Détection de données aberrantes à partir de motifs fréquents sans énumération exhaustive

Arnaud Giacometti, Arnaud Soulet

Université François-Rabelais de Tours, LI EA 6300
Campus de Blois, 41000 Blois
prenom.nom@univ-tours.fr

Résumé. La détection de données aberrantes (outliers) consiste à détecter des observations anormales au sein des données. Durant la dernière décennie, des méthodes de détection d'outliers utilisant les motifs fréquents ont été proposées. Elles extraient dans une première phase tous les motifs fréquents, puis assignent à chaque transaction un score mesurant son degré d'aberration (en fonction du nombre de motifs fréquents qui la couvre). Dans cet article, nous proposons deux nouvelles méthodes pour calculer le score d'aberration fondé sur les motifs fréquents (FPOF). La première méthode retourne le FPOF exact de chaque transaction sans extraire le moindre motif. Cette méthode s'avère en temps polynomial par rapport à la taille du jeu de données. La seconde méthode est une méthode approchée où l'utilisateur final peut contrôler l'erreur maximale sur l'estimation du FPOF. Une étude expérimentale montre l'intérêt des deux méthodes pour les jeux de données volumineux où une approche exhaustive échoue à calculer une solution exacte. Pour un même nombre de motifs, la précision de notre méthode approchée est meilleure que celle de la méthode classique.

1 Introduction

La détection des données aberrantes consiste à détecter les observations anormales au sein d'un jeu de données (Hawkins, 1980). Ce problème de détection des données aberrantes a d'importantes applications telles que la détection de fraudes bancaires ou d'intrusions réseau. Récemment, des méthodes de détection de données aberrantes ont été proposées pour les données catégorielles en utilisant le concept de motifs fréquents (He et al., 2005; Otey et al., 2006; Koufakou et al., 2011). L'idée clé de ces approches est de considérer le nombre de motifs fréquents couvrant chaque observation. Il est peu probable qu'une observation couverte par un grand nombre de motifs fréquents soit une donnée aberrante puisque les motifs fréquents correspondent aux « caractéristiques communes » du jeu de données. Ces méthodes de détection extraient d'abord tous les motifs fréquents du jeu de données et ensuite attribuent à chaque observation un score mesurant le degré d'aberration en comptabilisant les motifs fréquents qu'elle contient. Ces méthodes de détection de données aberrantes suivent donc un schéma en deux étapes : des motifs locaux vers un modèle global.

Les méthodes en deux étapes (Knobbe et al., 2008) visent à extraire exhaustivement tous les motifs locaux d'un jeu de données (première étape) afin de construire des modèles globaux

(deuxième étape) comme des classifieurs (Liu et al., 1998) ou des mesures calculées à partir de motifs (par exemple, FPOF (He et al., 2005) ou CPCQ (Liu et Dong, 2012)). L'exhaustivité de la première phase est souvent considérée comme un avantage crucial pour construire des modèles ou calculer des mesures de qualité. Cependant, la complétude de l'extraction de la première étape oblige à ajuster les seuils d'extractions, ce qui constitue une difficulté reconnue. Si les seuils sont trop bas, l'extraction devient irréalisable. Si les seuils sont trop élevés, certains motifs pourtant essentiels sont manqués. Enfin, l'exhaustivité conduit à d'énormes volumes de motifs. Pour un budget équivalent (en temps ou en nombre de motifs), les méthodes non-exhaustives peuvent produire des collections mieux adaptées à la construction du modèle de la seconde étape. De manière intéressante, une méthode non-exhaustive peut même garantir une certaine qualité sur la deuxième étape.

Cet article revisite le calcul du Frequent Pattern Outlier Factor (FPOF) en utilisant des méthodes non-exhaustives i.e., seul un échantillon est utilisé voire aucun motif. Nous proposons tout d'abord une méthode pour calculer le FPOF exact de chaque transaction. Etonnamment, notre méthode est non-énumérative dans le sens où aucun motif n'est généré (a fortiori, cela est également une méthode non-exhaustive). Pour cela, nous reformulons le FPOF en opérant directement sur les paires de transactions. Cette méthode calcule le FPOF en temps polynomial avec le nombre de transactions et le nombre d'items du jeu de données. Les expériences montrent que cette méthode parvient à calculer le FPOF exact sur des jeux de données où l'approche habituelle échoue. Ensuite, Nous proposons également une méthode approchée qui exploite un échantillon de motifs au lieu de la collection complète de motifs fréquents. En utilisant l'inégalité de Bennett, cette méthode sélectionne la taille de l'échantillon de manière à garantir une erreur maximale pour une confiance donnée. Les expériences montrent l'efficacité de cette méthode même avec une erreur maximale réduite.

2 Travaux relatifs

Dans cet article, nous nous concentrons sur les méthodes de détection des données aberrantes à base de motifs fréquents (He et al., 2005; Otey et al., 2006; Koufakou et al., 2011). Une vue plus large sur la détection de données aberrantes est, par exemple, disponible dans (Hawkins, 1980). Les méthodes fondées sur les motifs bénéficient des progrès de l'extraction de motifs réalisés au cours des deux dernières décennies. Ces méthodes ont un double intérêt. D'une part, elles sont bien adaptées pour gérer les données catégorielles contrairement à la plupart des autres méthodes dédiées aux données numériques. De plus, elles restent également opérationnelles pour les espaces de très grande dimension. La première approche (He et al., 2005) introduit le score d'aberration fondé sur les motifs fréquents (*frequent pattern outlier factor*, FPOF) qui exploite la collection complète des motifs fréquents. Otey et al. (2006) utilise une approche opposée en considérant des itemsets non fréquents. Plus récemment, Koufakou et al. (2011) a remplacé la collection de motifs fréquents par la représentation condensée des motifs non-dérivables (NDI) qui est plus compacte et moins coûteuse à extraire. Nous aimerions aller plus loin en montrant que le FPOF proposée dans (He et al., 2005) peut être calculé sans extraire le moindre motif ou en extrayant un échantillon très réduit.

