtre).

Cette contrainte permet d'extraire des règles de recommandation pour des emplacements qui apparaissent dans un nombre maximum d'emplacements consécutifs dans des séquences d'emplacements. Grâce à cette contraint, la relation séquentielle entre les emplacements est considérée pour éviter de recommander des emplacements éloignés.

Un autre avantage d'utiliser POSR au lieu de RSS est que la fréquence des motifs d'emplacement est généralement augmentée puisque de nombreux motifs contenant le même ensemble d'emplacements sont considérés comme le même motif. Ainsi, chaque motif peut être considéré comme apparaissant plus souvent que le RSS correspondant, ce qui permet de générer plus de règles et peut augmenter la couverture du système (sa capacité à faire une recommandation).

3. Exploration des règles de recommandation

L'idée centrale de *STS-Rec* est de générer des règles séquentielles à partir de l'ensemble des séquences d'emplacement, extraites à partir des données de check-in des utilisateurs du RSBL, puis d'utiliser ces règles pour la recommandation. Dans ce mémoire, nous appelons le processus d'exploration de règles séquentielles comme le processus d'exploration des règles de recommandation.

Dans cette section, nous fournissons une description détaillée sur ce processus. Plus précisément, nous décrivons notre processus d'exploration des règles de recommandation des règles *POSR*. Ensuite, nous détaillons notre nouvelle approche *IncPOSR* pour la fouille incrémentale des règles basée sur les arbres.

3.1. Extraction des règles de recommandation partiellement ordonnées

Pour extraire les règles de recommandation partialement ordonnées, l'algorithme *TRuleGrowth* [35] est utilisé. *TRuleGrowth* prend en entrée l'ensemble des séquences d'emplacement dans un historique d'emplacement LH, les seuils *minconf* et *minfreq* et donne en sortie l'ensemble des règles séquentielles partiellement ordonnées POSR.

> **Définition 1 (Historique des localisations).** L'historique de localisation ou emplacement (*LH*) d'un utilisateur *Ui* est l'ensemble de toutes les séquences de localisation de *Ui*, comme ce que l'on remarque dans Tableau 3.1.

Utilisateur	Jour	Check-in	Historique d'emplacement	Séquences d'emplacements
u_1	J_1	<u_1,8:15,E_6><u_1,10:03,E_5><u_1,10:56,E_7><u_1,14:34,E_9>	LH_{u1}	S_{11}: E_6,E_5,E_7,E_9
	J_2	<u_1,06:14,E_1><u_1,7:19,E_2><u_1,10:07,E_3><u_1,11:38,E_4><u_1,16:09,E_7><u_1,20:02,E_{12}><u_1,21:45,E_{10}>		S_{12}:E_1,E_2,E_3,E_4,E_7, E_{12},E_{10}
	J_3	<u_1,8:34,E_3><u_1,11:23 E_4><u_1,22:59, E_7>		S_{13}: E_3,E_4,E_7
u_2	J_1	<u_2, 5:25 E_1><u_2, 6:57 E_2><u_2, 09:50, E_9><u_2,11:36,E_3><u_2,12:02E_4><u_2,14:47, E_7><u_2,17:38, E_6><u_2,18:03, E_{10}>	LH_{u2}	S_{21}:E_1,E_2,E_9,E_3,E_4,E_7,E_6,$_{10}$
	J_2	<u_2,09:37,E_4><u_2,11:09,E_3><u_2,15:55,E_7><u_2,17:45, E_9>		S_{22}: E_4,E_3,E_7, E_9
	J_3	<u_2,10:39, E_3><u_2,11:58, E_4><u_2,23:02, E_{12}>		S_{23}: E_3,E_4,E_{12}
u_3	J_1	<u_3,4:15,E_5><u_3,07:03,E_6><u_3,12:53, E_{11}><u_3,16:03, E_7>	LH_{u3}	S_{31} : E_5,E_6,E_{11},E_7 S_{32} : E_5,E_6,E_3,E_4,E_1,E_8,E_{12}
	J_2	<u_3,5:18,E_5><u_3,07:03,E_6><u_3,10:58,E_3><u_3,11:39, E_4><u_3,20:20,E_1><u_3,21:35, E_8><u_3,23:28, E_{12}>		

Tableau 3. 1*:* Exemple de séquences d'emplacement et d'histoires.

Définition 2 (Motif d'emplacement). Un motif d'emplacement P_i est une sous-séquence de plusieurs séquences d'emplacement dans un historique d'emplacement *LH.*

Définition 3 (Fréquence d'un motif d'emplacement). La fréquence d'un motif P_i, notée *Fréquence* (P_i) indique le nombre de fois qu'un utilisateur a visité les emplacements de P_i dans *LH* sur le nombre total de séquences dans *LH*, P_i est considéré *fréquent* si sa *fréquence* dépasse un paramètre défini par l'utilisateur appelé *minfreq* (fréquence minimale).

Définition 4 (Règle séquentielle partiellement ordonnée POSR). Une règle de recommandation partiellement ordonnée $R : P_1 \rightarrow P_2$ est une règle séquentielle qui représente une relation entre deux motifs d'emplacement non ordonnés $P_i = E_x, E_{x+1}, \ldots E_n$, $P_j = E_y, E_{y+1}, \ldots, E_m$. La règle *R* est interprétée comme si un utilisateur avait visité les emplacements dans P_1 dans n'importe quel ordre, il visiterait alors ceux dans P_2 [39].

Le processus d'exploration de données de POSR, en utilisant l'algorithme *TRuleGrowth,* consiste à générer l'ensemble de règles valides de taille 1 * 1 (un item ou emplacement dans

l'antécédent et le conséquent de la règle). Ensuite, *TRuleGrowth* étend ces règles en appliquant deux procédures d'extension de règles : *LEFTEXPAND* et *RIGHTEXPAND.*

Ces deux procédures permettent de découvrir des règles plus larges et visent à récupérer des emplacements *fréquents* susceptibles de générer des règles valides en étendant l'antécédent (*LEFTEXPAND*) ou le conséquent (*RIGHTEXPAND).* Plus de détails sur la génération de règles *POSR* avec *TRuleGrowth* peuvent être trouvés dans [35].

Dans ce mémoire, et comme nous ne sommes intéressés qu'à récupérer un emplacement à la fois pour l'application de la recommandation des POI, une version modifiée de *TRuleGrowth* est proposée. Elle consiste à appliquer uniquement la procédure *LEFTEXPAND*. Par conséquent, seuls les antécédents de règles à partir de règles de tailles 1 * 1 sont étendus.

Contrairement aux règles *RSS* comme nous l'avons mentionné dans la motivation, les règles *POSR* sont plus flexibles en ce qui concerne les variations de l'ordre. Par exemple, soit $R : E_1, E_2, E_3 \rightarrow E_4$ une règle *POSR*, cette règle indique qu'une recommandation de l'emplacement E_4 peut être effectuée lorsqu'un utilisateur visite $<E_1, E_2, E_3>$, $<E_2, E_1, E_3>$, $<E_3, E_1, E_2>$, $<E_3, E_2, E_1>$, $< E_1, E_3, E_2>$ *ou* $<E_2, E_3, E_1>$. Par conséquent, l'ordre est complètement ignoré dans l'antécédent de la règle.

L'utilisation de ce type de règles peut considérablement améliorer les recommandations où l'utilisateur n'a pas besoin de visiter un ensemble d'emplacements dans l'antécédent d'une règle dans le même ordre pour recevoir une recommandation en utilisant cette règle.

De plus, l'avantage de l'utilisation de *POSR* à la place de *RSS* est que la fréquence des motifs d'emplacements est généralement augmentée car de nombreux motif contenant le même ensemble d'emplacements sont considérés comme le même motif d'emplacement .Chaque motif peut être considéré alors comme apparaissant plus souvent que dans un motif avec une règle *RSS.* Cela permet donc à générer plus de règles et augmenter par la suite la couverture du système (sa capacité à établir une recommandation).

Une autre limitation liée au processus d'exploration de *RSS* est que ce processus n'est pas adapté à la recommandation à court-terme. Ce type de recommandation est très utilisé de fait que les désirs des personnes à visiter des emplacements peuvent changer à long-terme ce qui rend une recommandation long-terme inutile dans plusieurs situations. A titre d'illustration, le processus *RSS* génère plusieurs règles qui incluent le motif E_1, E_2.

Par exemple considérant les historique d'emplacement présentés dans Tableau 3.1. Dans ce Tableau, les règles $R_1 : E_1, E_2 \rightarrow E_{10}$, $R_2 : E_1, E_2 \rightarrow E_7$ sont générées dont E_{10} apparaît comme cinquième et sixième emplacement après le dernier emplacement dans l'antécédent de la règle (c'est-à-dire E_2). Ce qui est considéré une recommandation long-terme et elle est donc moins utiles qu'une recommandation des emplacements plus proches. Pour surmonter ce type de situation et ne générer que les règles vers des endroits qui pourraient être visités dans la future proche, le processus de génération de *POSR* est étendu pour considérer une contrainte de fenêtre coulissante (window). Cette dernière permet d'explorer uniquement les règles à partir d'emplacements survenant dans une fenêtre coulissante, c'est-à-dire dans un nombre maximum d'emplacements consécutifs dans chaque séquence d'emplacements. À des fins d'illustration, Tableau 3.2(a) montre un échantillon de règles de recommandation POSR générées avec une taille de fenêtre $f = 4$, *minfreq* = 20%, *et minconf* = 0,4 (40%).

Comme on peut le voir dans Tableau 3.1 (c'est le même que le tableau de premier chapitre, sauf que nous l'avons rajouté la colonne de l'historique d'emplacement), la règle $R_3 : E_1, E_2 \rightarrow E_{10}$ n'est pas extraite car E_{10} apparaît comme cinquième et sixième emplacement après le dernier emplacement dans l'antécédent de la règle (c'est-à-dire E_2) dans les séquences ce qui dépassent *f*. L'utilisation d'une contrainte de fenêtre coulissante offre plusieurs avantages.

Le premier est qu'elle permet de considérer le facteur séquentiel et permet donc d'éliminer les règles qui ne satisfont pas à ce facteur pour recommander des POI à court ou à long terme. De plus, en utilisant cette contrainte, le nombre de règles de recommandation peut être considérablement réduit en découvrant seulement ceux qui sont pertinents pour une recommandation à court terme. Ce type de recommandation est plus intéressant car l'utilisateur est plus susceptible d'agir sur la base de recommandations à court terme que celles à long-terme.

