Une distance hiérarchique basée sur la sémantique pour la comparaison d'histogrammes nominaux

Camile Kurtz*

*Université de Strasbourg, LSIIT
ckurtz@unistra.fr

Résumé. La plupart des distances entre histogrammes sont définies pour comparer des histogrammes ordonnés (dont les entités représentées sont totalement ordonnées) ou des histogrammes nominaux (dont les entités représentées ne peuvent pas être comparées). Cependant, il n'existe aucune distance qui permette de comparer des histogrammes nominaux dans lesquels il est possible de quantifier des valeurs de proximité sémantique entre les entités considérées. Cet article propose une nouvelle distance permettant de pallier ce problème. Dans un premier temps, une hiérarchie d'histogrammes, obtenue par le biais d'une fusion progressive des entités considérées (prenant en compte leurs proximités sémantiques), est construite. Pour chaque étage de cette hiérarchie, une distance standard de comparaison d'histogrammes nominaux est calculée. Finalement, pour obtenir la distance proposée, ces différentes distances sont fusionnées en prenant en compte la cohérence sémantique associée aux niveaux de chaque étage de la hiérarchie. Cette distance a été validée dans le cadre de la classification de données géographiques. Les résultats obtenus sont encourageants et montrent ainsi l'intérêt et l'utilité de cette dernière pour des processus de fouille de données.

1 Introduction

Contexte Un histogramme représente la distribution des valeurs quantifiées d'une mesure parmi les valeurs des éléments d'un ensemble. Un tel ensemble peut regrouper, par exemple, les résultats d'une expérience, ou encore une population d'individus. Dans de nombreux domaines, comme celui de la fouille de données, il est nécessaire de classifier d'importants jeux de données dans lesquels chaque donnée est caractérisée par un ou plusieurs histogrammes. Par exemple, il est souvent nécessaire de classifier des populations en fonction de la distribution d'une mesure particulière (*e.g.*, la distribution de la taille des individus contenus dans ces populations). Les histogrammes sont donc des structures utiles pour modéliser de nombreux types de données et permettent de prendre en considération leurs propriétés statistiques.

Il existe différents types d'histogrammes relatifs à des types de mesure spécifiques : nominales, ordonnées et modulo (Cha et Srihari, 2002). Dans une mesure nominale, chaque valeur est nommée et/ou peut représenter une instance d'un concept sémantique particulier (*e.g.*, le concept FRUIT peut prendre différentes valeurs/instances comme Citron, Prune, Pomme, Clémentine, Abricot, *etc.*). Ainsi, un histogramme de type nominal peut modéliser la composition d'un panier de courses en fonction du nombre et des types de fruits qu'il contient. Dans un tel

histogramme, les niveaux de la mesure peuvent être permutés car il n'existe pas d'ordre (total) entre ces niveaux (propriété d'invariance à la permutation). Au contraire, dans une mesure ordonnée, les valeurs sont totalement ordonnées (*e.g.*, le prix des légumes peut être quantifié en 10 valeurs discrètes entre 1 et 10 euros). Ainsi, un histogramme de type ordonné peut modéliser la composition d'un panier de courses en fonction des prix des articles. Finalement, dans une mesure modulo, les valeurs forment un anneau dû à l'opération arithmétique modulo (*e.g.*, l'aiguille des minutes d'une montre peut prendre 60 valeurs). Comme les valeurs de mesures de type modulo peuvent aussi être ordonnées, nous considérons que les histogrammes de type modulo sont des cas particuliers des histogrammes de type ordonné.

Mesurer la distance entre histogrammes est une opération cruciale dans de nombreux domaines comme la classification (Jain et al., 1999), la reconnaissance de motifs (Duda et al., 2000), la catégorisation de textes (Fabrizio, 2002), *etc.* En effet, une telle distance permet d'évaluer la similarité entre les propriétés statistiques des données représentées. Depuis ces dernières décennies, de nombreuses mesures de similarité entre histogrammes ont été proposées. Celles-ci peuvent être divisées en deux catégories : les distances *barre-à-barre* et les distances *barres-croisées*.

Les distances barre-à-barre considèrent un histogramme comme un vecteur de dimension fixe et ne comparent seulement que le contenu des barres correspondantes des histogrammes. Pour comparer ces barres, il est possible d'utiliser différentes métriques/fonctions de distance. Les plus couramment utilisées sont : la distance de Manhattan (L_1), la distance Euclidienne (L_2), la distance χ^2, *etc.* Par la suite, ces distances seront notées D_{L_1}, D_{L_2}, et D_{χ^2}. Ces distances présentent des propriétés spécifiques discutées dans (Cha, 2008).

Comme ces distances ne comparent que les barres correspondantes, et qu'elles ignorent les corrélations entre les barres adjacentes, les distances barre-à-barre sont rapides à calculer et peuvent être utilisées pour mesurer des similarités dans de grands jeux de données. De plus, comme elles ne requièrent pas un ordre parmi les barres, elles peuvent être utilisées pour comparer des histogrammes nominaux ou ordonnés. Cependant, elle souffrent du problème de translation : une faible translation des valeurs de l'histogramme peut significativement affectée la distance entre histogrammes. De plus, les distances barres-à-barres sont fortement liées au choix de la « taille » des barres des histogrammes : des barres trop grosses n'offriront pas une capacité de discrimination suffisante tandis que des barres trop fines sépareront des caractéristiques similaires/corrélées en différentes barres qui ne seront jamais comparées.

Les distances barres-croisées permettent de prendre en compte ces problèmes. Elles considèrent un histogramme comme une estimation de la *fonction de densité de probabilité* et comparent aussi bien les barres correspondantes que les non-correspondantes. De nombreuses distances barres-croisées ont déjà été proposées pour la comparaison d'histogrammes. Parmi celles-ci, nous citerons les distances : quadratiques (Niblack et al., 1993), par correspondances exactes (*e.g.*, « Earth Mover's Distance ») (Rubner et al., 2000), et temporelles (Strelkov, 2008).

De telles distances permettent de prendre en considération la proximité entre les barres, améliorant ainsi l'évaluation de la dissimilarité entre histogrammes. Cependant, cet avantage induit un coût de calcul élevé. De plus, ces distances requièrent un ordre total parmi les barres, et peuvent ainsi seulement être utilisées pour comparer des histogrammes ordonnés.

Un problème sémantique En général, quand on compare des histogrammes nominaux, aucune information *a priori* relative à l'ordre et/ou aux relations parmi les barres, n'est dis-

TAB. 1 – *Histogrammes modélisant la composition des trois paniers P_1, P_2, P_3.*

	Citron	*Coing*	*Pomme*	*Clémentine*	*Abricot*	*Poire*	*Pêche*	*Cerise*	*Orange*	*Prune*
$P_1 - H_1$	9	0	0	0	0	0	0	0	1	0
$P_2 - H_2$	1	0	0	0	0	0	0	0	1	8
$P_3 - H_3$	1	0	0	0	0	0	0	0	8	1

ponible. Par conséquence, il est souvent proposé d'utiliser des distances barre-à-barre (Cha, 2008). Cependant, quand un histogramme nominal modélise la distribution des instances d'un concept sémantique, des informations à propos des relations entre ces instances peuvent être considérées. Dans ce contexte, il devient utile d'utiliser une distance permettant de prendre en compte les similarités sémantiques entre les distributions modélisées par ces histogrammes.

Considérons, par exemple, l'histogramme modélisant la composition d'un panier de courses en termes de fruits qui a été défini précédemment. Chaque barre de l'histogramme représente la proportion d'un type de fruit qui est une instance du concept sémantique FRUIT. Il est évident que l'instance Citron du concept FRUIT est plus proche de l'instance Orange que de l'instance Prune, car Citron et Orange sont tous les deux des agrumes. Bien qu'il soit impossible de déterminer un ordre total parmi ces instances, il est possible de déterminer des similarités sémantiques/thématiques entre les barres composant ces histogrammes nominaux.