Récemment, il y a eu une résurgence dans le domaine de l'extraction de motifs pour les méthodes non-exhaustives à travers l'échantillonnage de motifs (Boley et al., 2011). L'échantillonnage de motifs vise à accéder à l'espace de motif $\mathcal{L}$ par une procédure d'échantillonnage

$\mathcal{D}$

Trans.	Items		
t_1	A	B	
t_2	A	B	
t_3	A	B	
t_4			C

$\mathcal{D}'$

Trans.	Items		
t_1	A	B	
t_2	A	B	
t_3	A	B	
t_4			C
t_5	**A**	**B**	

$\mathcal{D}''$

Trans.	Items			
t_1	A	B		**D**
t_2	A	B		**D**
t_3	A	B		**D**
t_4			C	

TAB. 1 – *Trois jeux de données avec de légères variations*

efficace afin de simuler une distribution $\pi : \mathcal{L} \rightarrow [0, 1]$ qui est définie par rapport à une certaine mesure d'intérêt $m : \pi(.) = m(.)/Z$ où Z est une constante de normalisation (un cadre formel et des algorithmes sont, par exemple, détaillés dans (Boley et al., 2011)). De cette façon, l'utilisateur dispose d'un accès rapide et direct à l'ensemble du langage de motifs et ce sans paramètre (sauf éventuellement la taille de l'échantillon). L'échantillonnage de motifs a plutôt été introduit pour faciliter l'exploration de données interactive (van Leeuwen, 2014). Dans cet article, nous étudions l'utilisation de l'échantillonnage de motifs pour assigner le FPOF à chaque transaction. Avec un budget inférieur à celui d'une méthode exhaustive, on obtient une qualité finale élevée avec une erreur contrôlable.

3 Détecter des données aberrantes avec des motifs fréquents

3.1 Définitions

Soit $\mathcal{I}$ un ensemble de littéraux distincts appelés *items*, un itemset (ou un motif) est un sous-ensemble de $\mathcal{I}$. Le langage des itemsets correspond à $\mathcal{L} = 2^{\mathcal{I}}$. Un jeu de données transactionnel est un multi-ensemble d'itemsets de $\mathcal{L}$. Chaque observation de ce jeu de données est appelée *transaction*. Par exemple, le tableau 1 donne trois exemples de jeux de données transactionnels comportant 4 à 5 transactions t_i décrites avec 4 items A, B, C et D. La découverte de motifs tire avantage de mesures d'intérêt afin d'évaluer la pertinence des motifs. Par exemple, le *support* d'un motif X dans le jeu de données $\mathcal{D}$ est la proportion de transactions couvertes par X (Agrawal et al., 1994) : $supp(X, \mathcal{D}) = |\{t \in \mathcal{D} : X \subseteq t\}|/|\mathcal{D}|$. Un motif est dit *fréquent* quand son support excède un seuil minimal défini par l'utilisateur. L'ensemble de tous les motifs fréquents pour un seuil σ dans $\mathcal{D}$ est noté par $\mathcal{F}_\sigma(\mathcal{D})$: $\mathcal{F}_\sigma(\mathcal{D}) = \{X \in \mathcal{L} : supp(X, \mathcal{D}) \geq \sigma\}$.

Par la suite, nous manipulons des multi-ensembles de motifs qui sont des collections de motifs admettant plusieurs occurrences d'un même motif. La représentativité d'un multi-ensemble de motifs $\mathcal{P}$, dénotée $Supp(\mathcal{P}, \mathcal{D})$, est la somme des supports de chacun des motifs de $\mathcal{P}$: $Supp(\mathcal{P}, \mathcal{D}) = \sum_{X \in \mathcal{P}} supp(X, \mathcal{D})$. Le domaine de variation de $Supp(\mathcal{P}, \mathcal{D})$ est $[0, |\mathcal{P}|]$. Une représentativité élevée pour un multi-ensemble de motifs de cardinalité fixée signifie qu'il contient des motifs très communs au sein du jeu de données. Pour comparer le contenu de deux multi-ensembles de motifs, nous utilisons la semi-jointure, dénotée $\mathcal{P}_2 \triangleright \mathcal{P}_1$, qui retourne tous les motifs de $\mathcal{P}_2$ apparaissant dans $\mathcal{P}_1$: $\mathcal{P}_2 \triangleright \mathcal{P}_1 = \{X \in \mathcal{P}_2 : X \in \mathcal{P}_1\}$. Par exemple, $\{A, AB, A, D\} \triangleright \{C, A, B\} = \{A, A\}$.

3.2 Frequent Pattern Outlier Factor

Intuitivement, une transaction est plus représentative d'un jeu de données lorsqu'elle contient de nombreux motifs très fréquents au sein de ce même jeu de données. A l'inverse, une donnée aberrante contient seulement quelques motifs dont la fréquence est plutôt basse. Le *frequent pattern outlier factor* formalise cette intuition sous forme d'une mesure :

Définition 1 (FPOF) *Le frequent pattern outlier factor d'une transaction t dans $\mathcal{D}$ est défini de la manière suivante :*

$$\mathit{fpof}(t, \mathcal{D}) = \frac{Supp(2^t, \mathcal{D})}{\max_{u \in \mathcal{D}} Supp(2^u, \mathcal{D})}$$

Le domaine de variation du *fpof* est $[0, 1]$ où 1 signifie que la transaction est la plus représentative du jeu de données tandis que 0 signifie que la transaction est une donnée aberrante. D'autres normalisations sont possibles en modifiant le dénominateur comme $Supp(\mathcal{L}, \mathcal{D})$ ou $\sum_{t \in \mathcal{D}} Supp(2^t, \mathcal{D})$. Quel que soit la normalisation choisie, deux transactions restent ordonnées de manière similaire (cela n'affecte donc pas le tau de Kendall que nous utilisons dans la suite pour évaluer notre méthode). A noter qu'avec une modélisation markovienne de l'analyste, le score $\mathit{fpof}(t, \mathcal{D})$ correspond à la proportion de temps qu'un analyste dédiera à la transaction t suite à l'étude des motifs (Giacometti et al., 2014).