POSR avec fenêtre =4	
Règle	***Fréquence/confiance***
$R_1 : E_1 \rightarrow E_2$	Fréquence =0.25, *confiance* =0.66
$R_2 : E_1 \rightarrow E_3$	Fréquence =0.25, *Confiance*=0.66
$R_3 : E_2 \rightarrow E_3$	Fréquence =0.25, Confiance=1
$R_4 : E_2 \rightarrow E_4$	Fréquence =0.25, Confiance=1
$R_5 : E_3 \rightarrow E_4$	Fréquence =0.62, Confiance=0.83
$R_6 : E_3 \rightarrow E_7$	Fréquence =0.37, Confiance=0.5
$R_7 : E_4 \rightarrow E_7$	Fréquence =0.37, Confiance=0.5
$R_8 : E_5 \rightarrow E_6$	Fréquence =0.25, Confiance=0.66
$R_9 : E_1, E_2 \rightarrow E_3$	Fréquence =0.25, Confiance=1
$R_{10} : E_2, E_3 \rightarrow E_4$	Fréquence =0.25, Confiance=1
$R_{11} : E_3, E_4 \rightarrow E_7$	Fréquence =0.37, Confiance=0.5
$R_{12}: E_1, E_2, E_3 \rightarrow E_4$	Fréquence =0.125, Confiance=0.3
$R_{13}: E_2, E_3, E_4 \rightarrow E_7$	Fréquence =0.125, Confiance=01

(a) Avant arrangement

POSR avec fenêtre =4	
Règle	***Fréquence/confiance***
$R_1 : E_1, E2, E3 \rightarrow E_4$	*Fréquence=0.125, Confiance=0.3*
$R_2 : E_2, E3, E4 \rightarrow E_7$	*Fréquence* =0.125, *Confiance*=1
$R_3 : E_1, E_2 \rightarrow E_3$	Fréquence=0.25, *confiance* =1
$R_4 : E_2, E_3 \rightarrow E_4$	Fréquence=0.25, *confiance* =1
$R_5 : E_3, E_4 \rightarrow E_7$	Fréquence=0.37, *confiance* =0.5
$R_6 : E_2 \rightarrow E_3$	Fréquence=0.25, *confiance* =1
$R_7 : E_1 \rightarrow E_3$	Fréquence=0.25, *confiance* =0.66
$R_8 : E_1 \rightarrow E_2$	Fréquence=0.25, *confiance* =0.66
$R_9 : E_2 \rightarrow E_4$	Fréquence=0.25, *confiance* =1
$R_{10} : E_3 \rightarrow E_4$	Fréquence=0.62, Confiance=0.83
$R_{11} : E_4 \rightarrow E_7$	Fréquence=0.37, Confiance=0.5
$R_{12} : E_3 \rightarrow E_7$	Fréquence=0.37, Confiance=0.5
$R_{13} : E_5 \rightarrow E_6$	Fréquence=0.25, Confiance=0.66

(b) Après arrangement

Tableau 3. 2: Quelques règles de recommandation générées à l'aide POSR avec une contrainte de fenêtre =4.

4. Architecture du système *STS-Rec*

Dans cette section, nous présentons l'architecture de notre système recommandeur basé sur les règles. Comme illustré dans Figure 3.1, STS-Rec consiste à deux principales étapes : 1) Modélisation hors ligne et 2) Recommandation en ligne.

STS-Rec génère l'ensemble des historiques d'emplacement, puis il extrait les règles séquentielles qui sont utilisées ensuite pour la recommandation.

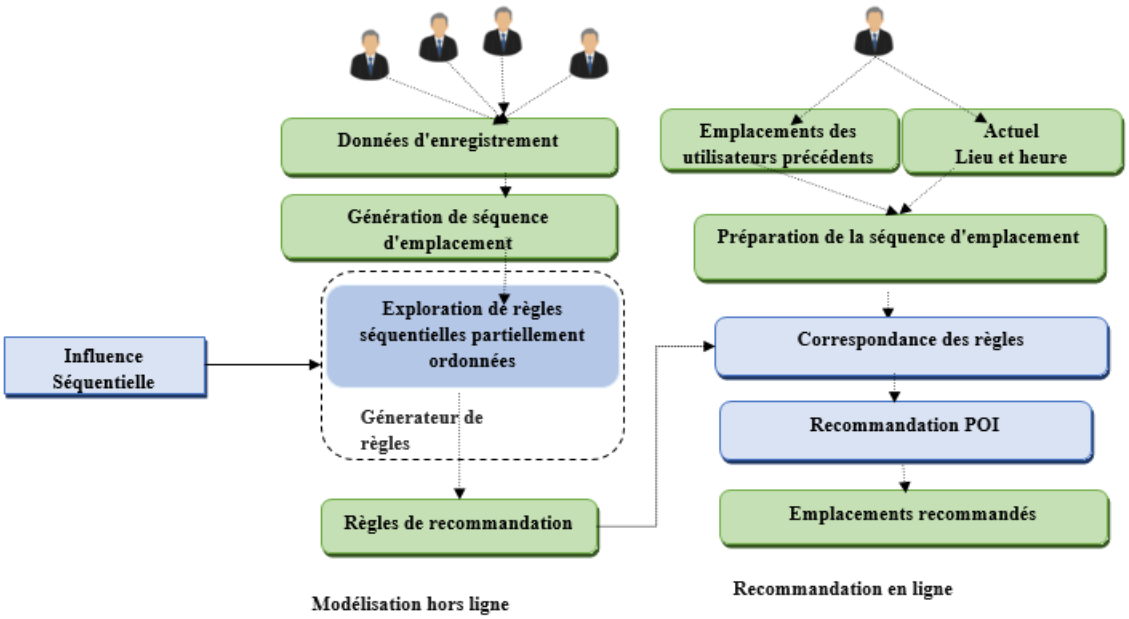


Figure 3. 1: Architecture du system STS-Rec.

4.1. Modélisation hors ligne

Elle est réalisée par deux composants, a) génération des séquences d'emplacement, et b) exploration des règles de recommandation.

4.1.1 Génération des séquences d'emplacement

Étant donné un ensemble d'utilisateurs $U= \{u_1, u_2, \ldots, u_n\}$ et un ensemble d'emplacements d'enregistrement des utilisateurs dans U, ce module génère l'historique des emplacements LH_i de chaque utilisateur dans U. Le LH_i d'un utilisateur donné u_i comporte l'ensemble des séquences d'emplacements visités chaque jour (c'est-à-dire la période de temps T = un jour).

4.1.2 Exploration de règles de recommandation

Cette section décrit d'abord le processus de génération de règles de recommandation. Ensuite, l'algorithme d'extraction de règles de recommandation incrémentale, nommé IncrPOSR est présenté.

4.1.2.1 Exploration incrémentale des règles de recommandation POSR

Au fil du temps, les intérêts d'un utilisateur peuvent changer ce qui peut affecter la qualité des recommandations. A titre d'exemple, de nouveaux marchés et restaurants peuvent s'ouvrir dans une région, ou l'utilisateur peut changer ses habitudes. Pour tenir compte ces changements, les règles doivent être périodiquement mises à jour. Pour cela, nous proposons, une approche

incrémentale pour d'exploration des règles de recommandation partiellement ordonnée, appelée *IncrPOSR*. Cette dernière traite l'historique d'emplacement *LH'* d'un RSBL où *LH'* inclut l'historique d'emplacement d'origine LH et l'historique incrémenté dénoté *ΔLH (LH '= LH + ΔLH).*

L'exploration incrémentale des règles de recommandation partiellement ordonnées comprend deux phases : la phase initiale et la phase incrémentale. Dans Figure 3.2, l'architecture de l'approche d'exploration incrémentale des règles séquentielles est illustrée.

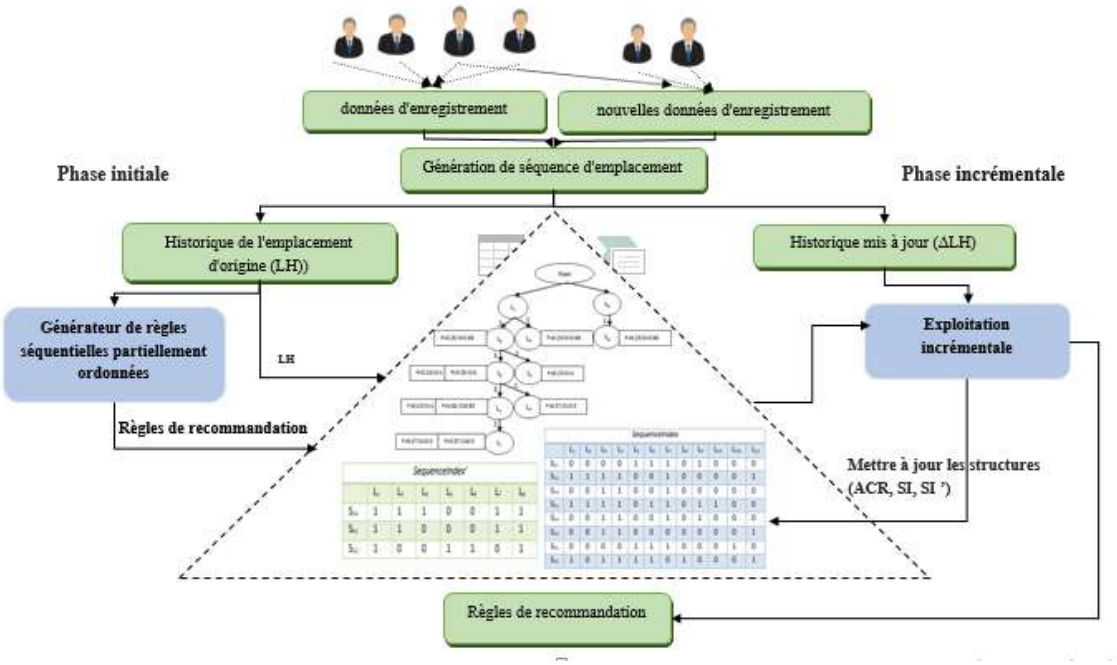


Figure 3. 2 : Architecture de l'approche incrémentale *(IncrPOSR).*

A. Phase initiale

Cette phase consiste à extraire les règles (RRS_{LH}) de l'historique d'emplacement d'origine *LH* en utilisant l'algorithme *TRuleGrowth* décrit ci-dessus. Ces règles sont ensuite considérées comme entrées dans la phase incrémentale pour découvrir les règles de recommandation *LH '*. A titre d'exemple, Tableau 3.3 montre un historique incrément des emplacements Δ*LH* de deux utilisateurs ou u_4 est un nouvel utilisateur qui a rejoint le RSBL.