Considérons maintenant trois paniers de courses P_1, P_2, P_3, chacun composé de dix fruits. Ces fruits peuvent prendre les valeurs {Citron, Coing, Pomme, Clémentine, Abricot, poire, Pêche, Cerise, Orange, Prune}. Ainsi, la composition d'un panier P_i peut être modélisée par un histogramme $H_i(\#\text{Citron}, \#\text{Coing}, \#\text{Pomme}, \ldots, \#\text{Orange}, \#\text{Prune})$ où $\#x$ représente le nombre d'occurrences de l'instance x dans le panier P_i. La composition des trois paniers considérés P_1, P_2, P_3 est présentée par le Tab. 1. Si l'on considère une distance classique barre-à-barre, par exemple la distance de Manhattan d_{L_1}, il y a la même distance entre H_1 et H_2 ($d_{L_1}(H_1, H_2) = 16$) qu'entre H_1 et H_3 ($d_{L_1}(H_1, H_3) = 16$). Cependant, il est évident que P_1 est sémantiquement plus proche de P_3 que de P_2 car P_1 et P_3 sont les deux des paniers d'agrumes. Ainsi, les distances barre-à-barre ne sont pas efficaces pour comparer de tels histogrammes, et il n'est pas possible d'utiliser une distance barres-croisées car les barres ne sont pas totalement ordonnées. Nous nommerons ce problème le *problème sémantique*.

Motivations et proposition Récemment, une nouvelle distance basée sur un modèle de calculs fin-à-grossier a été proposée (Ma et al., 2010). Ce modèle repose sur un processus itératif de fusion des barres les plus proches des histogrammes pour créer des histogrammes plus grossiers (*i.e.*, moins détaillés). Ce modèle permet ainsi de comparer les barres correspondantes et non-correspondantes afin de prendre en considération les corrélations entre les barres adjacentes. Cette distance, appliquée à la fouille d'images, a fourni des résultats encourageants.

Les distances basées sur un modèle de calculs fin-à-grossier semblent bien adaptées à traiter le problème sémantique énoncé précédemment. Dans le but d'illustrer cette proposition, revenons à l'exemple des paniers de courses (voir paragraphe précédent).

Considérons les trois paniers P_1, P_2, P_3 (chacun composé de dix fruits). Nous rappelons que la composition d'un panier P_i est modélisée par un histogramme

TAB. 2 – *Histogrammes modélisant les compositions des trois paniers P_1, P_2, P_3 (voir Tab. 1) après la création de l'instance Agrume.*

	Agrume	Coing	Pomme	Abricot	Poire	Pêche	Cerise	Prune
$P_1 - H'_1$	**10**	0	0	0	0	0	0	0
$P_2 - H'_2$	**2**	0	0	0	0	0	0	8
$P_3 - H'_3$	**9**	0	0	0	0	0	0	1

$H_i(\#\text{Citron}, \#\text{Coing}, \#\text{Pomme}, \ldots, \#\text{Orange}, \#\text{Prune})$. Supposons maintenant que nous fusionnions les instances Citron, Orange et Clémentine pour créer une nouvelle instance nommée Agrume. La composition d'un panier de courses P_i est maintenant modélisée par un histogramme $H'_i(\#\textbf{Agrume}, \#\text{Coing}, \#\text{Pomme}, \ldots, \#\text{Prune})$ où $\#\textbf{Agrume} = \#\text{Citron} + \#\text{Clémentine} + \#\text{Orange}$. La composition résultante des trois paniers P_1, P_2, P_3 est présentée dans le Tab. 2. La distance barre-à-barre de Manhattan d_{L_1} devient maintenant plus grande entre H'_1 et H'_2 ($d_{L_1}(H'_1, H'_2) = 16$) qu'entre H'_1 et H'_3 ($d_{L_1}(H'_1, H'_3) = 2$). Cette valeur de mesure reflète mieux les similarités sémantiques entre les paniers P_1 et P_3 qui sont tous les deux des paniers principalement composés d'agrumes.

Comme illustré dans (Ma et al., 2010), les distances basées sur un modèle fin-à-grossier sont dérivées des mesures de similarité barres-croisées. Cependant, quand les données présentent un problème sémantique justifiant l'utilisation de telles distances, un ordre total (obligatoire pour l'utilisation d'une distance barres-croisées) n'est généralement pas disponible.

Dans cet article, nous proposons de résoudre ce problème en définissant une distance basée sur un modèle de calculs fin-à-grossier qui ne dérive pas d'une distance barres-croisées. Cette nouvelle distance, nommée *Hierarchical Semantic-Based Distance* (HSBD), est basée sur l'utilisation de distances barre-à-barre mais permet de prendre en compte les corrélations sémantiques entre les instances considérées grâce à une mesure de proximité sémantique fournie par des connaissances du domaine. Cette distance combine l'efficacité des distances barre-à-barre (*e.g.*, faible coût de calcul) et les avantages offerts par les distances barres-croisées (*e.g.*, robustesse aux problèmes de translation d'histogramme et du choix de la taille des barres).

Cet article s'articule de la manière suivante. La section 2 introduit des définitions et notations préliminaires. La section 3 décrit la distance proposée dans cet article dédiée à la comparaison d'histogrammes nominaux. La section 4 propose une validation expérimentale. Des conclusions et perspectives sont finalement données en section 5.

2 Définitions préliminaires

Un intervalle sur $\mathbb{R}$, borné par $a, b \in \mathbb{R}$, sera noté $[a, b]$ tandis qu'un intervalle sur $\mathbb{Z}$, borné par $a, b \in \mathbb{Z}$, sera noté $[\![a, b]\!]$. Une liste $\mathcal{L}_v$ de v éléments e_i avec $i \in [\![0, v-1]\!]$ sera définie comme $\langle e_i \rangle_0^{v-1} = \langle e_0, e_1, \ldots, e_{v-1} \rangle$.

Histogramme Soit x une mesure, ou un attribut, qui peut prendre v valeurs dans l'ensemble $X = \{x_0, x_1, \ldots, x_{v-1}\}$. Soit A un ensemble de n éléments/objets. Chaque élément de A est associé à une valeur a par la mesure x. L'observation résultante de cette mesure sera notée

$A_x = \{a_1, a_2, \ldots, a_n\}$ où $a_i \in X$. L'histogramme de l'ensemble A_x relatif à la mesure x de A, noté $H(x, A)$ est une liste de v éléments comptant le nombre d'occurrences des valeurs de x parmi les a_i. Par souci de lisibilité, nous utiliserons la notation $H(A)$ au lieu de $H(x, A)$. L' histogramme $H(A)$ peut être défini comme $H(A) = \langle H_0(A), H_1(A), \ldots, H_{v-1}(A)\rangle$ où $H_i(A)$, $i \in [\![0, v-1]\!]$, dénombre les occurrences des éléments de A_x qui ont la valeur x_i. Chaque $H_i(A)$ est appelé une « barre » de l'histogramme $H(A)$ et peut être calculée comme

$$H_i(A) = \sum_{j=1}^{n} c_{ij} \quad \text{où} \quad c_{ij} = \begin{cases} 1 & \text{si } a_j = x_i \\ 0 & \text{sinon} \end{cases} \tag{1}$$

Les v valeurs de la mesure x sont généralement appelées des *niveaux de mesure* ou des *modalités* quand elles sont utilisées dans $H(A)$ pour indexer les distributions des valeurs des échantillons. Ces v valeurs sont aussi appelées des *instances* quand elles sont utilisées dans $H(A)$ pour indexer les distributions des instances possibles du concept sémantique représenté.

Distances entre niveaux de mesure Un histogramme $H(A)$ représente la distribution des valeurs quantifiées d'une mesure x parmi les échantillons d'un ensemble A. Relativement à deux types de mesures (*i.e.*, nominale et ordonnée), nous définissons deux fonctions d_{nom} et d_{ord} qui mesurent la différence entre deux niveaux $x_i, x_j \in X$. Dans la littérature, la différence entre deux niveaux de mesure est appelée la *distance au sol*.