Dans le premier jeu de données du tableau 1, t_1 est couverte par $\emptyset$ et A, B, AB dont le support est égal à 0.75 ($Supp(\{\emptyset, A, B, AB\}, \mathcal{D}) = 3.25$) tandis que t_4 est seulement couverte par $\emptyset$ et C dont le support est 0.25. Ainsi, $\mathit{fpof}(t_1, \mathcal{D}_1) = 3.25/3.25$ et $\mathit{fpof}(t_4, \mathcal{D}_1) = 1.25/3.25$. Dans cet exemple, t_4 semble être une donnée aberrante. Il est facile de voir que l'augmentation de la fréquence des motifs couvrant les premières transactions (e.g., avec $\mathcal{D}'$) décroît encore le FPOF de t_4. De manière similaire, l'augmentation du nombre de motifs couvrant les premières transactions décroît aussi le FPOF de t_4 (e.g., avec $\mathcal{D}''$).

3.3 Formulation du problème

Dans notre contexte, la détection de données aberrantes consiste à calculer le FPOF de chaque transaction :

Problème 1 (Problème exact) *Etant donné un jeu de données $\mathcal{D}$, calculer le FPOF de chaque transaction $t \in \mathcal{D}$.*

En pratique, le calcul exact du FPOF est réalisé en extrayant tous les motifs apparaissant au moins une fois dans le jeu de données (i.e., avec $\sigma = 1/|\mathcal{D}|$). Bien sûr, cette tâche coûteuse n'est pas faisable pour les jeux de données très volumineux. A la place, le FPOF est approché avec la collection des motifs fréquents correspondant à un seuil minimal de support supérieur à $1/|\mathcal{D}|$:

Définition 2 (σ-exhaustif FPOF) *Etant donné un seuil minimal de support σ, le σ-exhaustif FPOF d'une transaction t de $\mathcal{D}$ est défini de la manière suivante :*

$$\mathit{fpof}_\sigma(t, \mathcal{D}) = \frac{Supp(\mathcal{F}_\sigma(\mathcal{D}) \triangleright 2^t, \mathcal{D})}{\max_{u \in \mathcal{D}} Supp(\mathcal{F}_\sigma(\mathcal{D}) \triangleright 2^u, \mathcal{D})}$$

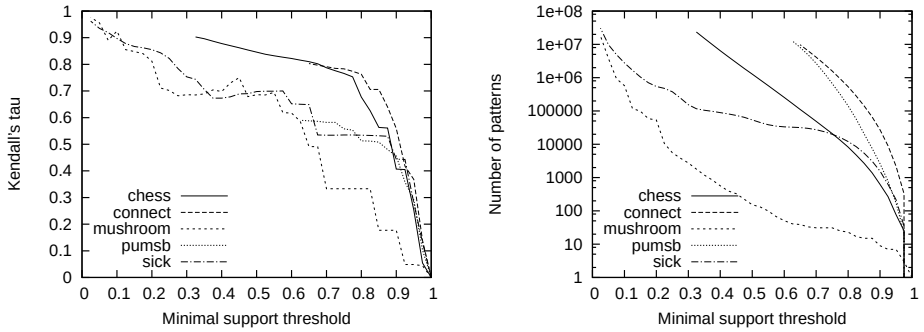

FIG. 1 – *Le tau de Kendall et le nombre de motifs pour un seuil minimal de support*

L'approximation devient précise lorsque le seuil minimal de support devient très bas. La figure 1 trace sur la partie gauche, le tau de Kendall de $fpof_\sigma$ comparé avec le *fpof* pour plusieurs jeux de données de l'UCI [1]. Malheureusement, quand le seuil minimal de support devient très faible, le nombre de motifs (cf. la partie droite de la figure 1) et le temps d'extraction explose. Par ailleurs, une telle approche ne donne aucune estimation de l'erreur commise. Avec un budget moindre (en nombre de motifs ou en temps), nous affirmons qu'il est possible d'obtenir une meilleure approximation du FPOF tout en ayant une borne maximale sur l'erreur.

Le tau de Kendall varie irrégulièrement selon le jeu de données pour un même seuil de support minimal. Il n'est donc pas aisé de fixer ce seuil afin d'obtenir une bonne efficacité et une bonne qualité. Pour cette raison, il nous paraît intéressant que l'utilisateur fixe l'erreur maximale qu'il tolère sur le résultat plutôt qu'un seuil lié à la méthode de calcul :

Problème 2 (Problème approché) *Etant donné un jeu de données $\mathcal{D}$, deux réels δ et ϵ, trouver une fonction $\widetilde{fpof}$ approximant le FPOF tel que pour chaque transaction $t \in \mathcal{D}$ $|fpof(t,\mathcal{D}) - \widetilde{fpof}(t,\mathcal{D})| \leq \epsilon$ avec une confiance $1-\delta$.*

Le problème 1 est un cas particulier du problème 2 en fixant $\epsilon = 0$ et $\delta = 0$.