Utilisateur	**Jour**	**Check-in**	**Histoire d'emplacement**	**Séquences d'emplacement**
u_1	J_4	<u_1,06:14,E_2><u_1,7:19,E_3><u_1,10:07 ,E_1>	ΔLH_{u1}	S_{14}: E_2,E_3,E_1,E_8,E_{11},E_7

		<u_1,11:38,L_8><u_1,16:09,L_{11}><u_1,20:02,L_7>		
u_4	J_4	<u_4,8:15,E_2><u_4,10:03,E_1><u_4,10:56, E_8> <u_4,14:34,E_7>	ΔLH_{u4}	S_{44}: E_2,E_1,E_8,E_7 S_{45}: E_5,E_6,E_1,E_8
	J_5	<u_4,05:20,E_5><u_4,07:23,E_6><u_4,10:34,E_1><u_4,11:30,E_8>		

Tableau 3. 3: Les modifications apportées à l'historique d'emplacement d'origine (ΔLH).

B. Phase incrémentale

Comme les lieux Check-in des utilisateurs arrivent en permanence aux fournisseurs de RSBL, les règles de recommandation extraites dans la phase initiale deviennent inutiles au fil du temps. Par conséquent, la nécessité d'une exploration incrémentale est augmentée.

1. Extraction incrémentale des règles de recommandation séquentielles RRS.

Une fois ΔLH est ajouté à LH', l'algorithme *IncrPOSR* est utilisé pour extraire progressivement les règles de recommandation *POSR* dans LH', en utilisant l'ensemble de règles de recommandation extraites dans la phase initiale RRS_{LH}.

> ***Formulation de problème*:** Étant donné l'historiques d'emplacement LH d'un ensemble d'utilisateurs *U* , un historique d'incrémental ΔLH, le seuil de *fréquence* minimale *(minfreq),* et le seuil de *confiance (minconf)*, et l'ensemble de règles valides dans LH (RRS_{LH}), le problème de l'exploration incrémentale des règles de recommandation séquentielle vise à identifier toutes les règles de recommandation valides ($RRS_{LH'}$) dans l'historique mis à jour LH '= LH ∪ ΔLH en utilisant RRS_{LH} et sans besoin d'extraire les *RRS* à partir de début.

L'idée de base de *IncrPOSR* est d'adopter trois structures de données 1) un arbre compact de règles (*ACR*) qui fournit une représentation compacte de *RRS* et 2) deux index de séquences (*SequenceIndex, SequenceIndex'*) principalement conçus pour assurer un calcul rapide des fréquences des motifs d'emplacement.

2. Arbre compact de règles (ACR).

> **Définition 5 (mouvement, ancêtre et descendant**). Étant donné une règle de recommandation $R : E_i, E_{i+1}, .., E_k \rightarrow E_p$, nous désignons par un mouvement *m*, la transition E_x, E_y entre deux emplacements consécutifs (par exemple, E_i, E_{i+1}) dans

l'antécédent de *R* ou entre le dernier emplacement dans l'antécédent de *R* (E_k) et son conséquent (E_p) où E_i est appelé l'ancêtre du mouvement *m Ancêtre (m)* .

Définition 6 (Arbre Compact des Règles ou *ACR*). L'arbre *ACR* est une représentation compacte des règles de recommandation. Il peut être formellement défini comme suit : *ACR* est une paire d'ensembles (*E*, A) où *E* est un ensemble d'emplacements (nœuds) et A = $E \times E$ est un ensemble d'arcs directionnels .Chaque nœud dans l'arbre $E_i \in E$ présente un emplacement où un POI. Dans *ACR*, chaque règle est représenté par une branche de la racine de l'arbre ou un nœud interne et se termine par un nœud interne ou une feuille. Par conséquent, un nœud dans *ACR* peut être à la fois un antécédent et un conséquent de plusieurs règles à la fois.

L'ensemble $A = \{e_1, e_2, \ldots e_m\}$ représente les mouvements entre les emplacements dans les règles de recommandation sous forme d'arc entre les nœuds d'arbre correspondant à ces emplacements. Un arc $e = (E_x E_y) \in E$ relie les nœuds Ex Ey dans *ACR* si et seulement si un mouvement entre *Ex et Ey* existe .Le mouvement peut provenir d'emplacements consécutifs à *Antec (R)* ou du dernier emplacement dans *Antec (R)* et le *conséquents (R).* On définit par $E^r_{ACR,}$ l'ensemble des arcs représentant les mouvements entre les emplacements dans *R* et reliant ses nœuds dans *ACR*.

De plus, une valeur de poids *P* est associée à chaque arc *e* reliant deux emplacements E_x, E_y dans ACR. Cette valeur de poids estime, le nombre de (la fréquence) que cet arc a été utilisé pour représenter les règles RRS. Les arcs où le nœud racine est leur ancêtre ne contiennent aucune valeur de poids.

a. Vecteur de comptage ou (count vector)

Nous associons à chaque règle R_i dans *ACR* un vecteur de comptage noté *Count_Vector* (R_i). Ce dernier est représenté avec une structure de données dynamique dans un tableau unidimensionnel et affecté au nœud correspondant au conséquent de R_i. Nous notons comme *CVect* l'ensemble de tous les vecteurs de comptage dans *ACR*.

Le vecteur de comptage d'une règle donnée R_i (*Count_Vector* (R_i)) est une structure unifiée qui pourrait englober les comptages de règles (*fréquence et confiance de (R)*) en plus des comptages de ses sous règles valides dans la même structure de données.

Une règle $R' : X' \rightarrow Y'$ est appelée une sous-règle d'une règle donnée $R : X \rightarrow Y$ si et seulement si $Y' = Y$ et X' partage un suffixe composé de son dernier emplacement avec ceux dans X. Compte tenu de l'ensemble des règles *POSR* décrites dans Tableau 3.2 (a),

la règle $R_3 : E_2 \rightarrow E_3$ est une sous règle de $R_9 : E_1, E_2 \rightarrow E_3$, et qui va partager le même *Count_vector* (R_9) dans *ACR*. La dernière cellule du *Count_Vector (R)* comporte les valeurs de comptage (*fréquence* et *confiance*) de *R* tandis que les autres cellules sont dédiées au stockage des *fréquences* et *confiances* des sous règles valides de *R*.

b. Construction de l'arbre

Exemple

Les figures 3.3, 3.4 et 3.5 illustrent les étapes de construction d'un ACR, en utilisant l'ensemble des règles présenté dans Tableau 3.2(b) générées de l'historique des emplacements d'origine LH présenté dans Tableau 3.1.

Dans Tableau 3.1 pour *minfreq* = 20% et *minconf* = 40%. Pour des raisons de concision, les alphabets *F* et *C* sont utilisés pour désigner respectivement la fréquence et la confiance de chaque règle. Les règles de Figure 2 (b) sont insérées une par une dans l'arbre.

En commençant par la première règle $R_1 : E_1, E_2 \rightarrow E_3$, le nœud E_1 est inséré en tant qu'enfant gauche de nœud Root puis, le nœud E_2 est l'enfant gauche du nœud E_1 puis E_3.

Les poids de chaque arc reliant deux nœuds (E_1, E_2 et E_2, E_3) sont fixés à 1. Dans *ACR* Un vecteur de comptage *Count_vector* (R_1) contenant deux cellules est créées et associé au nœud E_3.

La deuxième cellule comprend *la fréquence* et *la confiance* R_1 tandis que la première cellule est dédiée à une sous règle potentielle de *R* et annotée jusqu'à présent avec Nul. La deuxième règle $R_2 : E_2, E_3 \rightarrow E_4$ est inséré pour compléter la branche d'arbre de la première règle.

Le poids de l'arc E_2 E_3 est augmenté de 1 tandis que le poids d'arc reliant E_3, E_4 est met à 1. Similaire à R_1, un vecteur de comptage à deux cellules est créé pour contenir valeurs de fréquence et confiance de R_2 ainsi que le comptage d'une sous règle potentielle de R_2. Les autres

règles sont insérées de manière similaire pour obtenir l'arbre ACR.

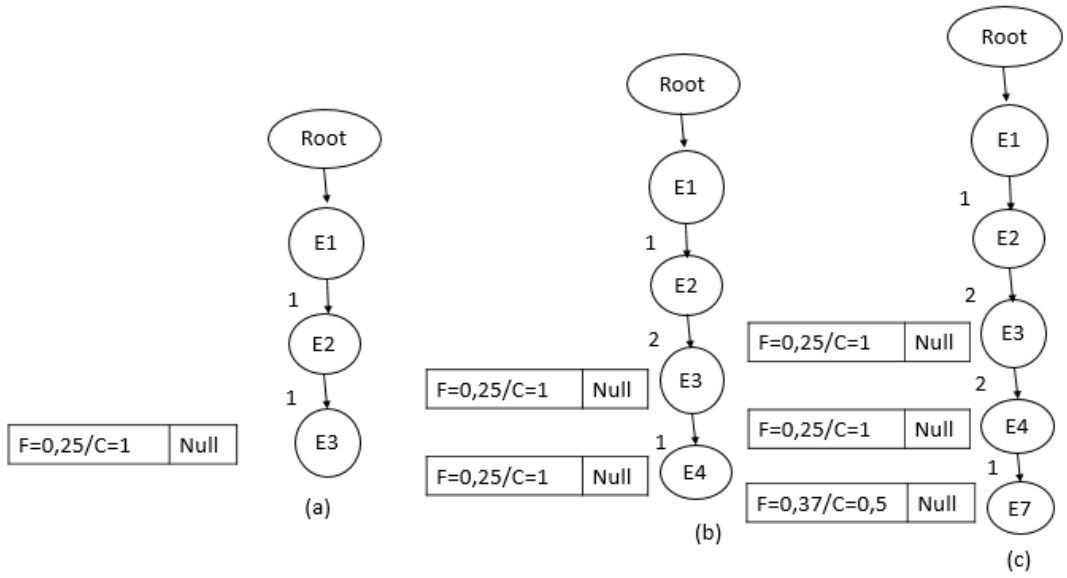


Figure 3. 3*:* L'arbre ACR après l'insertion des règles R_1, R_2, R_3.