Dans une mesure nominale, il n'y pas de relation entre les valeurs x_i. Ainsi, nous définissons la distance au sol entre ces valeurs comme valeurs « correspondantes » ou « non-correspondantes » :

$$d_{nom}(x_i, x_j) = \begin{cases} 0 & \text{si } x_i = x_j \\ 1 & \text{sinon} \end{cases} \tag{2}$$

Dans une mesure ordonnée, les valeurs x_i sont totalement ordonnées et il est possible de déterminer une distance atomique $\Delta_{(x_i,\, x_{i+1})} \in \mathbb{R}_+$ entre chaque paire de niveaux successifs x_i et x_{i+1}. Ainsi, nous définissons la distance au sol entre deux mesures ordonnées x_i et x_j comme la somme des distances atomiques entre chaque niveau successif de i à j :

$$d_{ord}(x_i, x_j) = \sum_{k=1}^{j-1} \Delta_{(x_k,\, x_{k+1})} \tag{3}$$

Quand les valeurs de la mesure sont numériques (*i.e.*, chaque $x_i \in \mathbb{R}$), la distance au sol entre deux niveaux de mesure ordonnés est la différence absolue entre ces niveaux :

$$d_{ord}(x_i, x_j) = \sum_{k=1}^{j-1} |x_k - x_{k+1}| = |x_i - x_j| \tag{4}$$

3 Distance proposée

Le calcul de la distance HSBD, entre deux histogrammes $H(A)$ et $H(B)$, requière deux paramètres : une matrice de dissimilarité $\mathcal{M}^{dis}$ modélisant les valeurs de proximité sémantique entre les instances de $H(A)$ et $H(B)$ et une distance d'histogrammes barre-à-barre D_{bin}.

Avant de pouvoir calculer la distance HSBD, la stratégie adoptée (basée sur un modèle de calculs fin-à-grossier) requière de définir un moyen de fusionner hiérarchiquement les différentes instances représentées par les histogrammes en « clusters » d'instances (*i.e.*, des instances de niveaux sémantiques plus élevés). Cette étape de pré-traitement, décrite dans la Sec. 3.1, repose principalement sur la construction d'un dendrogramme $\mathfrak{D}$ induit par $\mathcal{M}^{dis}$ et modélisant la hiérarchie de fusion des instances. Il est à noter que cette étape de pré-traitement ne doit être effectuée qu'une seule fois pour une matrice de dissimilarité donnée.

Une fois que le dendrogramme $\mathfrak{D}$ a été construit, la distance HSBD peut être calculée. Son calcul se décompose en deux étapes principales (détaillées en Sec. 3.2) :

- **Étape 1. Calcul des sous-distances barre-à-barre hiérarchiques**
  Durant un processus de fusion itératif scannant chaque étage du dendrogramme (de ses feuilles jusqu'à sa racine), les histogrammes liés à $H(A)$ and $H(B)$, et induits par la fusion des instances composant chaque cluster de l'étage courant, sont construits. Après chaque itération, la distance barre-à-barre D_{bin} est calculée entre chaque couple d'histogrammes créé.
- **Étape 2. Fusion des sous-distances barre-à-barre** Les distances barre-à-barre calculées pour tous les étages du dendrogramme, et l'énergie sémantique nécessaire pour aller d'un étage à l'autre, sont ensuite fusionnées en une fonction qui est finalement intégrée pour fournir la valeur de la distance HSBD.

3.1 Considérer la proximité sémantique

Matrice de dissimilarité Dans la section 2, nous avons défini la fonction d_{nom} entre deux valeurs de mesure nominale $x_i, x_j \in X$ comme valeurs « correspondantes » ou « non-correspondantes » (Eq. (2)). Cette définition ne permet pas de prendre en compte une proximité sémantique entre les barres d'un histogramme nominal.

Pour résoudre ce problème, cette distance au sol peut être étendue comme :

$$d_{nom}(x_i, x_j) = d_{nom}(x_j, x_i) = \begin{cases} 0 & \text{si } x_i = x_j \\ \alpha_{(x_i,\, x_j)} & \text{sinon} \end{cases} \tag{5}$$

où $\alpha_{(x_i,\, x_j)} \in]0, 1]$ reflète une valeur de dissimilarité sémantique entre x_i et x_j fournie par les connaissances du domaine de l'expert. Ainsi, il est possible de définir une matrice de dissimilarité $\mathcal{M}^{dis}$ de taille $v \times v$ qui modélise les relations entre chaque instance $x \in X = \{x_0, x_1, \ldots, x_{v-1}\}$ du concept considéré par l'histogramme :

$$\mathcal{M}^{dis} = \begin{bmatrix} \alpha_{(x_0,\, x_0)} & \cdots & \alpha_{(x_0,\, x_{v-1})} \\ \vdots & \ddots & \vdots \\ \alpha_{(x_{v-1},\, x_0)} & \cdots & \alpha_{(x_{v-1},\, x_{v-1})} \end{bmatrix} \tag{6}$$

Construction de la hiérarchie de fusion sémantique Le principe de l'approche proposée est de calculer plusieurs fois une distance barre-à-barre entre des paires d'histogrammes en fusionnant progressivement les barres/instances les plus proches pour créer des histogrammes plus grossiers (*i.e.*, de plus hauts niveaux sémantiques). Pour ce faire, il est nécessaire de définir une hiérarchie de fusion des instances qui permet de déterminer l'ordre des fusions entre les instances d'un concept sémantique.

En partant des valeurs de proximité sémantique contenues dans $\mathcal{M}^{dis}$, il est possible de calculer la hiérarchie de fusion des instances en utilisant un algorithme de classification hiérarchique ascendante (AHC). Cet algorithme construit hiérarchiquement des clusters d'instances minimisant une mesure d'inertie intra-cluster. Le critère de fusion utilisé est le « Average Linkage ».

Cette hiérarchie de fusion est modélisée par un dendrogramme $\mathfrak{D}$ de s étages, dont la racine est le cluster qui regroupe toutes les instances. Chaque étage de $\mathfrak{D}$ correspond à un niveau de sémantique particulier. La valeur minimale de s ($s_{min} = 2$) est atteinte quand $\mathcal{M}^{dis}$ est une matrice où $\alpha_{(x_i,\, x_j)} = 1$ si $x_i \neq x_j$ et $\alpha_{(x_i,\, x_j)} = 0$ sinon (*i.e.*, aucune connaissance du domaine). Dans ce cas, le dendrogramme présente un étage pour les feuilles qui sont les instances de base et un étage pour la racine.

Nous définissons, à partir de ce dendrogramme :

- une fonction $f_{\mathfrak{D}}$ qui prend en entrée l'indice k de l'étage S_k ($k \in [\![0, s-1]\!]$) et fournit en sortie la liste $\mathcal{L}_m^k$ composée des m listes de fusion d'instances $\mathcal{L}_{v_i}$ (with $i \in [\![0, m-1]\!]$) données par $\mathfrak{D}$ à cet étage.
- une fonction $h_{\mathfrak{D}}$ qui prend en entrée l'indice k de l'étage S_k et fournit en sortie sa hauteur $h_{\mathfrak{D}}(k)$ dans le dendrogramme $\mathfrak{D}$.

Cette hauteur $h_{\mathfrak{D}}(k)$ correspond à « l'énergie » nécessaire pour construire les clusters d'instances induits par l'étage S_k (*i.e.*, l'inertie inter-cluster calculée pendant la création de $\mathfrak{D}$).

3.2 Calcul de HSBD

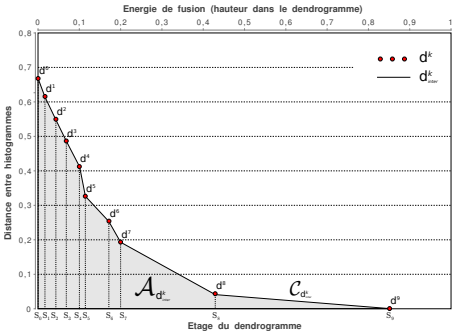
Une fois l'étape de pré-traitement réalisée, nous disposons d'un dendrogramme $\mathfrak{D}$ qui modélise la hiérarchie de fusion des instances. Il est maintenant possible de calculer la distance HSBD entre les deux histogrammes $H(A)$ et $H(B)$ composés de v barres.