4 Méthode non-énumérative exacte

Cette section traite le problème 1. Pour calculer le FPOF d'une transaction t, la définition 1 formule la représentativité en terme de support des motifs apparaissant dans t. L'idée est de reformuler cette mesure en considérant ce que chaque transaction u apporte à la transaction t. Par exemple, dans le jeu de données $\mathcal{D}$, le FPOF de la première transaction repose sur $Supp(\{\emptyset, A, B, AB\}, \mathcal{D})$ qui est égal à $|\{\emptyset, A, B, AB,\ \emptyset, A, B, AB,\ \emptyset, A, B, AB,\ \emptyset\}|/4$. Chaque sous-ensemble $\{\emptyset, A, B, AB\}$ résulte de l'intersection des motifs couvrant t_1 avec ceux couvrant une autre transaction t_i où $i \in \{1, 2, 3\}$. De cette manière,

1. Il s'agit de la proportion de paires de transactions qui sont ordonnées de la même manière avec le FPOF approché et le FPOF exact (cf. la section 6 pour une définition formelle).

Algorithm 1 Méthode non-énumérative exacte

Input: Un jeu de données $\mathcal{D}$
Output: FPOF calculé pour chaque transaction
1: Initialiser $\mathit{fpof}[t] \leftarrow 0$ pour $t \in \mathcal{D}$
2: **for all** $(t, u) \in \mathcal{D} \times \mathcal{D}$ **do**
3: $\quad \mathit{fpof}[t] \leftarrow \mathit{fpof}[t] + 2^{|t \cap u|}$
4: **end for**
5: Normaliser $\mathit{fpof}[t] \leftarrow \mathit{fpof}[t]/Z$ pour $t \in \mathcal{D}$ où $Z = \max_{u \in \mathcal{D}} \mathit{fpof}[u]$
6: **return** *fpof*

$Supp(\{\emptyset, A, B, AB\}, \mathcal{D}) = |\{\bigcup_{i \in \{1,\ldots,4\}} 2^{t_1} \cap 2^{t_i}\}|/|\mathcal{D}| = |\{\bigcup_{i \in \{1,\ldots,4\}} 2^{t_1 \cap t_i}\}|/|\mathcal{D}|$. Cette observation est au coeur de la propriété ci-dessous :

Propriété 1 (Reformulation) *Etant donné un jeu de données $\mathcal{D}$, le FPOF peut être reformulé de la manière suivante pour chaque transaction $t \in \mathcal{D}$:*

$$\mathit{fpof}(t, \mathcal{D}) = \frac{\sum_{u \in \mathcal{D}} 2^{|t \cap u|}}{\max_{v \in \mathcal{D}} \sum_{u \in \mathcal{D}} 2^{|v \cap u|}}$$

Preuve. Soit $\mathcal{D}$ un jeu de données. Etant donné $u \in \mathcal{D}$, nous définissons $\kappa(X, u) = 1$ si $X \subseteq u$ et 0 sinon. Pour chaque transaction $t \in \mathcal{D}$, nous obtenons :

$$\begin{aligned} Supp(2^t, \mathcal{D}) &= \frac{1}{|\mathcal{D}|} \sum_{X \subseteq t} \left(\sum_{\{u \in \mathcal{D} : X \subseteq u\}} 1 \right) = \frac{1}{|\mathcal{D}|} \sum_{X \subseteq t} \sum_{u \in \mathcal{D}} \kappa(X, u) = \frac{1}{|\mathcal{D}|} \sum_{u \in \mathcal{D}} \sum_{X \subseteq t} \kappa(X, u) \\ &= \frac{1}{|\mathcal{D}|} \sum_{u \in \mathcal{D}} \left(\sum_{X \subseteq t \wedge X \subseteq u} 1 \right) = \frac{1}{|\mathcal{D}|} \sum_{u \in \mathcal{D}} \left(\sum_{X \subseteq t \cap u} 1 \right) = \frac{1}{|\mathcal{D}|} \sum_{u \in \mathcal{D}} 2^{|t \cap u|} \end{aligned}$$

En injectant cette équation dans la définition 1, on prouve que la propriété 1 est correcte. □

D'un point de vue conceptuel, il est intéressant de noter que finalement, le FPOF d'une transaction est juste la somme de sa similarité avec chacune des transactions (où la similarité entre t et u est $2^{|t \cap u|}$). Cette mesure est donc assez proche des méthodes traditionnelles utilisant une distance entre les paires de données.

De manière intéressante, la formule du FPOF de la propriété 1 peut être calculée simplement avec une double boucle (cf. l'algorithme 1). Par conséquent, le premier résultat important de cet article est de montrer que le problème 1 peut être résolu en temps polynomial contrairement à ce qui avait été envisagé dans la littérature :

Propriété 2 (Complexité) *Le FPOF de toutes les transactions peut être calculé en temps $O(|\mathcal{D}|^2 \times |\mathcal{I}|)$.*

Par manque de place, cette preuve et plusieurs autres sont omises. A notre connaissance, notre proposition est la première méthode à calculer le FPOF en temps polynomial. Néanmoins, pour les jeux de données volumineux avec notamment beaucoup de transactions, cette complexité reste très élevée. Cela fait alors sens de recourir à des méthodes approchées dont le résultat sera obtenu bien plus rapidement.

5 Méthode non-exhaustive ϵ-approchée

Cette section traite le problème 2 en exploitant l'échantillonnage de motifs. Pour commencer, nous proposons une méthode pour approximer le FPOF à partir d'un échantillon de motifs tiré selon le support. Nous montrons alors comment choisir la taille de l'échantillon de manière à contrôler l'erreur maximale.