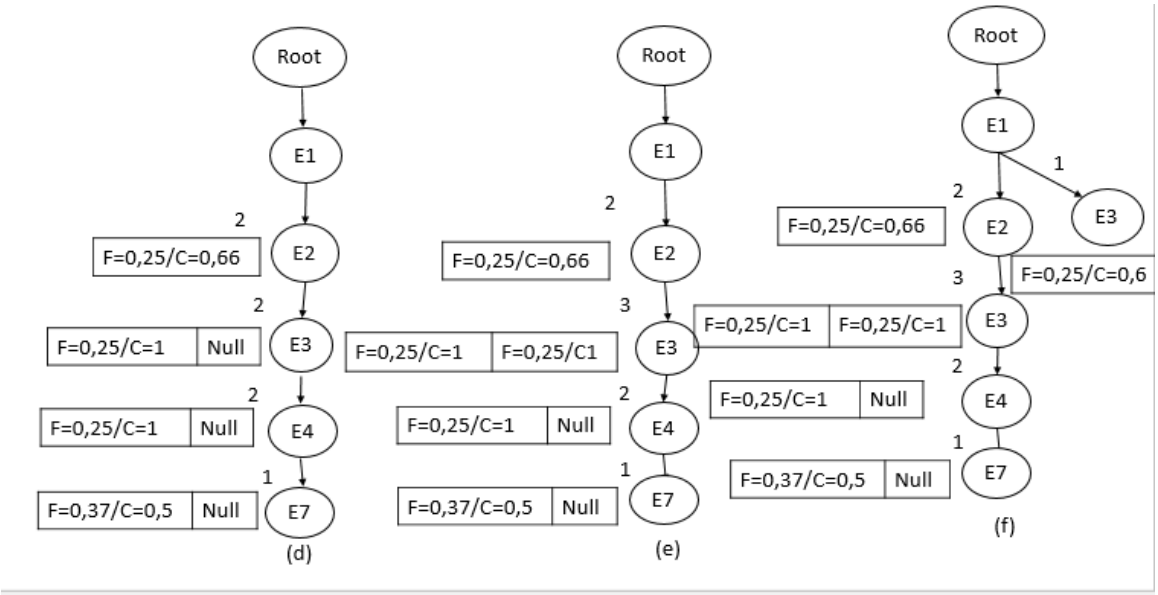


Figure 3. 4*:* l'arbre ACR après l'insertion des règles R_4,R_5, R_6**.**

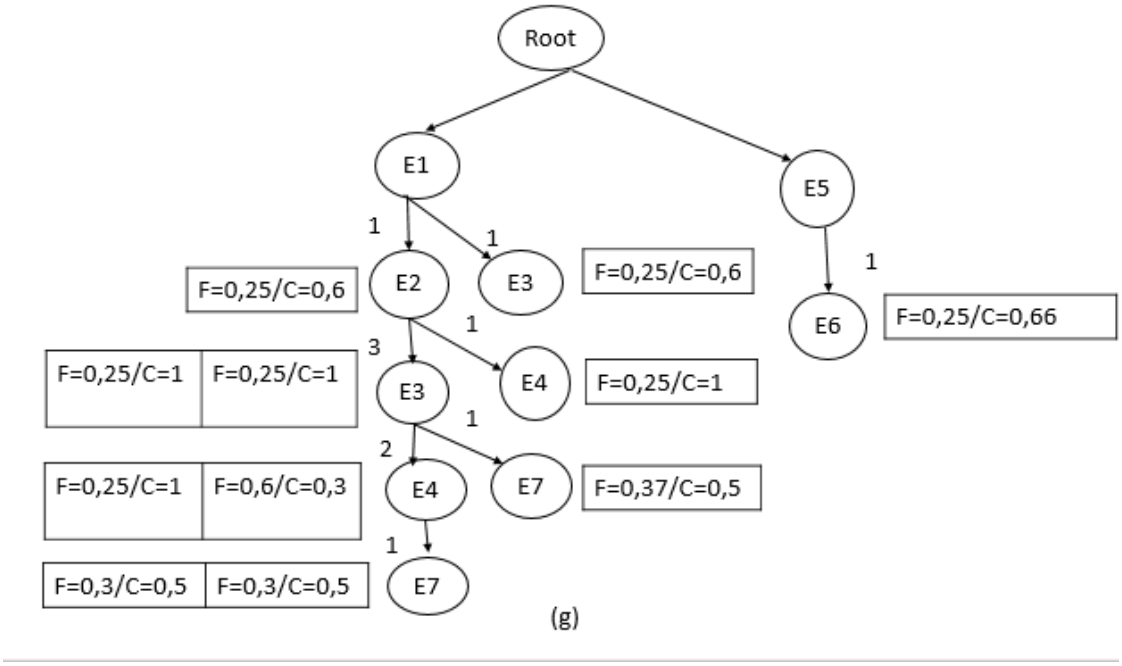


Figure 3. 5: L'arbre ACR après l'insertion de toutes les règles**.**

3. La structure *SequenceIndex*

Pour accélérer le calcul de *fréquence* des motifs et des règles dans la phase incrémentale, *Incr*POSR adopte une représentation bitmap verticale des séquences d'emplacement, appelée *SequenceIndex* (*SI*).

Cette dernière est un ensemble de vecteurs de bits indiquant de façon concise pour chaque emplacement E_i l'ensemble de séquences qu'il contient. Le bit correspondant à la séquence *S* du bitmap pour l'emplacement E_i est mis à 1 si la séquence *S* contient E_i et 0 sinon.

Le *SequenceIndex* stocke toutes les séquences d'emplacement dans l'historique d'emplacement, donc à chaque fois que la *fréquence* des motifs d'emplacement est demandée, *IncrPOSR* calcule leurs *fréquences* via une opération booléenne, sans avoir besoin d'analyser tout l'historique d'emplacement.

SI effectuent des intersections de bits (l'opération est au niveau du bit) entre les emplacements dans *S* et le bitmap dans *SequenceIndex*. Cette opération prend *O (1)* en complexité temporelle. Le nombre obtenu, dénoté comme fréquence (*S*) est divisé par *n*, où $n = |LH|$ est le nombre de séquences d'emplacement dans *SI*.

Figure 3.6 montres le *SequenceIndex* construit à partir de l'historique d'emplacement d'origine *LH* montré dans Tableau 3.1.

SequenceIndex													
	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}	E_{13}
S_{11}	0	0	0	0	1	1	0	0	0	1	0	1	0
S_{12}	1	1	1	1	1	0	0	1	0	0	1	0	0
S_{13}	0	1	0	0	1	1	0	0	0	0	0	0	0
S_{21}	1	1	1	0	1	1	1	0	1	0	1	0	0
S_{22}	1	1	1	0	0	0	1	0	0	0	0	0	0
S_{23}	0	0	0	1	0	0	0	0	1	0	0	1	0
S_{31}	0	0	0	0	1	1	0	1	0	1	0	0	0
S_{32}	0	0	0	1	0	1	1	0	1	1	0	1	1

Figure 3. 6 : La structure SequenceIndex pour LH**.**

4. La structure de SequenceIndex’

Pour améliorer encore les performances de la phase incrémentale, une structure bitmap intermédiaire similaire à *SequenceIndex,* est créée et consacrée à comprendre les nouvelles séquences d’emplacement de *ΔLH.*

À des fins d'illustration, la Figure 3.7 illustre la structure de *SequenceIndex’* basée sur Δ*LH* présenté dans Tableau 3.3 *SequenceIndex’* est principalement proposé pour accélérer le calcul de la fréquence et la confiance de motifs d’emplacement dans certains cas sans avoir besoin de charger *SequenceIndex.*

SequenceIndex’							
	E_1	E_2	E_3	E_5	E_6	E_7	E_8
S_{14}	1	1	1	0	0	1	1
S_{41}	1	1	0	0	0	1	1
S_{42}	1	0	0	1	1	0	1

Figure 3. 7: La structure SequenceIndex 'pour ΔLH**.**

5. Générer des règles candidates à partir de ΔLH

Cette étape consiste à générer puis à déterminer les règles valides qui pourraient être dérivées de l'historique d’emplacement *ΔLH.* Dans cette étape, *IncrPOSR* prend en entrée l'ensemble des séquences dans *ΔLH* et génèrent les règles valides potentielles *CRules* en respectant la contrainte de fenêtre.

L'algorithme commence par la génération des règles de taille 1 * 1, puis étendre l'antécédent de la règle pour générer des règles plus large. Chaque règle dans *CRules* doit satisfaire quatre principales conditions :

1- Un emplacement dans la conséquent de la règle de recommandation.

2- Le respect de l'ordre séquentiel des déplacements des utilisateurs enregistrés et envoyés aux fournisseurs RSBL. Notez qu'en faisant cela, la propriété de tolérance d'ordre de *POSR* ne sera pas affectée car les règles *POSR* préservent toujours un ordre partiel entre leurs antécédents et leurs conséquents.

3- Faites correspondre la définition des règles séquentielles afin qu'aucun emplacement commun n'apparaisse à la fois dans l'antécédent et dans le conséquent de la règle. Cette propriété évite de générer des règles non pertinentes, car il n'est pas pertinent de suggérer un POI déjà visité par l'utilisateur et juste donné en entrée.

4- Le respect de la contrainte de fenêtre lors de la génération des règles. Pour cela, les règles composées des emplacements qui n'apparaissaient pas dans une taille de fenêtre dans les séquences sont ignorées.

Par exemple, Figure 3.8 présente un ensemble de règles candidate dérivées de la séquence d'emplacement $S_{14} : E_2, E_3, E_1, E_8, E_{11}, E_7$ dans l'historique d'emplacement ΔLH présenté dans Tableau 3.3.