Étape 1. Calcul des sous-distances barre-à-barre hiérarchiques Pour calculer les distances barre-à-barre hiérarchiques durant le processus de fusion itératif, la fonction $\mathbf{d}^k$ a été définie. Cette fonction prend en entrée deux histogrammes $H(A)$ et $H(B)$ et l'indice k de l'étage S_k du dendrogramme et fournit en résultat la distance barre-à-barre D_{bin} calculée entre $H^k(A)$ et $H^k(B)$. Cette fonction $\mathbf{d}^k$ peut être définie comme :

$$\mathbf{d}^k(H(A), H(B)) = D_{bin}(H^k(A), H^k(B)) \tag{7}$$

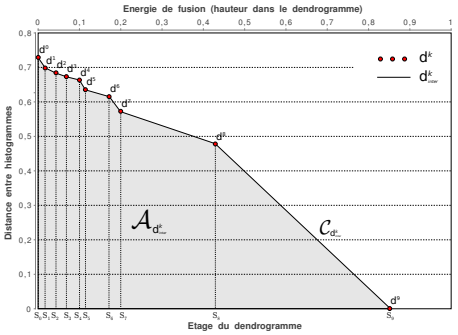
où $H^k(A)$ et $H^k(B)$ sont des histogrammes grossiers (fournissant un plus haut niveau de sémantique) induits par le regroupement des instances considérées à l'étage S_k. De tels histogrammes peuvent être construits en utilisant la fonction $f_{\mathfrak{D}}(k)$ qui fournit une liste $\mathcal{L}_m^k = \langle \mathcal{L}_{v_0}, \ldots, \mathcal{L}_{v_{m-1}} \rangle$ composée des m listes de fusion d'instances induits par l'étage S_k. Un histogramme $H^k(Y)$ pourra être défini comme $H^k(Y) = \langle H_0^k(Y), H_1^k(Y), \ldots, H_{m-1}^k(Y) \rangle$ où chaque barre $H_i^k(Y)$ peut être calculée avec $H_i^k(Y) = \sum_{j=0}^{v_i} H_j(Y)$. Par souci de lisibilité, la fonction $\mathbf{d}^k(H(A), H(B))$ peut aussi être notée $\mathbf{d}^k$. De plus, les valeurs induites par cette fonction seront appelées des valeurs de *sous-distance*.

Le processus de fusion itératif fin-à-grossier fonctionne de la manière suivante : dans un premier temps, la sous-distance barre-à-barre $\mathbf{d}^0$ est calculée pour $H(A)$ et $H(B)$ en considérant toutes les barres des histogrammes (*i.e.*, $f_{\mathfrak{D}}(0) = \langle \langle x_0 \rangle, \langle x_1 \rangle, \ldots, \langle x_{v-1} \rangle \rangle$ et $h_{\mathfrak{D}}(0) = 0$).



	Citron	Coing	Pomme	Clémentine	Abricot	Poire	Pêche	Cerise	Orange	Prune
$H(A)$	25	15	0	20	0	0	25	0	10	5
$H(B)$	5	0	5	40	0	10	0	0	30	10

(a) Couple d'histogrammes sémantiquement similaires.



	Citron	Coing	Pomme	Clémentine	Abricot	Poire	Pêche	Cerise	Orange	Prune
$H(A)$	0	25	25	0	5	15	0	0	0	30
$H(B)$	20	0	10	0	30	0	0	5	25	0

(b) Couple d'histogrammes sémantiquement différents.

FIG. 1 – *Représentation graphique des fonctions* $\mathbf{d}^k$ *et* $\mathbf{d}^k_{inter}$ *calculées entre des couples d'histogrammes exemples (modélisant la composition d'un panier de 100 fruits). Suivant le contenu des histogrammes* $H(A)$ *et* $H(B)$*, le comportement des fonctions* $\mathbf{d}^k$ *et* $\mathbf{d}^k_{inter}$ *diffère.*

Ensuite, après avoir « grimpé » à l'étage suivant S_i du dendrogramme, les barres les plus proches sémantiquement des histogrammes (données par $f_{\mathfrak{D}}(i)$) sont fusionnées et une nouvelle valeur de sous-distance $\mathbf{d}^i$ est calculée entre les deux histogrammes résultants $H^i(A)$ et $H^i(B)$. Cette nouvelle valeur de sous-distance permet d'évaluer la similarité à un niveau particulier de « taille de barres » (et ainsi de sémantique). Cette étape est répétée pour chaque étage $S_k, k \in [\![1, s-1]\!]$ jusqu'à ce que le nombre de barres soit égale à 1 (*i.e.*, le processus se termine quand la racine de l'arbre est atteinte). Une série de sous-distances hiérarchiques $\mathbf{d}^0$, $\mathbf{d}^1$, $\mathbf{d}^2$, …, $\mathbf{d}^{s-1}$ est ainsi calculée et sauvegardée. Il est à noter que $\mathbf{d}^{s-1}$ est toujours égale à 0 car les histogrammes $H^{s-1}(A)$ et $H^{s-1}(B)$ sont toujours composés d'une seule barre représentant l'instance du plus haut niveau sémantique du dendrogramme (*i.e.*, la racine).

Étape 2. Fusion des sous-distances barre-à-barre Une fois que les sous-distances hiérarchiques $\mathbf{d}^0$, $\mathbf{d}^1$, $\mathbf{d}^2$, …, $\mathbf{d}^{s-1}$ ont été calculées, il est possible de les fusionner pour obtenir une mesure globale de distance entre les deux histogrammes $H(A)$ et $H(B)$.

Considérons la représentation graphique de la fonction $\mathbf{d}^k$ illustrée en Fig. 1. Ces deux courbes représentent la fonction $\mathbf{d}^k$ calculée entre des couples d'histogrammes exemples. On peut observer que plus les valeurs de $\mathbf{d}^k$ décroissent rapidement, plus les histogrammes comparés sont sémantiquement/thématiquement similaires. Ainsi, pour obtenir la distance HSBD nous analysons le comportement de la fonction $\mathbf{d}^k$.

Pour ce faire, considérons la fonction de sous-distance $\mathbf{d}^k$ qui est définie pour $k \in [\![0, s-1]\!]$. Construisons maintenant la fonction $\mathbf{d}^k_{inter}$ qui associe à tout t, tel que t soit la hauteur $h_{\mathfrak{D}}(k)$ de chaque étage S_k du dendrogramme, sa valeur de sous-distance $\mathbf{d}^k$ correspondante. Pour

toutes les autres valeurs de t (les hauteurs « inter-étages »), la fonction $\mathbf{d}^k_{inter}$ est définie comme une interpolation linéaire de la fonction $\mathbf{d}^k$.

Soit $\mathcal{C}_{\mathbf{d}^k_{inter}}$ la courbe qui représente graphiquement la fonction $\mathbf{d}^k_{inter}$ et soit $\mathcal{A}_{\mathbf{d}^k_{inter}}$ l'aire en dessous de cette courbe (Fig. 1). Plus la valeur de l'aire $\mathcal{A}_{\mathbf{d}^k_{inter}}$ est faible, plus les histogrammes $H(A)$ et $H(B)$ sont sémantiquement/thématiquement similaires, et inversement. Ainsi nous utilisons cette aire pour calculer la similarité entre les deux histogrammes considérés. Nous définissons ainsi la distance HSBD entre $H(A)$ et $H(B)$ comme

$$\text{HSBD}(H(A), H(B)) = \int_0^{h_{\mathfrak{D}}(s-1)} \mathbf{d}^k_{inter}(H(A), H(B))(t)\, dt \tag{8}$$

où dt représente l'énergie nécessaire à la formation du niveau k de fusion (*i.e.*, la hauteur de l'étage S_k dans le dendrogramme).

Pour calculer HSBD nous utilisons la méthode des Trapèzes qui est une approche classique pour calculer une intégrale finie. Ainsi, HSBD peut être calculée de la façon suivante

$$\text{HSBD}(H(A), H(B)) = \frac{1}{2} \sum_{k=0}^{s-2} \left[\left(\mathbf{d}^{k+1} + \mathbf{d}^k \right) \left(h_{\mathfrak{D}}(k+1) - h_{\mathfrak{D}}(k) \right) \right] \tag{9}$$

4 Validation expérimentale

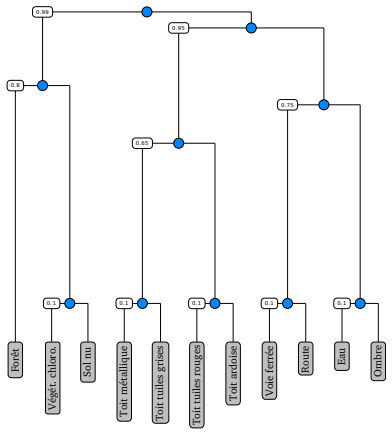
Données La capacité croissante des satellites à acquérir des images terrestres engendre des masses de données importantes. De ce fait, les méthodes (manuelles) d'extraction de connaissances deviennent difficilement utilisables et il est de plus en plus nécessaire d'analyser automatiquement ces données. La méthodologie classique consiste à classifier, d'une manière supervisée ou non, ces données en ensembles de classes de couverture des sols.