5.1 Echantillonnage de motifs pour le FPOF

Dans la section 3.3, nous avons vu que l'utilisation des motifs les plus fréquents est insuffisante pour approximer le FPOF avec précision car ils ne mesurent pas la singularité de chaque transaction qui repose aussi sur des motifs plus spécifiques (dont le support varie de faible à moyen). Inversement, ne pas considérer les motifs les plus fréquents serait aussi une erreur car ils contribuent significativement au FPOF. Une approche raisonnable est de sélectionner des motifs aléatoirement avec une probabilité proportionnelle à leur poids dans le calcul du FPOF. Typiquement, dans le jeu de données $\mathcal{D}$ du tableau 1, l'itemset AB est 3 fois plus important que l'itemset C dans le calcul du FPOF à cause de leur fréquence respective.

Ces dernières années, des techniques d'échantillonnage ont été proposées pour tirer aléatoirement des motifs proportionnellement à leur fréquence (Boley et al., 2011). De telles approches sont idéales pour nous apporter une collection adaptée de motifs. Bien sûr, il reste la tâche non-triviale d'approximer le FPOF en partant de cette collection :

Définition 3 (FPOF k-échantillonné) *Etant donné un entier $k > 0$, un FPOF k-échantillonné d'une transaction t de $\mathcal{D}$ est défini de la manière suivante :*

$$\mathit{fpof}_k(t, \mathcal{D}) = \frac{|\mathcal{S}_k(\mathcal{D}) \triangleright 2^t|}{\max_{u \in \mathcal{D}} |\mathcal{S}_k(\mathcal{D}) \triangleright 2^u|}$$

où $\mathcal{S}_k(\mathcal{D})$ est un échantillon de k motifs tirés selon le support : $\mathcal{S}_k(\mathcal{D}) \sim \mathit{supp}(\mathcal{L}, \mathcal{D})$.

Il est important de noter que $|.|$ est utilisé ici à la place de $Supp(., \mathcal{D})$ comme c'est le cas dans la définition 1. Comme la technique d'échantillonnage tient déjà compte de la fréquence quand elle tire des motifs, il n'est pas nécessaire d'impliquer le support à nouveau. En effet, le tirage est considéré avec remise pour que l'approximation du FPOF soit correcte (sans cette remise les motifs les plus fréquents seraient désavantagés). Ce tirage avec remise induit qu'un même motif peut avoir plusieurs occurrences au sein de l'échantillon $\mathcal{S}_k(\mathcal{D})$.

Pour une même taille d'échantillon k et pour une même transaction t, il est possible de calculer différentes valeurs du FPOF k-échantillonné à cause de l'échantillon $\mathcal{S}_k(\mathcal{D})$ qui varie. Mais, plus le seuil k sera élevé, moins l'écart entre ces différentes valeurs issues d'échantillons différents sera élevé. Par ailleurs, plus la taille de l'échantillon est grande, meilleure est l'approximation :

Propriété 3 (Convergence) *Etant donné un jeu de données $\mathcal{D}$, un FPOF k-échantillonné converge vers le FPOF pour toutes les transactions $t \in \mathcal{D}$.*

Preuve. $\mathcal{S}_k(\mathcal{D}) \sim \mathit{supp}(\mathcal{L}, \mathcal{D})$ signifie qu'il existe une constante $\alpha > 0$ tel que $\forall X \in \mathcal{L}$, $\lim_{k \to \infty} |\mathcal{S}_k(\mathcal{D}) \triangleright \{X\}| = \alpha \mathit{supp}(X, \mathcal{D})$. Ensuite, pour chaque transaction t, nous obtenons

Algorithm 2 Méthode non-exhaustive ϵ-approchée

Input: Un jeu de données $\mathcal{D}$, une confiance $1-\delta$, une borne ϵ
Output: Un FPOF k-échantillonné de toutes les transactions de $\mathcal{D}$ dont l'erreur est bornée par ϵ avec une confiance $1-\delta$

1: $\tilde{\epsilon} \leftarrow 1$; $\mathcal{S} \leftarrow \emptyset$
2: **while** $\tilde{\epsilon} > \epsilon$ **do**
3: $\quad \mathcal{S} \leftarrow \mathcal{S} \cup \{X\}$ où $X \sim supp(\mathcal{L}, \mathcal{D})$ *// ajouter un motif dans l'échantillon*
4: $\quad m \leftarrow \arg\max_{u \in \mathcal{D}} cov_{\mathcal{S}}(u)$ *// sélectionner la transaction de plus forte représentativité*
// estimer l'erreur maximale sur $cov_{\mathcal{S}}$
5: $\quad e_t \leftarrow \sqrt{2\overline{\sigma_t}\ln(1/\delta)/|\mathcal{S}|} + \ln(1/\delta)/(3|\mathcal{S}|)$ pour tout $t \in \mathcal{D}$
// estimer l'erreur maximale sur le FPOF
6: $\quad \tilde{\epsilon} \leftarrow \max_{t \in \mathcal{D}}\{\min\{1; (cov_{\mathcal{S}}(t) + e_t)/(cov_{\mathcal{S}}(m) - e_m)\} - cov_{\mathcal{S}}(t)/cov_{\mathcal{S}}(m)\}$
7: $\quad \tilde{\epsilon} \leftarrow \max_{t \in \mathcal{D}}\{cov_{\mathcal{S}}(t)/cov_{\mathcal{S}}(m) - \max\{0; (cov_{\mathcal{S}}(t) - e_t)/(cov_{\mathcal{S}}(m) + e_m)\}; \tilde{\epsilon}\}$
8: **od**
9: **return** $\langle cov_{\mathcal{S}}(t)/\max_{u \in \mathcal{D}} cov_{\mathcal{S}}(u)\rangle_{t \in \mathcal{D}}$

que : $\lim_{k\to\infty} |\mathcal{S}_k(\mathcal{D}) \triangleright 2^t| = \alpha \sum_{X \in 2^t} supp(X, \mathcal{D}) = \alpha Supp(2^t, \mathcal{D})$. En injectant ce résultat dans la définition 3, nous concluons que la propriété 3 est correcte. □