ID de séquence	***Séquence de localisation***	***Règles Candidates***			
S_{14}	$E_2\ E_3\ E_1\ E_8\ E_{11}\ E_7$	$E_2 \rightarrow E_3$	$E_2, E3 \rightarrow E_1$	$E_2, E3, E1 \rightarrow E_8$	$E_2, E3, E1, E_8 \rightarrow E_7$
		$E_3 \rightarrow E_1$	$E_3, E1 \rightarrow E_8$	$E_3, E1, E8 \rightarrow E_7$	
		$E_1 \rightarrow E_8$	$E_1, E8 \rightarrow E_7$	$E_2, E3, E1 \rightarrow E_7$	
		$E_1 \rightarrow E_{11}$	$E_1, E8 \rightarrow E_{11}$		
		$E_8 \rightarrow E_{11}$	$E_1, E11 \rightarrow E_7$		
		$E_8 \rightarrow E_7$	$E_8, E11 \rightarrow E_7$		
		$E_{11} \rightarrow E_7$	$E_2, E3 \rightarrow E_8$		
		$E_2 \rightarrow E_1$	$E_3, E3 \rightarrow E_7$		
		$E_3 \rightarrow E_8$	$E_2, E1 \rightarrow E_8$		
		$E_1 \rightarrow E_7$	$E_3, E8 \rightarrow E_7$		

Figure 3. 8 : Règles de candidature générées à partir de $S_{14} : E_2\ E_3\ E_1\ E_8\ E_{11}\ E_7$.

6. Mise à jour de l'arbre

Dans cette étape, *IncrPOSR* prend en entrée l'ancien arbre ACR (contenant les règles valides dans *LH*), l'ensemble des règles candidat *CRules* générées dans l'étape précédente,

SequenceIndex, SequenceIndex', les seuils de comptage *(minfreq, minconf)* et renvoie en sortie l'ACR mis à jour contentent compris l'ensemble des règles de recommandation valides de *LH'*.

Calcul de fréquence et de confiance. Lorsque l'historique de localisation *LH* est mis à jour en *LH', IncrPOSR* calcule les nouvelles valeurs de fréquence et confiance de chaque règle dans ACR et *CRules.* À cette fin, les structures *SequenceIndex, SequenceIndex'* ainsi que l'ACR sont utilisées. On peut distinguer deux cas.

1-Pour les règles dans ACR. Pour calculer les nouvelles valeurs de fréquence et de confiance des règles qui existent déjà dans ACR, la structure *SequenceIndex'* est utilisé avec les anciennes valeurs *(Fréquence* $(r)_{LH}$ *et Confiance* $(r)_{LH}$*)* dans le vecteur de comptage *(Count_vector)* de la règle en question. Pour cela, le chargement de *LH* ou même l'utilisation de *SequenceIndex* est évitée. Les nouvelles valeurs de comptage sont calculées comme suit :

$$fréquence(r)_{LH'} = \frac{fréquence(r)_{LH} \times |LH| + fréquence\ (r)_{\Delta LH} \times |\Delta LH|}{|LH| + |\Delta LH|}$$

$$confiance(r)_{LH'} = \frac{fréquence(r)_{LH} \times |LH| + fréquence\ (r)_{\Delta LH} \times |\Delta LH|}{\frac{fréquence(r)_{LH} \times |LH|}{confiance(r)_{LH}} + fréquence(Antec(r))_{\Delta LH} \times |\Delta LH|}$$

Lorsque la *fréquence(r)*$_{LH,}$ *la confiance(r)*$_{LH}$ est extraite du vecteur de comptage de *r* (Count_vector (r)) tandis que la *fréquence(r)*$_{\Delta LH}$ est la *fréquence (Antec (r))*$_{\Delta LH}$ sont calculés à l'aide de *SequenceIndex'*

2-Pour les règles dans CRules. Comme aucune connaissance préalable si ACR comprend une règle de *CRules*, le calcul de la fréquence et de la confiance de chaque règle dans *CRules* exploite l'union de *SequenceIndex* et Structures de *SequenceIndex'.* Dans ce cas, les valeurs de comptage sont calculées comme suit :

$$fréquence(r)_{LH'} = \frac{fréquence(r)_{LH} \times |LH| + fréquence(r)_{\Delta LH} \times |\Delta LH|}{|LH'|}$$

$$confiance(r)_{LH'} = \frac{fréquence(r)_{LH} \times |LH| + fréquence(r)_{\Delta LH} \times |\Delta LH|}{fréquence\ (Antec(r))_{LH} \times |LH| + fréquence(Antec(r))_{\Delta LH} \times |\Delta LH|}$$

Où la *fréquence(r)*$_{LH}$, la *fréquence (Antec(r))*$_{LH}$ sont calculés à partir de *SequenceIndex* tandis que la *fréquence (r)*$_{\Delta LH}$ est la *fréquence (Antec (r))*$_{\Delta LH}$ sont calculés à l'aide de *SequenceIndex'*.

À la fin de cette étape, la structure *SequenceIndex'* est recopié dans *SequenceIndex* et *sequenceIndex'* est supprimé par la suite.

7. Mettre à jour des structures

Dans cette étape, l'ancien ACR, construit à partir de *LH*, est ajusté pour intégrer de nouvelles règles valides parmi celles de *CRules* et supprimer les règles non valides qui existaient déjà dans ACR. Plusieurs cas peuvent être distingués.

1. La règle candidate est valide en *LH'*
2. La règle candidate n'est pas valide en *LH'*
3. Une règle valide dans *LH* est toujours valide dans *LH'*
4. Une règle valide dans *LH* devient invalide dans *LH'*

1. La règle candidate est valide en LH'. Dans ce cas, la règle valide est insérée dans l'arbre. Ce cas peut se produire lorsque les séquences nouvellement insérées de ΔLH augmentent les fréquences ou confiances de certains motifs de localisation qui ont été peu fréquents ou génèrent des règles non valides en *LH*. Ce cas peut aussi de produire si ΔLH inclus un nouveau ensemble de motifs fréquents qui n'existent pas auparavant dans *LH* et qui permet de générer des règles avec une valeur de confiances supérieures à minconf. Par exemple, la règle candidate $R_7 : E_1 \rightarrow E_8$ dans Figure 3.9 est une règle valide et elle est insérée dans ACR.

2. La règle candidate n'est pas valide en LH'. Lorsqu'une règle candidate est considérée comme non valide, aucune modification ne doit être effectuée dans l'arbre. Une règle candidate $r = X \rightarrow Y$ pourrait être invalide dans deux cas: (1) Fréquence (r) <*minfreq* et confiance (r) <*minconf* ou (2) fréquence (r)> = *minfreq* mais confiance (r) <*minconf.*

3. Une règle valide dans LH est toujours valide dans LH'. Dans ce cas, comme la règle existe déjà dans ACR, *IncrPOSR* met à jour *Count_Vector (r)* avec les nouvelles valeurs de fréquence et de confiance. Ce cas peut se produire lorsque la *fréquence* et *la confiance* de la règle augmentent, diminuent mais reste supérieures des seuils ou restent les mêmes en insérant ΔLH.

4. Une règle valide dans LH devient invalide dans LH'. Dans ce cas, la règle non valide est supprimée de l'arbre. Même si de nombreuses règles se chevauchent dans ACR, la suppression d'une règle d'ACR se fait simplement en diminuant les poids des arcs reliant les nœuds qui représentent la règle, puis en modifiant la fréquence la confiance de la règle r dans

Count_Vector (r) à Null. Ce processus de suppression des règles non valides du ACR est répété jusqu'à aucune règle non valide n'existe dans ACR.

À la fin de ce processus, une étape supplémentaire de nettoyage est appliquée. Cette étape vise initialement à supprimer les vecteurs vides (avec une valeur Null dans toutes les cellules) et les nœuds isolés.

Ensuite, les segments d'arbre isolés qui représentent des règles sont connectés à l'arbre en trouvant les branches de l'arbre qui correspondent totalement ou partiellement aux règles restantes et en les attachant à ces branches. Sinon, les segments isolés sont attachés au nœud racine si aucune branche correspondante n'est trouvée. Par exemple, les Figures 3.10, 3.11, 3.12 et 3.13 illustrent un exemple de suppression de règles non valides de l'arbre.

Figure 3.9 montre le nouvel ensemble de règles valides parmi celles de ACR et *CRules* pour *minfreq* = 0,2, *minconf* = 0,4 et *f* = 3.

Règle	**Statut de la règle**		**Les valeurs de fréquence et confiance**
R_1:E→E_3	Valide pour LH'	Existe dans ACR et compte les valeurs mises à jour	Fréquence=0.27, Confiance =0.75
R_2: E_3→E_4	Valide pour LH'	Existe dans ACR et compte les valeurs mises à jour	Fréquence=0.45, Confiance =0.71
R_3: E_3→E_7	Valide pour LH'	Existe dans ACR et compte les valeurs mises à jour	Fréquence=0.27, Confiance =0.42
R_4: E_4→E_7	Valide pour LH'	Existe dans ACR et compte les valeurs mises à jour	Fréquence=0.27, Confiance =0.5
R_5: E_5→E_6	Valide pour LH'	Existe dans ACR et compte les valeurs mises à jour	Fréquence=0.27, Confiance =0.75
R_6: E_3,E_4→E_7	Valide pour LH'	Existe dans ACR et compte les valeurs mises à jour	Fréquence=0.27 Confiance =0.5
R_7: E_1→E_8	Valide pour LH'	Une règle candidate valide dans LH'	Fréquence=0.27 Confiance =0.5
R_8: E_1→E_2	Non valide pour LH '	À supprimer de l'ACR	Fréquence=0.18, Confiance =0.33
R_9: E_1→E_3	Non valide pour LH '	À supprimer de l'ACR	Fréquence=0.18, Confiance =0.33
R_{10}: E_2→E_4	Non valide pour LH '	À supprimer de l'ACR	Fréquence=0.18, Confiance =0.5
R_{11}: E_1,E_2→E_3	Non valide pour LH '	À supprimer de l'ACR	Fréquence=0.18, Confiance =0.5
R_{12}: E_2,E_3→E_4	Non valide pour LH '	À supprimer de l'ACR	Fréquence=0.18, Confiance =0.66

Figure 3. 9: L'ensemble des règles valides dans LH'.