Pour valider la distance proposée, nous l'avons appliquée à la classification de blocs urbains, qui peuvent être définis comme les ensembles minimaux urbains fermés par des voies de communication. La principale originalité de cette tâche est de classifier des ensembles de blocs urbains dans lesquels chaque bloc est caractérisé par sa composition « au sol » en termes d'objets urbains élémentaires (*e.g.*, maisons individuelles, jardins, routes, *etc.*). Ainsi, un bloc urbain U_i peut être caractérisé par un histogramme $H_i(\#\text{Toit tuiles rouges}, \#\text{Toit ardoise}, \ldots, \#\text{Végétation chlorophyllienne})$ où Toit tuiles rouges, Toit ardoise, ..., Végétation chlorophyllienne sont des instances du concept sémantique OBJET URBAIN (Fig. 2(a)).

La principale difficulté est de parvenir à regrouper, dans une même classe, différents objets qui ne sont pas caractérisés par des histogrammes statistiquement similaires. L'exemple suivant illustre ce problème. Considérons le bloc U_i caractérisé par un histogramme $H_i(15, 3, \ldots, 10)$ (*i.e.*, 15 toits en tuiles rouges, 3 toits en ardoises, ..., 10 parcelles de végétation) et un bloc U_j caractérisé par un histogramme $H_j(3, 22, \ldots, 10)$ (*i.e.*, 3 toits en tuiles rouges, 22 toits en ardoises, ..., 10 parcelles de végétation). Du point de vue de l'expert, ces deux blocs doivent être groupés dans la même classe « Blocs d'habitations individuelles » car ils sont tous les deux composés de maisons individuelles (avec des toits rouges en tuiles ou en ardoise) et des parcelles de végétation. Une solution pour résoudre ce problème consiste à utiliser un processus de classification associé à une distance prenant en compte les corrélations sémantiques des données. Pour valider l'utilité de la distance HSBD, nous proposons de l'intégrer dans un processus de classification pour traiter de telles données.

x_i	*Ombre*	*Eau*	*Route*	*Voie ferrée*	*Sol nu*	*Végét. chloro.*	*Forêt*	*Toit ardoise*	*Toit tuil. rouges*	*Toit tuil. grises*	*Toit métallique*
Ombre	0.0	0.1	1.0	1.0	1.0	1.0	0.8	0.8	0.8	0.8	0.8
Eau	–	0.0	0.5	0.5	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Route	–	–	0.0	0.1	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Voie ferrée	–	–	–	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Sol nu	–	–	–	–	0.0	0.1	0.8	1.0	1.0	1.0	1.0
Végét. chloro.	–	–	–	–	–	0.0	0.8	1.0	1.0	1.0	1.0
Forêt	–	–	–	–	–	–	0.0	1.0	1.0	1.0	1.0
Toit ardoise	–	–	–	–	–	–	–	0.0	0.1	0.4	0.9
Toit tuiles rouges	–	–	–	–	–	–	–	–	0.0	0.4	0.9
Toit tuiles grises	–	–	–	–	–	–	–	–	–	0.0	0.1
Toit métallique	–	–	–	–	–	–	–	–	–	–	0.0

(a) Matrice de dissimilarité $\mathcal{M}^{dis}$.



(b) Dendrogramme $\mathfrak{D}$ à 7 étages.

FIG. 2 – *Connaissances du domaine. (a) Matrice de dissimilarité associée aux instances du concept* OBJET URBAIN. *(b) Dendrogramme calculé via l'étape de pré-traitement (Sec. 3.1).*

Nous disposons de trois jeux de données (nommés **D1**, **D2** et **D3**) chacun composé d'un ensemble de blocs urbains dont les contours ont été manuellement extraits par un expert et d'une carte haute résolution qui informe de la composition de chacun de ces blocs en termes d'objets urbains élémentaires. Chaque bloc urbain U_i a ainsi été caractérisé par un histogramme de « composition » $H(U_i)$ qui modélise la décomposition de ce bloc en fonction de la distribution des onze instances du concept sémantique OBJET URBAIN dans la carte haute résolution. Les onze instances du concept sémantique OBJET URBAIN sont listées en Fig. 2(a).

Expérimentations Pour modéliser les relations sémantiques entre les instances des histogrammes, une matrice de dissimilarité $\mathcal{M}^{dis}$ de taille 11×11 a été fournie par l'expert (Fig. 2(a)). À partir de cette matrice, un dendrogramme de sept étages a été construit (en utilisant le pré-traitement, voir Sec. 3.1) pour stocker l'ordre de fusion des instances (Fig. 2(b)).

Les algorithmes de classification supervisée requièrent des exemples d'apprentissage pour entraîner le modèle de classification. Dans notre cas, la définition de tels exemples d'apprentissage est une tâche très complexe pour l'expert. En effet, le grand nombre de classes extractibles induit un grand nombre d'exemples à définir. De plus, ces exemples étant fortement dépendant des données, ils ne peuvent pas être ré-utilisés pour la classification d'autres jeux de données. Pour ces raisons, nous avons choisi d'utiliser un algorithme de classification non-supervisée qui ne nécessite pas la définition de tels exemples d'apprentissage. Nous avons ainsi employé l'algorithme de clustering K-MEANS (qui ne requière pas de paramètre difficile à estimer *a priori*) pour classifier les blocs urbains construits précédemment.

Pour procéder, la distance HSBD a été directement intégrée dans l'algorithme K-MEANS pour comparer les histogrammes classifiés. Nous avons appliqué l'algorithme K-MEANS en associant la distance HSBD à différentes sous-distances barre-à-barre (HSBD_{L_1}, HSBD_{L_2}, et HSBD_{χ^2}). Pour comparer la distance HSBD à d'autres distances existantes, l'algorithme

TAB. 3 – *Résultats et scores d'évaluation pour les trois jeux de données considérés.*

		$\overline{\mathcal{F}} \pm \sigma$		$\mathcal{K} \pm \sigma$	
Jeu de données	Distance barre-à-barre	D_{bin}	HSBD	D_{bin}	HSBD
D1	L_1	0.65 ± 0.02	$\mathbf{0.71 \pm 0.01}$	0.76 ± 0.02	$\mathbf{0.79 \pm 0.01}$
	L_2	0.63 ± 0.03	$\mathbf{0.69 \pm 0.02}$	0.75 ± 0.03	$\mathbf{0.78 \pm 0.02}$
	χ^2	0.59 ± 0.02	$\mathbf{0.65 \pm 0.02}$	0.72 ± 0.03	$\mathbf{0.75 \pm 0.02}$
D2	L_1	0.67 ± 0.02	$\mathbf{0.72 \pm 0.01}$	0.77 ± 0.02	$\mathbf{0.86 \pm 0.02}$
	L_2	0.64 ± 0.01	$\mathbf{0.70 \pm 0.02}$	0.76 ± 0.02	$\mathbf{0.83 \pm 0.01}$
	χ^2	0.61 ± 0.02	$\mathbf{0.68 \pm 0.03}$	0.73 ± 0.01	$\mathbf{0.76 \pm 0.02}$
D3	L_1	0.63 ± 0.01	$\mathbf{0.66 \pm 0.01}$	0.73 ± 0.01	$\mathbf{0.76 \pm 0.01}$
	L_2	0.60 ± 0.03	$\mathbf{0.63 \pm 0.01}$	0.73 ± 0.02	$\mathbf{0.75 \pm 0.01}$
	χ^2	0.58 ± 0.02	$\mathbf{0.62 \pm 0.02}$	0.71 ± 0.01	$\mathbf{0.73 \pm 0.02}$

K-MEANS a aussi été appliqué en utilisant les distances classiques barre-à-barre D_{L_1}, D_{L_2}, et D_{χ^2}. Ces comparaisons permettent d'évaluer les avantages apportés par l'utilisation de la distance HSBD par rapport à l'utilisation d'une distance classique barre-à-barre (*e.g.*, HSBD$_{L_1}$ *vs.* D_{L_1}, HSBD$_{L_2}$ *vs.* D_{L_2} et HSBD$_{\chi^2}$ *vs.* D_{χ^2}). Les résultats fournis par l'algorithme K-MEANS étant sensibles à son initialisation, chaque test a été répété dix fois en faisant varier les « graines » de l'algorithme. Nous avons ainsi pu calculer la variance σ obtenue pour chaque série de tests et pour chaque indice d'évaluation considéré (décrits dans le paragraphe suivant).