Au-delà de la convergence, l'intérêt de cette approche est la rapidité de sa convergence bien supérieure à celle du FPOF σ-exhaustif comme le montre l'étude expérimentale (cf. Section 6). Cette rapidité s'accompagne d'une bonne efficacité pratique grâce à la complexité raisonnable de l'échantillonnage de motifs :

Propriété 4 (Complexité) *Un FPOF k-échantillonné est calculable en temps $O(k \times |\mathcal{I}| \times |\mathcal{D}|)$.*

Etant donné un nombre de motifs k, un FPOF k-échantillonné est ainsi efficace à calculer. Cependant, le choix de ce seuil k est à la fois difficile et essentiel afin d'obtenir l'approximation désirée comme suggéré par le problème 2. La section suivante présente une méthode itérative pour éviter le choix de ce seuil par l'utilisateur.

5.2 Majoration de l'erreur

Cette section montre comment bien déterminer une taille k d'échantillon pour calculer un FPOF k-échantillonné satisfaisant les paramètres spécifiés par l'utilisateur (erreur maximale pour une confiance donnée). L'idée est de tirer un échantillon de motifs et d'estimer l'erreur maximale pour le FPOF en utilisant un résultat statistique appelé l'inégalité de Bennett. Si l'erreur calculée est inférieure à celle souhaitée par l'utilisateur, l'algorithme retourne un FPOF échantillonné grâce à l'échantillon courant. Sinon, l'algorithme augmente la taille de l'échantillon en tirant un nouveau motif et ainsi de suite.

Nous utilisons l'inégalité de Bennett pour estimer l'erreur courante d'un échantillon car ce résultat statistique est vrai indépendamment de toute distribution. Après k observations indépendantes d'une variable aléatoire r comprise dans $[0, 1]$, l'inégalité de Bennett nous garantit qu'avec une confiance $1-\delta$, la vraie moyenne de r est au moins $\overline{r} - \epsilon$ où $\overline{r}$ et $\overline{\sigma}$ sont respectivement la moyenne et la variance observée avec l'échantillon et

$$\epsilon = \sqrt{\frac{2\overline{\sigma}\ln(1/\delta)}{k}} + \frac{\ln(1/\delta)}{3k}$$

Dans notre cas, la variable aléatoire est le nombre moyen de motifs contenus dans l'échantillon $\mathcal{S}_k \sim supp(\mathcal{L}, \mathcal{D})$ qui couvre la transaction t. Elle est notée $cov_{\mathcal{S}_k}(t)$ et définie formellement ainsi : $cov_{\mathcal{S}_k}(t) = |\mathcal{S}_k \triangleright 2^t|/k$. Il est facile de réécrire le FPOF k-échantillonné à partir de $cov_{\mathcal{S}_k}$: $fpof_k(t, \mathcal{D}) = cov_{\mathcal{S}_k}(t)/\max_{u \in \mathcal{D}} cov_{\mathcal{S}_k}(u)$. L'inégalité de Bennett avec cette définition nous permet de borner le FPOF exact :

Propriété 5 (Bornes sur le FPOF k-échantillonné) *Etant donné $\mathcal{D}$ et la confiance $1-\delta$, le FPOF k-échantillonné de la transaction t est borné de la manière suivante :*

$$\max\left\{0, \frac{cov_{\mathcal{S}_k}(t) - \epsilon_t}{cov_{\mathcal{S}_k}(u) + \epsilon_u}\right\} \leq fpof(t, \mathcal{D}) \leq \min\left\{\frac{cov_{\mathcal{S}_k}(t) + \epsilon_t}{cov_{\mathcal{S}_k}(u) - \epsilon_u}, 1\right\}$$

où $\mathcal{S}_k \sim supp(\mathcal{L}, \mathcal{D})$, $u = \arg\max_{v \in \mathcal{D}} cov_{\mathcal{S}_k}(v)$ et $\epsilon_t = \sqrt{2\overline{\sigma_t}\ln(1/\delta)/k} + \ln(1/\delta)/(3k)$.

L'algorithme 2 retourne le FPOF approché de chaque transaction en garantissant une erreur maximale inférieure à ϵ avec une confiance $1-\delta$. La boucle principale itère jusqu'à ce que l'erreur maximale estimée $\tilde{\epsilon}$ soit inférieure au seuil souhaité ϵ. Les lignes 4-7 calculent cette erreur maximale $\tilde{\epsilon}$ en utilisant la propriété 5. Si l'erreur maximale est en dessous de ϵ, la ligne 9 retourne le FPOF k-échantillonné avec l'échantillon courant $\mathcal{S}$. Sinon, un motif supplémentaire est tiré (ligne 3). Comme désiré par le problème 2, l'algorithme 2 contrôle l'approximation du FPOF de l'ensemble des transactions :

Propriété 6 (Justesse) *Etant donné un jeu de données $\mathcal{D}$, une confiance $1-\delta$, une borne ϵ, l'algorithme 2 retourne un FPOF k-échantillonné de chaque transaction t de $\mathcal{D}$ approximant le FPOF avec une erreur majorée par ϵ avec une confiance $1-\delta$.*

6 Etude expérimentale

Cette étude expérimentale a pour objectif de comparer la vitesse de notre méthode non-énumérative exacte avec la méthode exhaustive exacte, mais aussi d'estimer la qualité de notre méthode ϵ-approchée non-exhaustive face à la méthode σ-exhaustive. Par manque de place, nous ne considérons pas de nouvelles expériences montrant l'intérêt du FPOF pour détecter les données aberrantes comme cet aspect a déjà été largement détaillé dans la littérature (He et al., 2005; Koufakou et al., 2011). Les expérimentations ont été conduites sur des jeux de données provenant de l'UCI Machine Learning repository et du FIMI repository. Le tableau 2 donne des caractéristiques des jeux de données dans les premières colonnes. Les expérimentations ont été réalisées sous Linux sur un processeur Xeon 2.5 GHz et 2 GB de mémoire RAM. Chaque mesure reportée est la moyenne arithmétique de 10 mesures répétées.