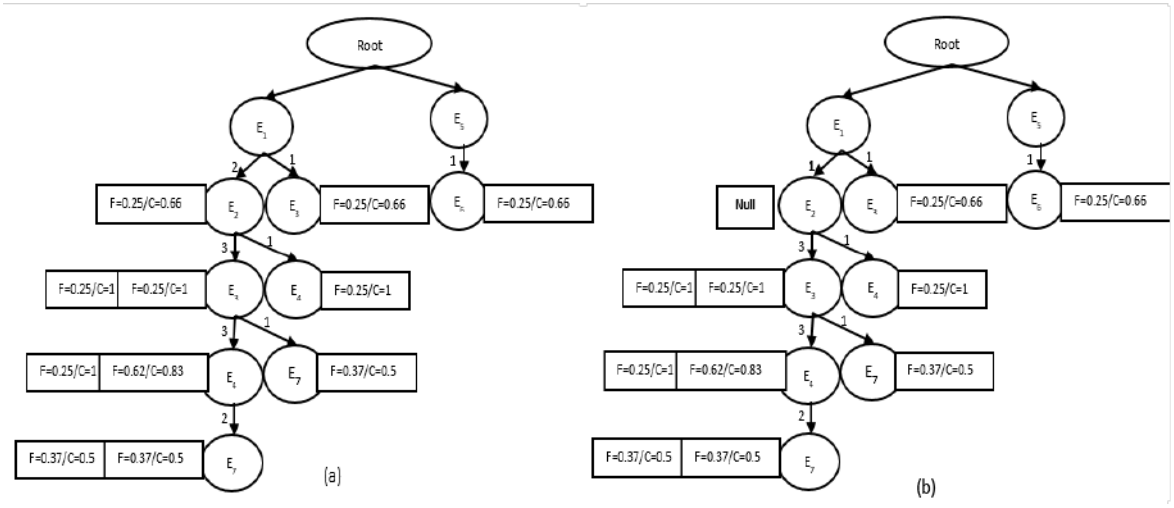


Figure 3. 10 : (a) L'état initial du ACR (b) ACR après suppression de la règle $E_1 \rightarrow E_2$.

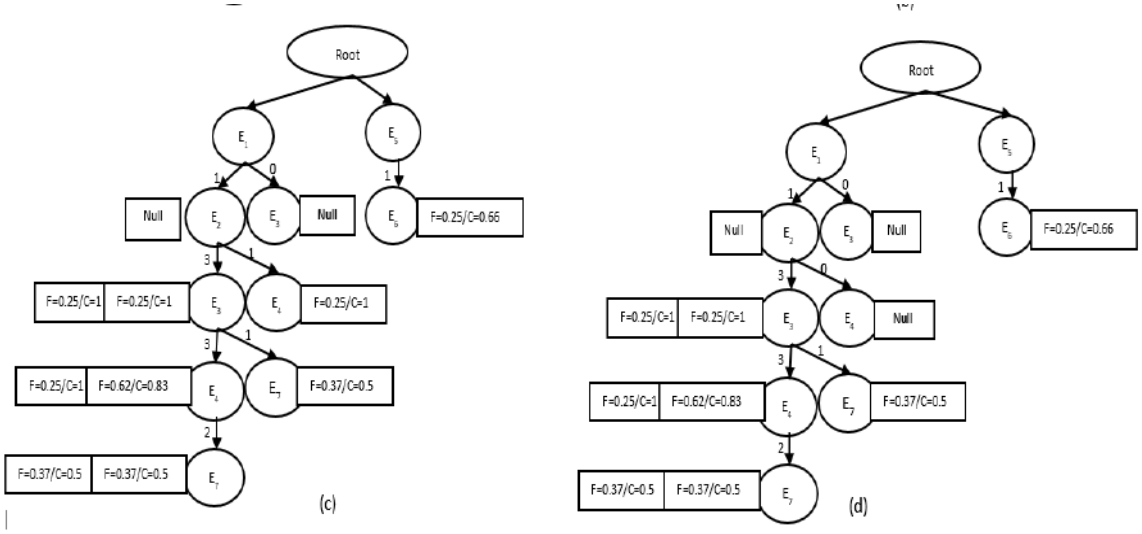


Figure 3. 11: (c) ACR après suppression de la règle $E_1 \rightarrow E_3$, (d) ACR après suppression de $E_2 \rightarrow E_4$,

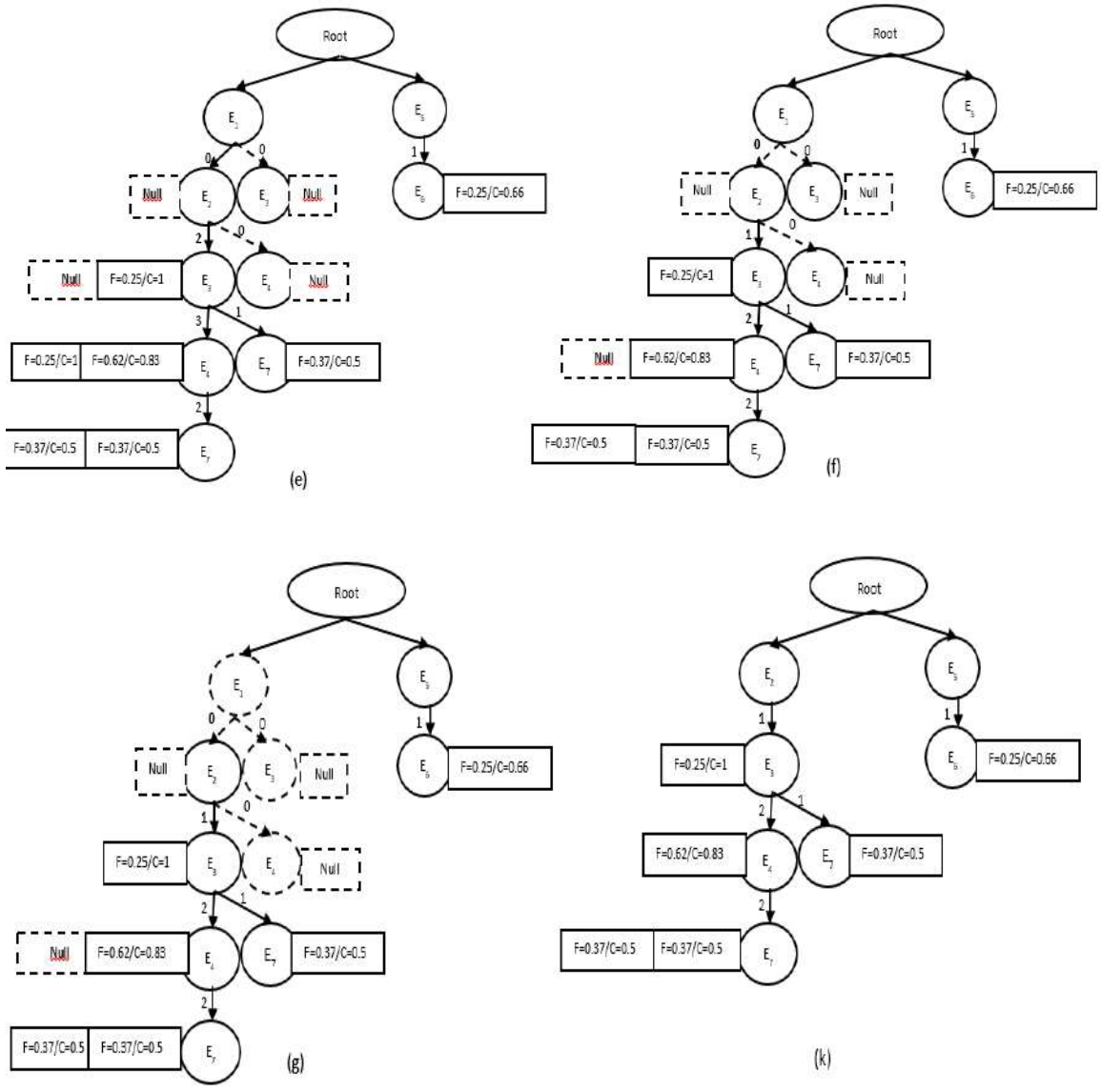


Figure 3. 12 :(e) ACR après suppression de E_1, E_2 →E3, (f) ACR après suppression de E_2, E_3 →E_4. (g, k) nettoyage et attachement des processus et ACR après la suppression de toutes les règles invalides dans LH '.

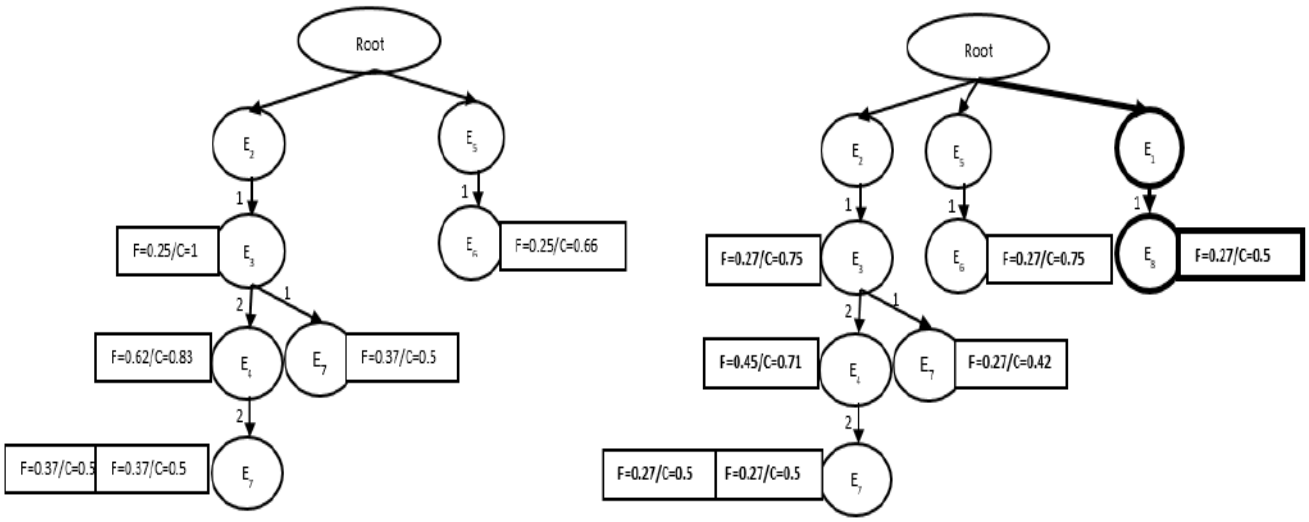


Figure 3. 13*:* Mise à jour de l'arbre.

4.1.3 Recommandation en ligne

Ce module est composé de trois composants : (a) Préparation de la séquence d'emplacement (b) Correspondance des règles et (c) Recommandation de POI.

4.1.3.1 Préparation de la séquence d'emplacement

Ce composant collecte et agrège les POI (check-in) d'un utilisateur et les transformé en séquences POI.