De ces jeux de données, nous avons choisi avec l'expert d'extraire 9 classes thématiques. L'algorithme K-MEANS a ainsi été instancié avec 9 clusters pour chaque jeu de données.

Évaluation des résultats L'évaluation d'un résultat de clustering est une tâche complexe car il est difficile de trouver une mesure objective évaluant la qualité d'un cluster. Une stratégie classique consiste à calculer et à comparer les inerties intra et inter clusters obtenues sur les différents résultats (*i.e.*, évaluation non supervisée). Cependant, dans notre cas, l'algorithme de clustering est instancié en faisant varier la distance utilisée pour classifier les données (engendrant ainsi une nouvelle définition de l'inertie). Il ne semble donc pas pertinent d'utiliser l'inertie pour évaluer et comparer la qualité des résultats obtenus.

Nous avons donc considéré des techniques d'évaluation supervisées qui consistent à comparer les résultats de clustering à des données manuellement labellisées. Pour ce faire, les résultats ont été comparés à des cartes de vérités terrain fournies par l'expert. Nous avons calculé l'indice Kappa $\mathcal{K}$ qui est une mesure de la précision des résultats (relativement à la vérité terrain) et la moyenne harmonique $\overline{\mathcal{F}}$ des F-mesures associées aux classes extraites.

Les scores d'évaluation obtenus pour chaque jeu de données sont présentés dans le Tab. 3. À partir de ces résultats, on observe que les scores de Kappa et de F-mesure sont toujours supérieurs quand K-MEANS est utilisé avec la distance HSBD que quand il est utilisé avec la distance barre-à-barre correspondante D_{bin}. En particulier, les meilleurs scores ont été obtenus quand HSBD est combinée à la sous-distance de Manhattan D_{L_1}. Par ailleurs, les moins bons scores ont été obtenus quand HSBD est combinée à la sous-distance χ^2. Néanmoins, ces scores demeurent toujours supérieurs à ceux obtenus en utilisant une distance barre-à-barre classique.

Ces résultats montrent ainsi l'intérêt et l'utilité de la distance HSBD pour la comparaison d'histogrammes nominaux portant une sémantique. D'une manière plus générale, ces validations, dans le contexte de la classification de données géographiques, mettent en avant l'intérêt de cette distance pour des tâches de fouille de données.

5 Conclusion et perspectives

Cette article a présenté une distance dédiée à la comparaison d'histogrammes nominaux. La principale originalité de cette mesure est de prendre en compte les relations de proximité/corrélation sémantique entre les barres des histogrammes considérés. De plus, cette distance est basée sur un modèle de calculs fin-à-grossier apportant une solution aux problèmes liés à la translation des valeurs de l'histogramme et au choix de la taille des barres.

Cette distance a finalement été validée dans le cadre de la classification de données géographiques. Les résultats encourageants obtenus avec cette distance ont ainsi montré l'utilité de cette dernière pour des processus de fouille de données. Par la suite, nous prévoyons d'étudier d'une manière plus formelle son comportement, ainsi que son applicabilité à d'autres domaines comme la fouille de textes ou la classification de motifs symboliques.

Références

Cha, S. H. (2008). Taxonomy of nominal type histogram distance measures. In *Proceedings of the American Conference on Applied Mathematics*, pp. 325–330. WSEAS.

Cha, S. H. et S. N. Srihari (2002). On measuring the distance between histograms. *Pattern Recognition 35*(6), 1355–1370.

Duda, R. O., P. E. Hart, et D. G. Stork (2000). *Pattern Classification*. Wiley, New York.

Fabrizio, S. (2002). Machine learning in automated text categorization. *ACM Computing Surveys 34*, 1–47.

Jain, A. K., M. N. Murty, et P. J. Flynn (1999). Data clustering : A review. *ACM Computing Surveys 31*(3), 264–323.

Ma, Y., X. Gu, et Y. Wang (2010). Histogram similarity measure using variable bin size distance. *Computer Vision and Image Understanding 114*(8), 981–989.

Niblack, C. W. et al. (1993). QBIC project : Querying Images By content, Using Color, Texture and Shape. In *Proceedings of the SPIE Conference on Storage and Retrieval for Image and Video Databases*, Volume 1908, pp. 173–187.

Rubner, Y., C. Tomasi, et L. J. Guibas (2000). The Earth Mover's Distance as a metric for image retrieval. *International Journal of Computer Vision 40*, 99–121.

Strelkov, V. V. (2008). A new similarity measure for histogram comparison and its application in time series analysis. *Pattern Recognition Letters 29*(13), 1768–1774.

Summary

The usual distances defined for histogram comparison are generally devoted either to ordinal histograms (related to entities equipped with a total ordering) or nominal histograms (related to entities which can not be compared). However, there does not exist any distance for nominal histograms related to entities whose semantic/thematic proximity can be quantified. In this article, we propose a new distance devoted to this issue.

Extraction de Dépendances Fonctionnelles Approximatives: une Approche Incrémentale

Ekaterina Simonenko* et Noël Novelli**

* LRI-CNRS UMR 8623
Université Paris-Sud XI; F-91405 Orsay Cedex
ekaterina.simonenko@lri.fr
** LIF-CNRS UMR 6166 – Case 901
Université d'Aix-Marseille; Faculté des Sciences de Luminy;
F-13288 Marseille Cedex 9
noel.novelli@lif.univ-mrs.fr

Résumé. La découverte de dépendances fonctionnelles (DF) à partir d'une relation existante est une technique importante pour l'analyse de Bases de Données. L'ensemble des DF exactes ou approximatives extraites par les algorithmes existants est valide tant que la relation n'est pas modifiée. Ceci est insuffisant pour des situations réelles où les relations sont constamment mises à jour.
Nous proposons une approche incrémentale qui maintiens à jour l'ensemble des DF valides, exactes ou approximatives selon une erreur donnée, quand des tuples sont insérés et supprimés. Les résultats expérimentaux indiquent que lors de l'extraction de DF à partir d'une relation continuellement modifiée, les algorithmes existants sont sensiblement dépassés par notre stratégie incrémentale.

1 Contexte

Les Dépendances Fonctionnelles (DF) représentent les contraintes d'intégrité les plus courantes et les plus importantes en Bases de Données (Mannila et Räihä (1994)). Une DF entre 2 attributs (X, Y) notée $X \rightarrow Y$ est vraie dans une relation si les valeurs de Y sont totalement déterminées par les valeurs de X (Codd (1970)). Le problème de l'extraction de DF est le suivant : "Étant donnée une relation r, trouver toutes les DF qui sont valides dans r". Les Dépendances Fonctionnelles Approximatives (DFA) généralisent les DF et sont définies comme "les DF qui sont presque valides dans r, i.e. quelques tuples doivent être retirés de la relation r pour que la DF $X \rightarrow Y$ soit vraie dans r" (Kivinen et Mannila (1995)). Des DFA apparaissent dans les relations s'il existe une dépendance naturelle entre les attributs mais certains tuples contiennent des erreurs ou représentent une exception. Comme des erreurs peuvent être présentes dans les BD, les DF approximatives sont particulièrement intéressantes.
Récemment, la taille des bases de données a augmenté significativement voire de façon infinie pour les flux de données, rendant les algorithmes existants inefficaces. Ces approches ne peuvent considérer que des relations figées. Quand un tuple est ajouté ou supprimé, l'ensemble des DF valides doit être recalculé.
Les algorithmes les plus efficaces pour l'inférence de DF sont TANE (Huhtala et al. (1998)),

DEPMINER (Lopes et al. (2000)), FASTFDS (Wyss et al. (2001)) et FUN (Novelli et Cicchetti (2001)). Ces approches extraient l'ensemble des DF minimales et non triviales, la couverture canonique de DF. Cette dernière est équivalente à l'ensemble des DF (Codd (1970)). D'un point de vue formel, DEPMINER et FASTFDS sont basés sur la caractérisation des parties gauches des dépendances minimales non triviales comme étant l'ensemble des transversaux minimaux d'un hypergraphe (Mannila et Räihä (1994)). En revanche, TANE et FUN énumèrent toutes les sources de DF possibles et déterminent si ces sources induisent des DF minimales non triviales. Concernant les DF Approximatives (DFA), leur extraction est faite de façon efficace par DEPMINER, TANE et FUN. TANE et FUN reprennent l'une des mesures d'erreur définie dans Mannila et Räihä (1994); Kivinen et Mannila (1995) (appelé $g3$) comme la proportion minimale de tuples qu'il suffit de supprimer de la relation pour que la DF soit satisfaite par tous les tuples restants. L'erreur est égale à 0 si la DF est exacte, et proche de 1 si la DF n'est vérifiée que par un petit nombre de tuples.
Ces travaux permettent l'extraction des DF d'une relation figée. A notre connaissance seule l'approche INCFDS (Gasmi (2010)) prend en compte les modifications (insertion de tuples uniquement) de la relation pour maintenir à jour l'ensemble des DF exactes valides. L'approche est basée sur la détection des modifications induites lors de l'insertion de tuples dans la relation sur les ensembles en accord (puis bien sûr sur les maximaux et leurs complémentaires).