6.1 Méthode non-énumérative exacte

Le tableau 2 reporte les temps d'exécution requis pour calculer le FPOF exact en utilisant la méthode exhaustive (i.e., $1/|\mathcal{D}|$-exhaustive FPOF), et nos méthodes non-enumérative exacte et approchée (colonnes 4 à 6). Il est important de noter que la méthode exhaustive bénéficie de l'implémentation originale de LCM qui est particulièrement reconnue (cf. le FIMI repository). La méthode non-enumérative exacte est efficace et rivalise avec l'approche exhaustive. Son

$\mathcal{D}$	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{I}\rvert$	Exh. (s)	Non-enum. (s)	0.1-Approchée. (s)
chess	3,196	75	439.5	1.1	**0.3**
connect	67,557	129	748.5	**577.7**	**176.6**
mushroom	8,124	119	**0.4**	5.9	2.0
pumsb	49,096	7,117	time out	**1,970.5**	**175.0**
retail	88,162	16,470	**8.7**	5,969.9	**1.3**
sick	2,800	58	0.8	**0.5**	**0.5**

TAB. 2 – *Temps d'exécution des méthodes exactes et de la méthode 0.1-approchée*

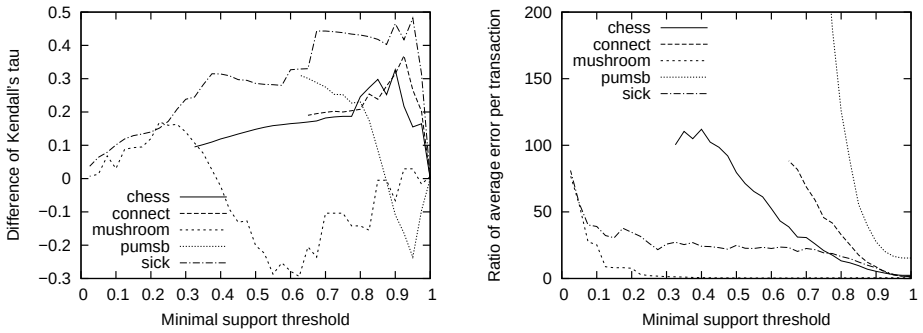

FIG. 2 – *L'erreur du FPOF σ-exhaustif et du FPOF k-échantillonné par rapport à σ*

atout principal est de garantir le calcul du FPOF exact même avec les jeux de données où la méthode exhaustive échoue (e.g., pumsb où l'exécution a été arrêtée après 5h). En particulier, la méthode exacte non-énumérative est très efficace sur les jeux de données denses car sa complexité est indépendante du nombre de motifs.

6.2 Méthode non-exhaustive ϵ-approchée

Cette section compare notre méthode d'échantillonnage pour approximer le FPOF avec l'approche σ-exhaustive comme référence. Pour cela, nous utilisons le tau de Kendall pour comparer l'ordre issu de chaque méthode approchée f avec le vrai ordre issu du FPOF (calculé grâce à une méthode exacte) :

$$\tau(f,\mathcal{D}) = \frac{|\{(t,u) \in \mathcal{D}^2 : sgn(f(t,\mathcal{D}) - f(u,\mathcal{D})) = sgn(\mathit{fpof}(t,\mathcal{D}) - \mathit{fpof}(u,\mathcal{D})\}|}{|\mathcal{D}|^2}$$

et de la même manière, nous avons aussi calculé l'erreur moyenne par transaction : $\varepsilon(f,\mathcal{D}) = \sum_{t\in\mathcal{D}} |f(t,\mathcal{D}) - \mathit{fpof}(t,\mathcal{D})|/|\mathcal{D}|$.

La partie gauche de la figure 2 trace la différence entre le tau de Kendall du FPOF k-échantillonné et celui du FPOF σ-exhaustif (quand une courbe est au-dessus de 0, cela signifie que l'ordre du FPOF k-échantillonné est meilleur que celui du FPOF σ-exhaustif). La partie droite reporte l'erreur moyenne du FPOF k-échantillonné divisée par celle du FPOF σ-exhaustif (quand la courbe est au-dessus de 1, cela signifie que l'erreur moyenne du FPOF