4.1.3.2 Correspondance des règles.

Ce composant essaye de trouver un l'ensemble des règles de recommandation, parmi celles générées par le module de modélisation hors ligne, qui correspond à l'ensemble des POI précédemment visités par l'utilisateur.

4.1.3.3 Recommandation de POI.

Une fois l'ensemble des règles correspondantes est obtenu, le module de recommandation de POI sélectionne les N règle ayant la plus grande valeur de confiance, et montrent leurs conséquents à l'utilisateur comme des emplacements possibles qu'il pourrait être intéressé à visiter.

5. Conclusion

Ce chapitre présent notre système de recommandation nommée STS-Rec. Ce dernier qui prend en compte les influences séquentielles sur le comportement de mobilité pour la recommandation de POI.

Chapitre 3

STS-Rec utilise dans un premier temps des règles de recommandation en utilisant un nouveau type des règles séquentielles et utilisent ensuite ces règles pour suggérer de nouveaux lieux aux utilisateurs standards, l'approche proposée est incrémentale et elle n'exige pas un ordre strict des emplacements dans des règles générée.

Chapitre 4

Étude expérimentale

Chapitre 4

1. Introduction

Ce chapitre décrit l'évaluation du système *STS-Rec* proposé et qui base sur l'approche *IncPOSR*. Pour mieux étudier son efficacité, les performances de *STS-Rec* sont comparées avec un système basé sur l'approche non incrémentale de système appelée *SSR-Rec* et qui base sur l'algorithme *TRuleGrowth*. Le code source de cet algorithme peut être téléchargé de la bibliothèque SPMF [36]. Les sections suivantes présentent le cadre expérimental, les expériences menées et discute des résultats obtenus.

2. Enivrement d'expérimentation

Les expériences menées ont été réalisées sur un ordinateur équipé d'un processeur Intel doté 2 Go de RAM et 200 Go de disque dur. L'approche proposée a été mise en œuvre en Java en utilisant l'implémentation algorithme TRuleGrowth.

2.1 Jeux de données (Datasets)

Pour évaluer les performances de notre système, des expériences ont été réalisées en utilisant deux datasets de check-in réel et à grande échelle nommées *Brightkite* [37] et *Gowalla* [38]. Dans ces jeux de données, un check-in est décrit par un ID utilisateur, un ID d'emplacement, une latitude d'emplacement, une longitude d'emplacement et un horodatage. Ces jeux de données ont été largement utilisés pour la recommandation et la prédiction de localisation.

1. Brightkite

Brightkite était un fournisseur de services de réseaux sociaux basés sur la localisation (*RSBL*) où les utilisateurs partageaient leur emplacement en check-in. Le réseau d'amitié a été collecté à l'aide de leur application publique. Ce jeu de données contient les interactions entre les utilisateurs ainsi que le lieu d'enregistrement de chaque utilisateur.

Dataset	**Utilisateurs**	**Période**
Britghkite	58.228	Avr 2008-Oct 2010

Tableau 4. 1: Les statistiques des datasets**.**

2.2 Modèles évalués

Notre système de recommandation *STS-Rec* basé sur l'approche incrémentale constituant d'une structure d'arbre et des structures bitmap a été comparé avec le *SSR-Rec* (Système de recommandation basé sur POSR mais non incrémentale). Ce système génère des règles de recommandation séquentielles qui utilisent l'algorithme *TRuleGrowth*, mais base sur l'approche classique non incrémentale.

2.3 Les métriques d'évaluation

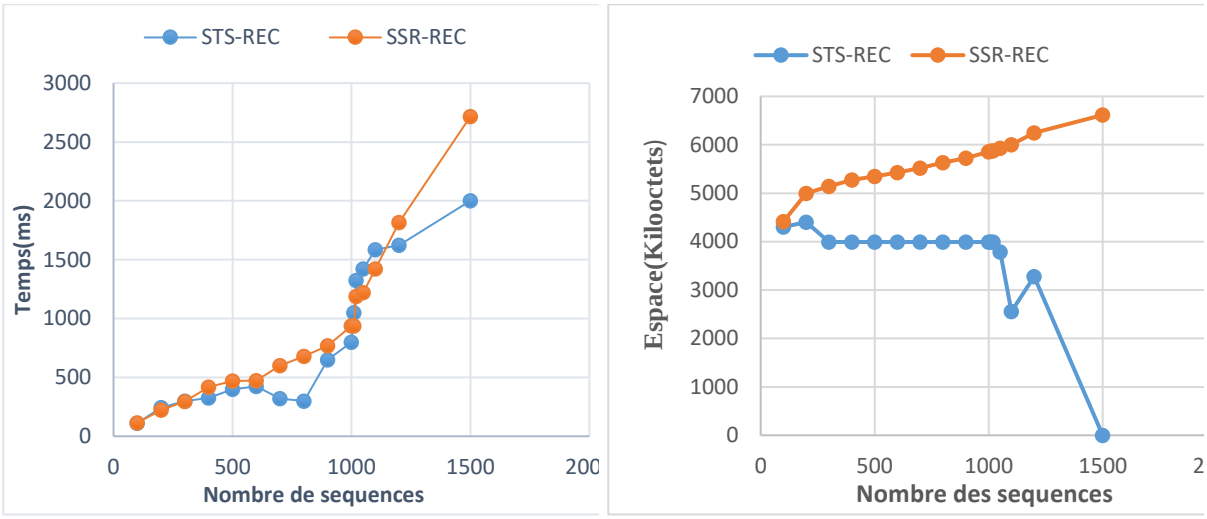
Pour évaluer la performance de notre système de recommandation, deux mesures bien connues et bien utilisé notamment l'espace de stockage et le temps d'exécution.

3 Expériences

X) Etude de la scalabilité

Dans cette expérience, nous visons à étudier la scalabilitée en augmentant le nombre des séquences de localisation considérés. La scalabilitée est un facteur important pour la plupart des tâches de recommandation pour mesurer la capacité de système de recommandation à maintenir ses fonctionnalités et ses performances en cas d'une forte demande (nombre élevé des check-in).

Dans cette expérience, le nombre de séquences de localisation a été varié de 100 à 1500 séquences et la taille de la fenêtre a été utilisée 2. Comme illustré dans Figure 4.1 (pour Brightkite), le temps d'exécution ainsi que l'espace ont augmenté avec l'augmentation de nombre des séquences pour SSR-Rec par contre au système STS-Rec, il utilise moins de temps et moins d'espace, malgré l'augmentation des séquences. La raison derrière ces résultats est qu'en augmentant le nombre des séquences, plus de temps et d'espace sont nécessaires pour extraire et stocker les règles séquentielles**.**



Temps - Brightkite (ms)

Espace - Brightkite (KiloOctets)

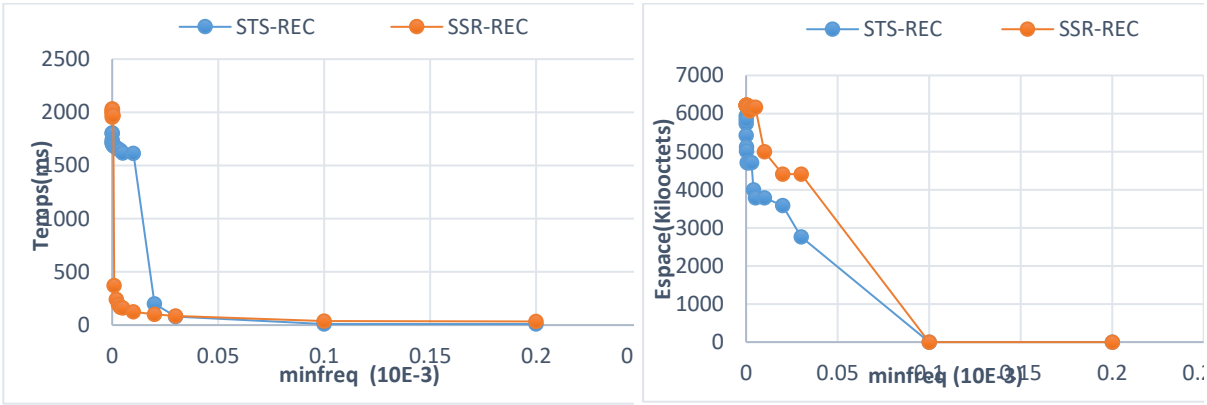
Figure 4. 1 : Etude de la scalabilité.

3.1 Effet des paramètres

Cet ensemble d'expériences évalue l'influence de la variation des paramètres: *minfreq* et *minconf* sur les performances de systèmes de recommandations.

Y) Impact de la variation de minfreq

Dans une cette expérience, l'influence de variation du seuil *minfreq* sur le temps et l'espace a été évaluée. Les résultats présentés dans Figure 4.2 montrent qu'en augmentant *minfreq*, le temps et l'espace diminuent jusqu'à ce qu'ils soient nul pour SSR-Rec et STS-Rec mais ce dernier a de meilleurs résultats que SSR-Rec (non incrémentale). Cela est due de la réalité qu'en augmentant *minfreq* peu de règles fréquentes sont extraites ce qui nécessitent moins de temps et l'espace pour extraire ces règles et les stocker par la suite.



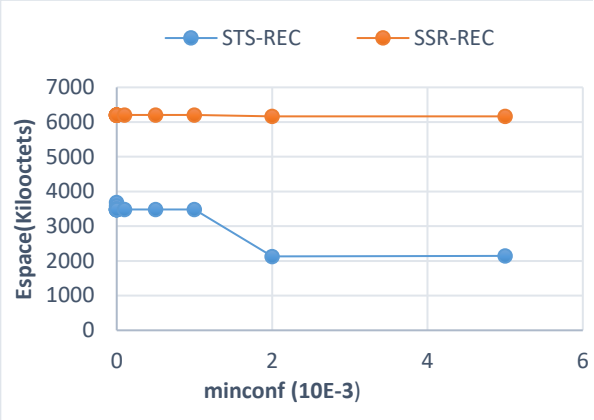
Temps - Brightkite (ms)

Espace - Brightkite (KiloOctets)

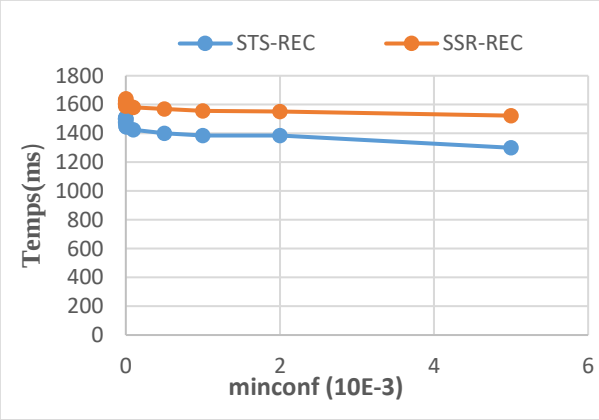
Figure 4. 2: Impact de la variation du seuil minfreq

Z) Impact de la variation de minconf

Dans Figure 4.3, nous examinons l'effet de la variation du seuil minconf sur le temps d'exploration des règles et l'espace mémoire requis pour le stockage de ces règles. Comme prévu, et de même que *minfreq*, l'augmentation de *minconf* conduit la diminution de temps et d'espace en raison que le nombre des règles valides se diminue de ce fait mais STS-Rec a de meilleurs résultats avec beaucoup moins de temps et d'espace que SSR-Rec.