2 Approche Incrémentale pour la découverte de DFA

Nous désignons par R un schéma de relation et r une relation sur R. Un *sous-ensemble maximal* S de $X \subseteq R$ est un sous-ensemble de X tel que $S \subset X$ et $|S| = |X| - 1$. La cardinalité d'une combinaison d'attributs X dans r, noté $|X|_r$ représente le nombre de valeurs distinctes de X dans r. $|r|$ représente la cardinalité de r c'est-à-dire le nombre de tuples de r. Pour chaque combinaison d'attributs X, l'ensemble de ses valeurs réelles, le domaine actif, dans la relation r est noté $ADom_r(X)$.

2.1 Définitions

Définition 1 *Dépendances Fonctionnelles (DF)*
Soit $X, A \subset R$ une combinaison d'attributs. La dépendance fonctionnelle entre X et A, notée $X \rightarrow A$, est valide dans r si est seulement si $\forall\, t_1, t_2$ deux tuples de r, si $t_1[X] = t_2[X]$, alors $t_1[A] = t_2[A]$.
$X \rightarrow A$ est une DF minimale si et seulement si : $\forall\, X' \subset X$, $X' \not\rightarrow A$.
$X \rightarrow A$ est une DF non-triviale si et seulement si : $A \not\subseteq X$.

Définition 2 *(Kivinen et Mannila (1995)) Dépendances Fonctionnelles Approximatives (DFA)*
Une DF entre X et A est dite approximative suivant l'erreur ε, notée $X \rightarrow^{\varepsilon} A$, si $g3(X \rightarrow A) \leq \varepsilon$, où la fonction $g3$ calcule la proportion, par rapport à $|r|$, des tuples qu'il faut retirer à r pour que la DF $X \rightarrow A$ soit valide.

2.2 Extraction Incrémentale de DFA

Soit $Remove_r(X \rightarrow^{\varepsilon} A)$, où X désigne une combinaison d'attributs et A un attribut, un ensemble minimal de tuples à retirer de la relation r pour que la DF exacte $(X \rightarrow A)$ soit

valide. Soit $e_r(X \rightarrow^{\varepsilon} A) = |Remove_r(X \rightarrow^{\varepsilon} A)|$, le nombre de tuples à enlever de r pour rendre la DF $(X \rightarrow A)$ valide. Dans la suite, $e_r(X \rightarrow^{\varepsilon} A)$ est souvent appelé "erreur".
Le Théorème suivant décrit l'influence de l'insertion d'un tuple t sur $e_r(X \rightarrow^{\varepsilon} A)$.

Théorème 1 *Étant donnée une DFA $X \rightarrow^{\varepsilon} A$ valide sur r, $e_{r \cup t}(X \rightarrow^{\varepsilon} A) \geq e_r(X \rightarrow^{\varepsilon} A)$*

Preuve : Soit t le tuple qui va être inséré. Considérons $t[X]$ la valeur de t sur la combinaison d'attributs X.
Si $t[X] \notin ADom_r(X)$: le tuple t ne peut pas violer la DF $X \rightarrow A$, puisque la valeur de t sur X est nouvelle. Donc, $e_r(X \rightarrow^{\varepsilon} A)$ reste le même : $e_{r \cup t}(X \rightarrow^{\varepsilon} A) = e_r(X \rightarrow^{\varepsilon} A)$.
Sinon $t[X] \in ADom_r(X)$:

1. Si $t[XA] \notin ADom_r(XA)$, on peut en déduire que soit $t[A] \notin ADom_r(A)$, soit $t[A]$ n'a encore jamais été associée à $t[X]$. Dans les 2 cas t viole la DF $X \rightarrow A$. Or, $t[XA] \notin ADom_r(XA) \Rightarrow t[XA] \notin ADom(Remove_r(X \rightarrow^{\varepsilon} A)[XA])$, et donc $Remove_{r \cup t}(X \rightarrow^{\varepsilon} A) = Remove_r(X \rightarrow^{\varepsilon} A) \cup t$ et $e_{r \cup t}(X \rightarrow^{\varepsilon} A) = e_r(X \rightarrow^{\varepsilon} A) + 1$. Il n'est donc pas nécessaire de recalculer $e_r(X \rightarrow^{\varepsilon} A)$.
2. Si $t[XA] \in ADom_r(XA)$:
   (a) Si $t[XA] \notin ADom(Remove_{r \cup t}(X \rightarrow^{\varepsilon} A)[XA])$,
   alors $t[XA] \in ADom(\ Remove_{r \cup t}(X \rightarrow^{\varepsilon} A)[XA])$ puisque
   $t[XA] \in ADom_r(XA)$, donc t satisfait la DF $X \rightarrow A$ et $e_r(X \rightarrow^{\varepsilon} A)$ ne change pas.
   (b) Si $t[XA] \in ADom(Remove_{r \cup t}(X \rightarrow^{\varepsilon} A)[XA])$, ce qui signifie que les tuples avec la même valeur sur XA que t, sont considérés comme non-vérifiant la DF $X \rightarrow A$. Malheureusement, ajouter t dans $Remove_r(X \rightarrow^{\varepsilon} A)$ ne suffit pas, car après avoir incrémenté la cardinalité de $Remove_r(\ X \rightarrow^{\varepsilon} A)$, il est nécessaire de vérifier sa minimalité ce qui implique le recalcule complet de $Remove_{r \cup t}(X \rightarrow^{\varepsilon} A)[XA]$. □

Il est clair que lors de l'insertion d'un tuple, l'erreur ne peut qu'augmenter de 1. De plus, la preuve de ce théorème énumère tous les cas à considérer et permet donc de ne recalculer $e_{r \cup t}(X \rightarrow^{\varepsilon} A)$ que lorsque c'est strictement nécessaire.

Le théorème suivant donne le résultat analogue pour la suppression d'un tuple.

Théorème 2 *Étant donnée une DFA $X \rightarrow^{\varepsilon} A$ valide sur r, $e_{r \setminus t}(X \rightarrow^{\varepsilon} A) \leq e_r(X \rightarrow^{\varepsilon} A)$*

Preuve : La preuve de ce théorème n'est pas explicitée ici par manque de place mais elle suit le même cheminement que celle du théorème 1. □
Pour maintenir l'ensemble des DFA valides dans une relation selon une erreur ε donnée, une représentation interne basée sur la caractérisation des DFA introduite dans Novelli (2000) est utilisée. Celle-ci est une table contenant pour chaque combinaison d'attributs X, sa quasi-fermeture approximative X_r° et sa fermeture approximative $X_r^{\oplus}$. Quand un tuple est ajouté ou supprimé d'une relation, pour certaines DFA, l'erreur $e_r(X \rightarrow^{\varepsilon} A)$ change, et donc la représentation interne doit être mise à jour pour découvrir le nouvel ensemble de DFA minimales valides.

2.3 Algorithme AFD-DYNAMICUPDATE

L'algorithme, AFD-DYNAMICUPDATE, que nous décrivons ci-après maintient à jour l'ensemble des DFA valides dans une relation suivant une erreur donnée. Il s'appuie sur les théorèmes 1 et 2 ainsi que sur leur preuve pour minimiser le nombre de recalcul d'erreur de chaque DFA. Le recalcule de l'erreur n'est appelé que si c'est strictement nécessaire. Lorsqu'un tuple est ajouté, la représentation correspondante doit être mise à jour. Cela consiste à vérifier quelles DFA doivent être ajoutées ou supprimées de l'ensemble des DFA valides. Une approche par niveau est utilisée pour le parcours des candidats.
Pour la suppression, seule la fonction `RecalculateError` est modifiée suivant le Théorème 2.

