Un assistant utilisateur pour le choix et le paramétrage des méthodes de fouille visuelle de données

Abdelheq Et-tahir Guettala*, Fatma Bouali***,*, Christiane Guinot**,*, Gilles Venturini*

*Université François Rabelais Tours, Laboratoire d'Informatique
64 avenue Jean Portalis, 37200 Tours, France
{abdelheq.guettala,venturini}@univ-tours.fr
http://www.antsearch.univ-tours.fr
**CERIES, 20 rue Victor Noir, 92521 Neuilly-sur-Seine Cedex
christiane.guinot@ceries-lab.com
***Université de Lille2, IUT, Dpt STID
25-27 Rue du Maréchal Foch, 59100 Roubaix, France
Fatma.Bouali@univ-lille2.fr

Résumé. Nous nous intéressons dans cet article au problème de l'automatisation du processus de choix et de paramétrage des visualisations en fouille visuelle de données. Pour résoudre ce problème, nous avons développé un assistant utilisateur qui effectue deux étapes : à partir des objectifs annoncés par l'utilisateur et des caractéristiques de ses données, le système commence par proposer à l'utilisateur différents appariements entre la base de données à visualiser et les visualisations qu'il gère. Ces appariements sont générés par une heuristique utilisant une base de connaissances sur les visualisations et la perception visuelle. Ensuite, afin d'affiner les différents paramétrages suggérés par le système, nous utilisons un algorithme génétique interactif qui permet aux utilisateurs d'évaluer et d'ajuster visuellement ces paramétrages. Nous présentons une évaluation utilisateur qui montre l'intérêt de notre système pour deux tâches.

1 Introduction

En général, les systèmes de visualisation sont exploités par des experts du domaine pour accomplir des tâches d'exploration et d'analyse de leurs données dans des buts précis. Cependant, ces systèmes peuvent se transformer en outils complexes pour des utilisateurs novices, à la fois à cause des interfaces mais également à cause du temps passé à trouver un paramétrage de la visualisation qui réponde au mieux aux besoins des utilisateurs. Un véritable système de fouille visuelle de données ne doit pas exiger des connaissances de la part des utilisateurs, mais plutôt les guider dans le processus d'exploration et d'analyse de leur ensemble de données (Wong, 1999). Si l'on se limite au domaine de la fouille visuelle de données et de la visualisation d'information, il existe peu d'assistants utilisateur cités dans la littérature qui utilisent un processus automatisé pour aider les utilisateurs dans le choix et le paramétrage des visualisations. (Mackinlay, 1986) a développé un outil de présentation graphique (APT) à base

de règles pour automatiser le processus de visualisation. La principale fonctionnalité de ce système consiste à utiliser un algorithme d'inférence pour synthétiser la représentation graphique d'une base de données. ViA est un autre assistant visuel interactif Healey et al. (2008) plus récent qui a été conçu dans le même but mais qui utilise un autre type de raisonnement. ViA s'appuie sur deux modules dont le premier comporte un moteur de recherche qui sert à générer des appariements entre les attributs de données et les attributs visuels. Le deuxième module regroupe un ensemble de moteurs d'évaluations dont chacun est responsable d'évaluer un seul attribut visuel (couleur, luminance, etc.). Dans cet article, nous allons donc nous intéresser aux assistants utilisateur pour la fouille visuelle de données. L'objectif de ces assistants est de permettre à l'utilisateur de choisir et de paramétrer automatiquement des visualisations. Ils s'appuient généralement sur une modélisation des visualisations, des données et des objectifs de l'utilisateur. Nous présentons dans la section 2 notre assistant utilisateur qui permet d'une part de guider des utilisateurs novices dans le processus de choix et de paramétrage automatique de visualisations, et d'autre part d'ajuster interactivement le paramétrage initialement suggéré avec un algorithme génétique interactif. La section 3 conclut par une discussion des résultats et des perspectives faisant suite à ce travail.