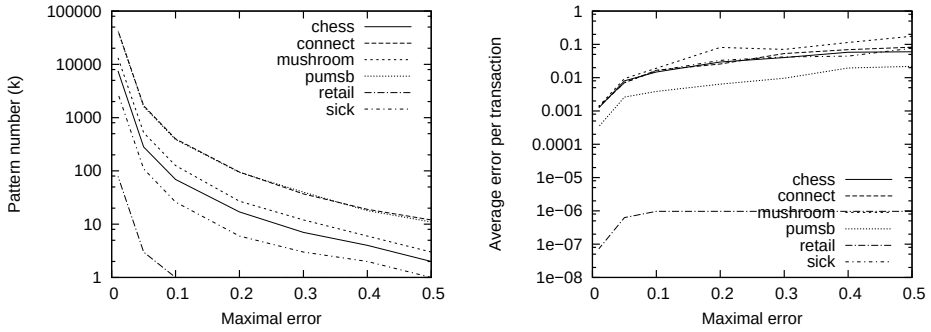

FIG. 3 – *Nombre de motifs et erreur réelle selon l'erreur maximale* ϵ

k-échantillonné est plus petite que celle du FPOF σ-exhaustif). Pour chaque point, le seuil minimal de support σ est utilisé comme paramètre de la méthode σ-exhaustive. En même temps, la taille de l'échantillon k est fixée par le nombre de motifs extraits correspondant au seuil de support σ : $k = |\mathcal{F}_\sigma(\mathcal{D})|$. Le FPOF k-échantillonné est clairement plus précis que le FPOF σ-exhaustif pour un même budget en motifs quand celui-ci augmente (i.e., diminution du seuil minimal de support). Pour certains jeux de données (e.g., `chess` et `sick`), la différence est même toujours positive. Le ratio des erreurs moyennes montre que notre approche par échantillonnage est une meilleure approximation du FPOF (notamment quand le nombre de motifs devient grand).

La figure 3 trace le nombre de motifs et l'erreur réelle de la méthode ϵ-approchée en variant la borne ϵ (pour $\delta = 0.1$). Comme attendu, plus l'erreur autorisée ϵ est petite, plus le nombre de motifs requis dans l'échantillon est élevé. Par conséquent, plus long est le temps d'exécution. Il est intéressant de noter que notre méthode 0.1-approchée est souvent plus rapide que les méthodes exactes (cf. le tableau 2). Enfin, l'erreur moyenne réelle par transaction de la méthode approchée est toujours très inférieure à celle souhaitée ϵ (par exemple, $\epsilon = 0.1$ donne une erreur réelle inférieure à 0.01). Cet écart résulte de l'inégalité de Bennett qui ne fait aucune hypothèse sur la distribution.

7 Conclusion

Nous avons revisité le calcul du FPOF en extrayant le moins possible de motifs voire aucun. Malgré cette contrainte, notre proposition de méthode exacte a une complexité mieux adaptée à certains grands jeux de données. Notre méthode approchée utilisant une technique d'échantillonnage apporte des garanties supplémentaires sur le résultat en bornant l'erreur. Les expérimentations ont montré l'intérêt de ces deux approches en terme de rapidité et de précision par rapport à la méthode traditionnelle où un maximum de motifs sont extraits.

Grâce à l'échantillonnage, notre proposition combine donc la puissance éprouvée des méthodes fondées sur les motifs en ajoutant une garantie supplémentaire sur la qualité du résultat sans sacrifier la vitesse. Nous pensons qu'elle peut être étendue à d'autres mesures impliquant des motifs ou des modèles composés de motifs. Nous voudrions aussi adapter notre approche

pour construire des algorithmes anytime. Dans le cas du FPOF, cela consiste à étendre l'échantillon de motifs indéfiniment jusqu'à ce que l'utilisateur final souhaite interrompre le processus. L'algorithme retourne alors le FPOF obtenu avec l'échantillon courant et l'erreur estimée.

Remerciements. Ce travail a été partiellement soutenu par le CNRS, PEPS 2015, projet Préfute.

Références

Agrawal, R., R. Srikant, et al. (1994). Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, Volume 1215, pp. 487–499.

Boley, M., C. Lucchese, D. Paurat, et T. Gärtner (2011). Direct local pattern sampling by efficient two-step random procedures. In *Proc. of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 582–590.

Giacometti, A., D. H. Li, et A. Soulet (2014). Balancing the analysis of frequent patterns. In *Advances in Knowledge Discovery and Data Mining*, pp. 53–64. Springer.

Hawkins, D. M. (1980). *Identification of outliers*, Volume 11. Springer.

He, Z., X. Xu, Z. J. Huang, et S. Deng (2005). FP-outlier : Frequent pattern based outlier detection. *Computer Science and Information Systems 2*(1), 103–118.

Knobbe, A., B. Crémilleux, J. Fürnkranz, et M. Scholz (2008). From local patterns to global models : The lego approach to data mining. In *From local patterns to global models : proceedings of the ECML PKDD 2008 Workshop*, pp. 1–16.

Koufakou, A., J. Secretan, et M. Georgiopoulos (2011). Non-derivable itemsets for fast outlier detection in large high-dimensional categorical data. *Knowledge and information systems 29*(3), 697–725.

Liu, B., W. Hsu, et Y. Ma (1998). Integrating classification and association rule mining. In *Proc. of the 4th international conference on Knowledge Discovery and Data mining*.

Liu, Q. et G. Dong (2012). CPCQ : contrast pattern based clustering quality index for categorical data. *Pattern Recognition 45*(4), 1739–1748.

Otey, M. E., A. Ghoting, et S. Parthasarathy (2006). Fast distributed outlier detection in mixed-attribute data sets. *Data Mining and Knowledge Discovery 12*(2-3), 203–228.

van Leeuwen, M. (2014). Interactive data exploration using pattern mining. In *Interactive Knowledge Discovery and Data Mining in Biomedical Informatics*, pp. 169–182. Springer.

Summary

Outlier detection consists in detecting anomalous observations. Recently, outlier detection methods have proposed to mine all frequent patterns in order to compute the outlier factor of each transaction. This paper provides exact and approximate methods for calculating the frequent pattern outlier factor without exhaustive enumeration. We propose an algorithm that returns the exact FPOF without mining any pattern. We also present an approximate method where the user controls the maximum error on the estimated FPOF. A study shows the interest of both methods for large datasets where exhaustive mining fails to provide the exact solution. The accuracy of our approximate method outperforms the baseline approach.