Algorithm 1 AFD-DYNAMICUPDATE (R, $TupleInQuestion$, ε, $g3[]$ $Levels$)

Input :
R : L'ensemble d'attributs de la relation considérée
$TupleInQuestion$: Le tuple inséré
ε : L'erreur maximale admissible
Input/Output :
$g3[]$: Les erreurs (par rapport à $|r|$) de DF $X \rightarrow^{\varepsilon} A$
$Levels$: La représentation

for all $i \in [1..Levels.NbLevel]$ **do**
  for all candidate $l \in L_i$ **do**
    for all subset $s \subset l.candidate$ / $|s| = |l.candidate| - 1$ **do**
      $X := s.candidate$
      $A := \{l.candidate - s.candidate\})$
      $g3[X \rightarrow^{\varepsilon} A] := RecalculateError(l, Remove_r(X \rightarrow^{\varepsilon} A), ErrorChanges)$
      if $ErrorChanges$ **then**
        if $g3[X \rightarrow^{\varepsilon} A] \leq \varepsilon$ **then**
          if $A \notin s.closure$ **then**
            $s.closure := s.closure \cup A$
            for all superset $S \supset s.candidate$ / $|s| = |l.candidate| + 1$ **do**
              $UpdateQuasiClosureAdd(S, A)$
        else if $g3[X \rightarrow^{\varepsilon} A] > \varepsilon$ **then**
          if $A \in s.closure$ **then**
            $s.closure := s.closure - A$
            for all superset $S \supset s.candidate$ / $|s| = |l.candidate| + 1$ **do**
              $UpdateQuasiClosureDelete(S, A)$
  DisplayFDs($L_i - 1$)

2.4 Expérimentations pour AFD-DYNAMICUPDATE

L'algorithme a été implémenté en C++ avec QT4. Plusieurs expérimentations ont été réalisées sur un ordinateur équipé d'un processeur Pentium 4 cadencé à 3GHz avec 1Go de RAM. Pour les comparaisons entre AFD-DYNAMICUPDATE et FUN, nous commençons à partir d'une relation vide et ajoutons un par un chaque tuple (donc #tuples fois) [1]. Pour chaque

1. Les résultats obtenus pour les suppressions sont similaires à ceux obtenus pour l'ajout.

tuple inséré, l'approche FUN est exécutée. Les temps d'exécution de FUN donnés dans les comparaisons correspondent à la somme des temps de calcul de FUN pour chaque relation modifiée. Les figures 1(A) et 1(B) détaillent pour différentes valeurs des paramètres des relations synthétiques, les temps d'exécution de AFD-DYNAMICUPDATE. La figure 1(A) montre que AFD-DYNAMICUPDATE est indépendant de la corrélation des données (30%, 50%, 70%), et bien sûr qu'il est exponentiel au nombre d'attributs. La figure 1(B) illustre que l'algorithme est linéaire au nombre de tuples ajoutés dans la relation.

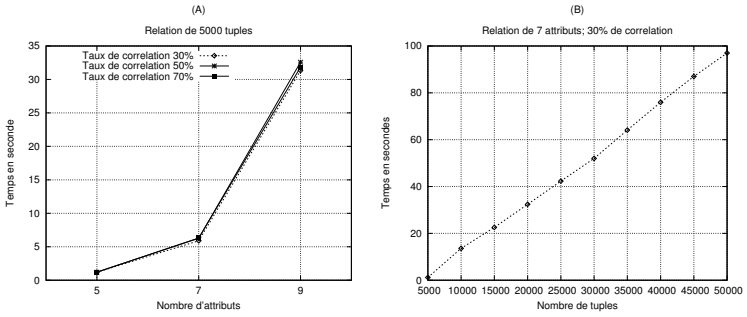


FIG. 1 – *Temps d'exécution pour différents nombres d'attributs, différentes corrélations, et différents nombres de tuples*

La table suivante décrit les caractéristiques des données réelles et les temps d'exécutions avec AFD-DYNAMICUPDATE et FUN.

Relation names	#attributes	#tuples	AFD-DYNAMICUPDATE	FUN
TombNecropolis	7	1 846	5,700s	82,770s
someCompanies	6	40 316	11m15s	1h 27m

Le tableau ci-dessus est très clair : notre approche est beaucoup plus efficace que FUN pour le calcul incrémental de Dépendances Fonctionnelles.

3 Conclusion et perspectives

Nous avons proposé une nouvelle approche pour l'extraction de DFA et un algorithme associé nommé AFD-DYNAMICUPDATE. Contrairement à d'autres algorithmes, celui-ci est une approche incrémentale qui permet la mise à jour de l'ensemble des DFA valides quand la relation évolue. Nous caractérisons l'influence de l'évolution d'une relation sur l'ensemble des DFA valides. Pour cela, nous n'utilisons pas la relation, mais une représentation de celle-ci, ce qui rend l'algorithme efficace et exploitable.
Pour montrer les avantages et la faisabilité de notre approche, nous avons comparé AFD-DYNAMICUPDATE avec FUN. Les résultats des expérimentations obtenus sous les mêmes conditions (cf. Section 2.4) montrent de bonnes propriétés de passage à échelle de l'approche lorsque la relation change (ajout ou suppression de tuples).
Par ailleurs, ce travail sera utilisé pour la détection visuelle de changements d'habitude ou de

détections d'erreurs. Un outil de visualisation *on line* et *off line* (à l'aide de scénarii, séquence d'instructions d'ajout et de suppression de tuples) est en cours de réalisation pour visualiser l'évolution de l'ensemble des DF valides au cours du temps. Cet outil montrera les différents ensembles de DF (exactes ou approximatives) valides pour une relation à plusieurs instants ce qui permettra de mieux comprendre les changements d'habitudes ou les apparitions d'erreurs.

Références

Codd, E. (1970). A Relational Model of Data for Large Shared Data Banks. *Communication of the ACM 13*(6), 377–387.

Gasmi, G. (2010). Incfds : un nouvel algorithme d'inférence incrémentale des dépendances fonctionnelles. In *EGC'10*, pp. 303–314.

Huhtala, Y., J. Karkkainen, P. Porkka, et H. Toivonen (1998). Efficient Discovery of Functional and Approximate Dependencies. In *ICDE*, pp. 392–401.

Kivinen, J. et H. Mannila (1995). Approximate Dependency Inference from Relations. *Theoretical Computer Science 149*(1), 129–149.

Lopes, S., J. Petit, et L. Lakhal (2000). Efficient Discovery of Functional Dependencies and Armstrong Relations. In *EDBT*, pp. 350–364.

Mannila, H. et K. Räihä (1994). Algorithms for Inferring Functional Dependencies from Relations. *Data and Knowledge Engineering 12*(1), 83–99.

Novelli, N. (2000). *Extraction de Dépendances Fonctionnelles dans les Bases de Données : une Approche Data Mining*. Ph. D. thesis, Université Aix-Marseille II.

Novelli, N. et R. Cicchetti (2001). Functional and Embedded Dependency Inference : A Data Mining Point of View. *Information Systems 26*(7), 477–506.

Wyss, C. M., C. Giannella, et E. L. Robertson (2001). Fastfds : A heuristic-driven, depth-first algorithm for mining functional dependencies from relation instances - extended abstract. In *DaWaK*, pp. 101–110.

Summary

Functional Dependency (FD) inference from existing relations is an important data base analysis technique. The problem has been treated by using data mining approach, by the algorithms TANE, DEPMINER, FASTFDS et FUN. Since the FD set, inferred by these algorithms is valid as long as the relation remains unchanged, they become efficient in the real life situations when the relation is constantly updated. It is particularly important to be able to extract approximate FDs, allowing to take into account errors and exceptions in the given data. We propose an incremental approach, allowing to update the valid set of approximate FDs, while tuples are inserted or deleted. The algorithm employs a level-wise strategy to keep the internal representation of the relation up-to-date. Experimental results show that during FDs extraction from a constantly updated relation, existing algorithms are significantly outperformed by our incremental approach.