2 Modèle proposé

2.1 Modèle des données, des visualisations et des objectifs utilisateur

Afin d'extraire l'ensemble des caractéristiques les plus pertinentes des données utilisateur, nous avons défini un modèle pour les formaliser. Notons $D = \{d_1, ..., d_n\}$ la base des n données à visualiser. Chaque donnée d_i est définie par k attributs de données A_1, .., A_k dont chacun est caractérisé par un type t_i et une importance u_i. Notre système gère différents types de données (numérique/quantitatif, symbolique/ordinal ou nominal, temporel, image, son, texte, lien Web, etc.). La valeur de l'importance u_i est définie dans l'intervalle $[0, 100]$. Elle représente l'intérêt que porte l'utilisateur à l'attribut A_i, et peut être déterminée manuellement par l'utilisateur en fonction de ses connaissances a priori ou automatiquement via des méthodes de sélection de variables. Si aucune connaissance n'est disponible alors les u_i prennent toutes la même valeur.

En s'inspirant principalement des travaux de (Bertin, 1983) et de (Card et al., 1999), nous avons développé une base de connaissances qui nous permet de formaliser et structurer les différentes visualisations gérées par notre système. Nous définissons une visualisation V_i par ses éléments graphiques (points, lignes, formes 2D ou 3D). Chaque élément graphique de V_i est caractérisé par ses attributs visuels, et nous notons l'ensemble de tous les attributs visuels de V_i par A_{i1}, .., A_{im}. A chaque attribut visuel A_{ij} est associé un type visuel t_{ij} (position, taille, couleur, etc.), un type d'attribut de données dont la valeur sera utilisée pour renseigner l'attribut visuel, et un degré d'importance v_{ij}. Les valeurs v_{ij} sont déterminées selon la capacité que peut avoir un attribut visuel à représenter tel ou tel type d'attribut de données. Pour cela, nous définissons une matrice d'importance "type d'attribut visuel $\times$ type d'attribut de données" dont les valeurs sont déterminées à partir d'études comme (Mackinlay, 1986).

De plus, pour chacune des visualisations de la base de connaissances, nous décrivons quels sont les objectifs qu'elle pourra atteindre, sa dimension visuelle (1D, 2D, 3D) et aussi sa catégorie visuelle (temporelle, relationnelle, etc.). Pour représenter les objectifs O_j qu'une visua-

lisation V_i va permettre d'atteindre, nous attribuons à chaque objectif un poids o_{ij}. Ces poids sont calculés en fonction des caractéristiques et travaux connus sur chacune des visualisations représentées.

2.2 Algorithme d'appariement et choix d'une visualisation

La phase d'appariement entre les visualisations et les données se déroule en deux étapes. La première étape consiste à sélectionner les visualisations qui sont les plus compatibles avec les objectifs de l'utilisateur. Ces derniers sont spécifiés via une liste préétablie (découvrir des classes, avoir une vue d'ensemble, etc.) présentée sous forme de questionnaire. La deuxième étape d'appariement consiste à sélectionner, selon t_{ij} et t_i ainsi que l'ordre décroissant de v_{ij} et u_i, pour chaque attribut visuel $A_{ij(j=1..m)}$ d'une visualisation V_i, un attribut de données $A_{i(i=1..k)}$. L'assistant calcule pour chaque étape un score d'appariement. Pour une visualisation donnée, ce score est un produit scalaire entre les importances (des objectifs/des attributs de données) annoncés par l'utilisateur et les poids (des objectifs de la visualisation/des attributs visuels) renseignés dans la base de connaissances. A la fin de cette phase, plusieurs visualisations avec chacune un paramétrage sont proposées à l'utilisateur (voir figure1). Pour l'aider à choisir une visualisation, l'assistant prévisualise chacune d'elles avec les données D en ordonnant les visualisations de manière décroissante selon le score d'appariement. La capacité d'explorer de manière comparative les visualisations sur les données de l'utilisateur ainsi que la possibilité de tester dynamiquement toutes les fonctionnalités des visualisations sur ses données est l'un des avantages de notre assistant. En plus, lorsque cela est possible, nous avons ajouté la possibilité de sélectionner des données dans une visualisation et de voir apparaître les données sélectionnées dans les autres grâce à la technique nommée "brushing" et décrite dans (Becker et Cleveland, 1987). Cette partie de notre assistant peut donc être vue non seulement comme un outil permettant à l'utilisateur de choisir interactivement une visualisation sans avoir à spécifier manuellement un paramétrage, mais également comme une interface multi-visualisations.