Espace - Brightkite (KiloOctets)



Temps - Brightkite (ms)

Figure 4. 3 : Impact de la variation de *minconf.*

4 Conclusion

Dans ce chapitre, une étude expérimentale a été menée avec le jeu de données réel (Brightkite). Les résultats ont montré que le recommandeur proposé a d'excellentes performances par rapport au modèle SSR-Rec (basé sur POSR mais non incrémentale), ainsi à travers les résultats, nous avons remarqué que l'approche STS-Rec (l'approche incrémentale) est plus rapide et demande moins d'espace par rapport à l'approche non incrémentale (dans l'utilisation du temps et de l'espace mémoire) dans les trois expériences que nous avons fait.

Conclusion générale

Conclusion générale

Le développement des technologies mobiles et l'utilisation généralisée des appareils mobiles, a provoqué l'émergence d'un grand nombre de réseaux sociaux basés que les locations proposant des divers services comme la recommandation de localisation.

Dans ce mémoire, nous avons proposé un système de recommandation appelé *STS-Rec* qui considère l'influence séquentielle pour la recommandation de POI. *STS-Rec* extrait d'abord les règles de recommandation en utilisant un nouveau type de générateur de règles séquentielles, puis utilise ces règles pour suggérer de nouveaux sites aux utilisateurs à l'aide de l'algorithme IncrPOSR (basé sur une modèle d'arbre et deux structures).

Contrairement aux autres modèles de la littérature, l'approche qu'on propose est incrémentale et ne nécessite pas un ordre strict des emplacements dans des règles extraites ce qui permet de réaliser une recommandation pertinente et plus précise.

Une étude expérimentale approfondie a été menée à l'aide de jeu de donnée RSBL réels Brightkite. Les résultats ont montré que *STS-Rec* présente d'excellentes performances par rapport à la version non incrémentale de système.

En ce qui concerne les futurs travaux, de nombreuses améliorations sont envisagées, telles que la prise en compte de 1) l'influence social qui prend en compte les intérêts des amis de l'utilisateur, 2) l'influence temporelle (par exemple, aller à l'université le matin, aller au restaurant à midi et au cinéma après le dîner, etc. ...), 3) l'influence géographique qui prend en compte la distance entre les emplacements, et choisit l'emplacement le plus proche de l'utilisateur, et à partir de là, il sera possible de faire des recommandations et de prédictions plus précise et qui ne reposent pas uniquement sur des données séquentielles.

Conclusion générale

Références bibliographiques

Références bibliographiques

[1] C. Cheng, R. Jain, et E. van den Berg, « Location prediction algorithms for mobile wireless systems », in *Wireless internet handbook: technologies, standards, and application*, 2003, p. 245–263.

[2] M. Ye, P. Yin, et W.-C. Lee, « Location recommendation for location-based social networks », in *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, 2010, p. 458–461.

[3] J. Bao, Y. Zheng, et M. F. Mokbel, « Location-based and preference-aware recommendation using sparse geo-social networking data », in *Proceedings of the 20th international conference on advances in geographic information systems*, 2012, p. 199–208.

[4] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, et Y. Rui, « GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation », in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, p. 831–840.

[5] J.-B. Griesner, T. Abdessalem, et H. Naacke, « POI recommendation: Towards fused matrix factorization with geographical and temporal influences », in *Proceedings of the 9th ACM Conference on Recommender Systems*, 2015, p. 301–304.

[6] Y.-L. Zhao, L. Nie, X. Wang, et T.-S. Chua, « Personalized recommendations of locally interesting venues to tourists via cross-region community matching », *ACM Trans. Intell. Syst. Technol. TIST*, vol. 5, n° 3, p. 1–26, 2014.

[7] K. W.-T. Leung, D. L. Lee, et W.-C. Lee, « CLR: a collaborative location recommendation framework based on co-clustering », in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, 2011, p. 305–314.

[8] M. Ye, P. Yin, W.-C. Lee, et D.-L. Lee, « 6 », in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, 2011, p. 325–334.

[9] E. Cho, S. A. Myers, et J. Leskovec, « Friendship and mobility: user movement in location-based social networks », in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, p. 1082–1090.

[10] C. Cheng, H. Yang, I. King, et M. R. Lyu, « Fused matrix factorization with geographical and social influence in location-based social networks », in *Twenty-sixth AAAI conference on artificial intelligence*, 2012.

[11] H. Li, Y. Ge, R. Hong, et H. Zhu, « Point-of-interest recommendations: Learning potential check-ins from friends », in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, p. 975–984.

[12] H. Gao, J. Tang, X. Hu, et H. Liu, « Exploring temporal effects for location recommendation on location-based social networks », in *Proceedings of the 7th ACM conference on Recommender systems*, 2013, p. 93–100.

[13] Q. Yuan, G. Cong, Z. Ma, A. Sun, et N. M.- Thalmann, « Time-aware point-of-interest recommendation », in *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, 2013, p. 363–372.

[14] D. Yang, D. Zhang, Z. Yu, et Z. Wang, « A sentiment-enhanced personalized location recommendation system », in *Proceedings of the 24th ACM conference on hypertext and social media*, 2013, p. 119–128.

[15] C. Cheng, H. Yang, I. King, et M. R. Lyu, « Fused matrix factorization with geographical and social influence in location-based social networks », in *Twenty-sixth AAAI conference on artificial intelligence*, 2012.

[16] B. Liu, Y. Fu, Z. Yao, et H. Xiong, « Learning geographical preferences for point-of-interest recommendation », in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, p. 1043–1051.

[17] B. Liu, H. Xiong, S. Papadimitriou, Y. Fu, et Z. Yao, « A general geographical probabilistic factor model for point of interest recommendation », *IEEE Trans. Knowl. Data Eng.*, vol. 27, n° 5, p. 1167–1179, 2014.

Références bibliographiques

[18] W. Wang, H. Yin, S. Sadiq, L. Chen, M. Xie, et X. Zhou, « SPORE: A sequential personalized spatial item recommender system », in *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, 2016, p. 954–965.

[19] J. Sang, T. Mei, et C. Xu, « Activity sensor: Check-in usage mining for local recommendation », *ACM Trans. Intell. Syst. Technol. TIST*, vol. 6, n° 3, p. 1–24, 2015.

[20] B. Liu, Y. Fu, Z. Yao, et H. Xiong, « Learning geographical preferences for point-of-interest recommendation », in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, p. 1043–1051.

[21] X. Wang *et al.*, « Semantic-based location recommendation with multimodal venue semantics », *IEEE Trans. Multimed.*, vol. 17, n° 3, p. 409–419, 2014.

[22] D. Hand, H. Mannila, ET Smyth, P. « Principles of Data Mining». Bradford Books». Adaptive computation and Machine Learning Series, 2001.

[23] H. Jiawei, ET M. Kamber, « Data Mining Concepts and Techniques », Morgan Kaufmann Publishers, 2001.

[24] DONIA HAMMAMI, ALYA LETAIF, «Techniques du Data Mining», a Proven Plan.11/5/2016.

[25] M. Zaki, J. SPADE, «An Efficient Algorithm for Méninge Frequent Sequences». Ma-chine Learning 42(1/2), 31–60 (2001).

[26] B. Liu, W. Hsu, Y. Ma, « Integrating Classification and Association Rule Mining». In: Fourth International Conference on Knowledge Discovery and Data Mining (KDD 1998), pp. 80–86. AAAI Press, (1998).

[27] R. Rakotomala, «Arbre de décision», Revue MODULAD, numéro 33, 2005.

[28] J. Baltié, « Datamining : ID3 et C4.5», Promotion Spécialisation S.C.I.A. Ecole pour l'informatique et techniques avancées, 2002,

[29] W. McCulloch, W. Pitts, «a Logical Calculus of Ideas Immanent in Nervous Activity», Bulletin of Mathematical Biophysics 5:115-133, 1943.

[30] R. Quinlan: C4.5 «Programs for Machine Learning», Morgan Kaufmann Publishers Inc., 1993.

[31] K. Thearling, « An Introduction to Data Mining », sur thearling.com (consulté le 2 mai 2011).

[32] R.Agrwale, A. Srikant. «Fast algorithms for mmining association rule». Proc, pp. 787-499, VLDB 1994.

[33] H. Jiawei, M. Kamber, «Data Mining Concepts and Techniques», published by Morgan Kauffman, 2nd Ed, 2006.

[34] R. Agrawal, R. Srikant. «Fast algorithms for mining association rules in large databases». Research Report RJ 9839, IBM Almaden Research Center, San Jose, California, June 1994.

[35] Philippe Fournier-Viger, Cheng-Wei Wu, Vincent S. Tseng, Longbing Cao et Roger Nkambou « Mining partially-ordered sequential rules common to multiple sequences », IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, p. 2203–2216, 2015.

[36] Fournier-Viger, P., et al. SPMF: à Java open-source pattern mining Library. The Journal of Machine Learning Research, 2014. 15(1): p. 3389-3393.

[37] Loubna Boujlaleb, A., et al, «a Feature Selection for Community Evolution Prediction in Location-Based Social Network: Gowalla and Brightkite », p. 404-412

[38] Colleen Cuddy, N, et al, « Location-Based Services: Foursquare and Gowalla, Should Libraries Play? » Published online: p336-343, 03 Dec 2010.

[39] P Fournier-Viger, R Nkambou, VSM Tseng , «RuleGrowth: Mining Sequential Rules Common to Several Sequences by Pattern-Growth», p2-3,

Références bibliographiques

Références bibliographiques