2.3 Algorithme génétique interactif (AGI)

Une fois que l'utilisateur a choisi une visualisation V_i, il se peut que son appariement ne soit pas parfaitement adapté à ses besoins ou qu'il souhaite le modifier. Pour cette raison, nous avons dévéloppé une deuxième interface qui permet d'améliorer encore ce paramétrage grâce à une étape interactive. Pour cette étape, nous avons défini un algorithme génétique interactif (AGI) (Dawkins, 1986), dans lequel l'utilisateur remplace la fonction de qualité utilisée dans les algorithmes génétiques (Holland, 1975) pour évaluer les individus de la population P. Dans notre AGI, un individu I de $P(t)$ va représenter un paramétrage possible de la visualisation V_i sous la forme d'un vecteur de poids (importance) de l'ensemble des attributs de données A_i. Ce vecteur vient donc influencer directement l'appariement avec les attributs visuels. Le codage des individus que nous avons utilisé est donc une représentation en nombres réels (Wright, 1991), chaque gène représentant alors la nouvelle importance de l'attribut de données correspondant. Chaque individu va être représenté visuellement en chargeant les données dans la visualisation avec le paramétrage spécifié. Pour cela, nous utilisons une interface dans laquelle 8 paramétrages (donc 8 visualisations) seront représentés (voir figure2). A partir du paramétrage initial, 8 individus (I_1, .., I_8) sont générés. Les 8 visualisations correspondantes

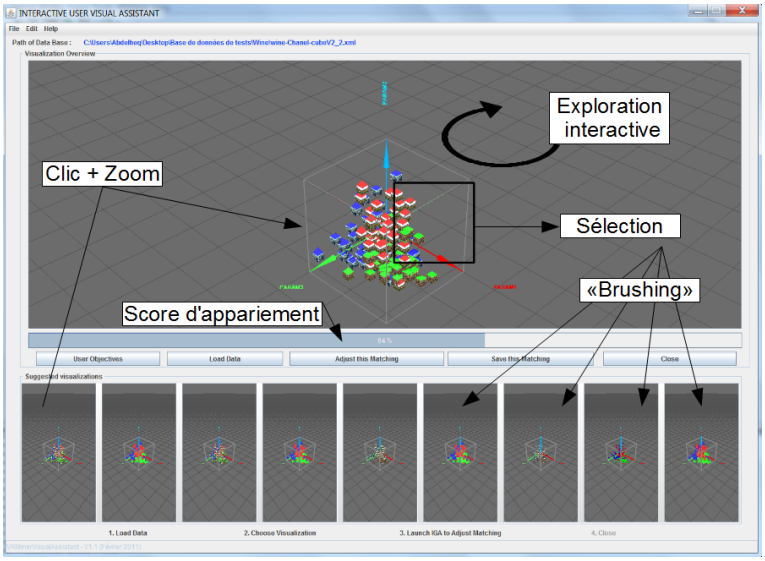

FIG. 1 – *Résultats proposés par l'assistant après le chargement de l'ensemble D.*

sont présentées à l'utilisateur en les appliquant sur D. Ainsi, l'utilisateur peut sélectionner les visualisations ayant les paramétrages qui lui paraissent les meilleurs, et qui peuvent être enregistrés pour une utilisation ultérieure. Ils servent aussi à générer de nouveaux paramétrages en appliquant les opérateurs génétiques (mutation et croisement). La manière d'appliquer ces derniers dépend du nombre d'individus sélectionnés par l'utilisateur et permet de réajuster les vecteurs de poids qui leurs correspondent.

3 Résultats et conclusions

Dans le but d'expérimenter notre système nous avons réalisé une évaluation utilisateur dans laquelle nous avons comparé les principaux avantages et inconvénients de notre assistant par rapport à un autre système de visualisation appelé VRMiner (Azzag et al., 2005) qui s'appuie sur une interface avec un paramétrage manuel. Nous avons défini pour cela deux tâches. La tâche T1 consiste à obtenir une visualisation fixée d'avance et a donc pour but de tester la première interface de notre outil. On indique à l'utilisateur que l'on souhaite représenter 4 attributs numériques, un attribut image et un attribut texte, et il doit obtenir une visualisation représentant ces données. Nous avons généré une base spécifique pour cette tâche avec 150 données et les attributs correspondants (4 numériques, 1 image, 1 texte, 1 classe). La tâche T2 consiste à obtenir une visualisation la moins bruitée possible et dans laquelle trois classes se distinguent. Les 150 données sont décrites par 30 attributs numériques et 1 attribut classe. Le bruit dans les attributs est généré de manière croissante et l'ordre des attributs dans la base est aléatoire. Trois attributs parmi les 30 sont non bruités et représentent la réponse exacte. Cette tâche a donc pour objectif de tester l'AGI et l'amélioration du paramétrage initial. Pour chaque tâche

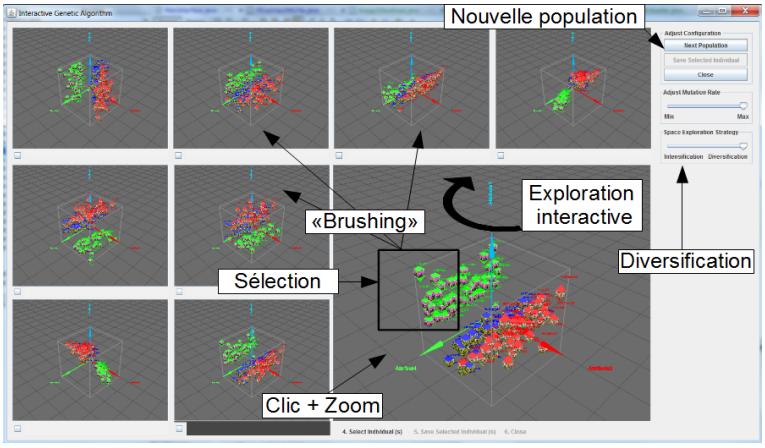

FIG. 2 – *Interface d'optimisation génétique du paramétrage.*

nous avons utilisé la même base de données pour les deux systèmes mais avec un renommage des variables et un changement d'ordre afin d'éviter un effet d'apprentissage. L'ordre des tests a été randomisé. Notre protocole a porté sur 15 participants dont l'âge varie de 21 ans à 31 ans et dont le niveau d'étude est supérieur à bac+2 en Informatique. La réalisation d'un test a duré en moyenne une heure et demi pour chaque participant. Les résultats obtenus pour la tâche T1, montrent que tous les utilisateurs ont résolu cette tâche avec les deux systèmes. Cependant, la durée passée par les participant sur T1 avec l'assistant est 31 ± 16, tandis qu'avec l'autre système cette données est 145 ± 90. Nous constatons ainsi que les temps mis pour répondre à T1 sont très nettement en faveur de l'assistant. Pour la tâche T2, 6 personnes sur les 15 ont abandonné avec l'interface manuelle. Pour cette tâche nous avons défini une mesure de qualité pour évaluer graduellement les réponses des utilisateurs. Cette mesure de qualité donne un score à chaque attribut, ce score étant inversement proportionnel au bruit que nous y avons rajouté. Si les attributs non bruités ont été trouvé, la qualité est de 1, et si les attributs les plus bruités ont été choisis, la qualité vaut 0. Les valeurs de qualité obtenues sont de 0.37 ± 0.34 sans l'assistant, et de 0.64 ± 0.10 avec l'assistant, ce qui montre que la qualité des réponses avec notre assistant est supérieure à celles obtenues sans assistant. En fait, nous avons remarqué que les utilisateurs qui n'ont pas pu accomplir T2 sans assistant réalisaient de nombreux essais d'appariement manuel mais comme il s'agit de tester 30 variables sur troix axes (X, Y, Z) pour trouver la réponse correcte, ils abondonnaient. Par contre, l'utilisation de l'AGI facilite et accèlere la convergence vers les meilleurs réponses. Cela met en avant les capacités de notre outil pour l'ajustement des paramètres, la sélection d'attributs et l'exploration des données.

L'un des apports majeurs de notre outil est qu'il propose une nouvelle technique d'automatisation du processus de visualisation qui repose sur deux étapes. La première étape permet de suggérer plusieurs visualisations avec les meilleurs paramétrages possible appliqués directement sur les données utilisateur. La deuxième étape consiste à optimiser et affiner ces paramétrages de manière visuelle et interactive. Un autre avantage de notre système est qu'il s'appuie sur une architecture générique pouvant intégrer facilement de nouvelles visualisa-

tions, et nous allons le compléter avec des visualisations 2D classiques. Notre outil peut également servir pour explorer de manière interactive une base de données, soit avec un mode multi-visualisations comme dans la première interface, soit avec un mode d'exploration interactive comme dans la deuxième interface. L'analyse des résultats de l'évaluation utilisateur que nous avons réalisée montre que les interfaces que nous avons développées sont avantageuses pour des utilisateurs novices car elles n'exigent pas de connaissances a priori de leur part. Les utilisateurs ont montré aussi de manière subjective l'intérêt qu'ils ont porté pour l'utilisation ultérieure de notre système. Une perspective, en cours d'étude également, consiste à prendre en compte les retours utilisateur pour améliorer les recommandations du système.

Références

Azzag, H., F. Picarougne, C. Guinot, et G. Venturini (2005). Vrminer : A tool for multimedia database mining with virtual reality. *Processing and Managing Complex Data for Decision Support* (Ea 2101), 318–339.

Becker, R. A. et W. S. Cleveland (1987). Brushing scatterplots. *Technometrics 29*(2), 127–142.

Bertin, J. (1983). *Semiology of graphics*. Berlin: University of Wisconsin Press.

Card, S. K., J. D. Mackinlay, et B. Shneiderman (1999). *Readings in Information Visualization: Using Vision to Think (Interactive Technologies)*. Morgan Kaufmann.

Dawkins, R. (1986). *The Blind Watchmaker*. San Mateo: Norton.

Healey, C., S. Kocherlakota, V. Rao, R. Mehta, et R. St Amant (2008). Visual perception and mixed-initiative interaction for assisted visualization design. *IEEE transactions on visualization and computer graphics 14*, 396–411.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

Mackinlay, J. (1986). Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics 5*, 110–141.

Wong, P. C. (1999). Guest editor's introduction: Visual data mining. *IEEE Computer Graphics and Applications 19*, 20–21.

Wright, A. (1991). Genetic algorithms for real parameter optimization. *Foundations of genetic algorithms 1*, 205–218.

Summary

We deal in this paper with the problem of automating the process of choosing a visualization and its parameters in visual data mining. To solve this problem, we have developed a user assistant that performs 2 steps: the system starts by suggesting to users different matchings between their database and the possible visualizations. These matchings are generated by using a knowledge-based heuristic. Then, in order to refine the different parameter settings suggested by the system, we use an interactive genetic algorithm which allows users to visually evaluate and adjust these parameters. We present a user evaluation that confirms the interest of our system in two tasks.