Approche connexionniste pour l'analyse des données issues d'usage d'Internet : Classification et Visualisation

Khalid Benabdeslem* et Younès Bennani**

*Université Lyon1/LIESP, 8 Boulevard Niels Bohr, 69622 Villeurbanne, France
kbenabde@bat710.univ-lyon1.fr
**Université Paris13/LIPN, 99 Avenue J-B. Clément, 93430 Villetaneuse, France
younes.bennani@lipn.univ-paris13.fr

Résumé. Dans ce papier, nous présentons une chaîne complète de fouille de données comportementales issues des navigations des clients de sites web commerciaux. Nous présentons plus particulièrement, le développement d'une approche d'apprentissage non supervisé pour ce type de données stockées sous forme de traces de navigation dans des fichiers Log.
La première partie de cette étude concerne le problème du codage des données qui seront ensuite utilisées pour l'analyse. En effet, actuellement, les sites web sont dynamiques et les pages ne peuvent pas être caractérisées par des variables fixes comme : la hiérarchie des adresses URL, le contenu, etc. Elles sont représentées par des identificateurs numériques n'ayant aucun « sens » et qui servent comme adresses de récupération des informations dans des bases de données pour remplir les contenus des pages. Pour cette raison, nous proposons une nouvelle méthode de codage de sessions à partir du fichier Log. Cette technique consiste à caractériser une page donnée par un vecteur de poids d'importance de passage, i.e. par ses poids de précédence et de successions relatives à toutes les autres pages qui apparaissent dans le fichier Log.
Dans la deuxième partie de ce travail, nous analysons les propriétés des cartes topologiques de Kohonen et nous proposons une version adaptée aux données comportementales. Cette étape nous permet (1) de construire une cartographie du site web tel qu'il est aperçu par les clients (2) de regrouper les pages pour un objectif de codage de sessions (3) et de projeter les interactions des clients du fichier Log sur la cartographie sous forme de trajectoires symbolisant leurs comportements.

1 Introduction

Les sites Web représentent actuellement une véritable source de production de grands volumes d'informations. Cependant, ce gisement d'information ne représente pas une évidence de compréhension des utilisateurs qui se retrouvent généralement perdus devant de telles quantités d'informations, Zeboulon et al. (2001). Une technique de e-Mining est donc nécessaire pour comprendre les interactions des utilisateurs pour répondre aux mieux à leurs besoins prioritaires. Le e-Mining est une chaîne complète de fouille de données qui permet d'analyser des formes basées sur des interactions pour extraire des connaissances sur les comportements des utilisateurs dans les sites Web, Cadez et al. (2000). Dans ce contexte, nous définissons une session d'utilisateur comme une séquence temporelle des pages qui l'ont intéressé durant son parcours dans le site.

Par ailleurs, comprendre le comportement des utilisateurs dans les sites est devenu un souci majeur pour les propriétaires de ces sites. Cette compréhension se traduit par l'adaptation de leurs sites aux comportements et besoins des utilisateurs, Benabdeslem et al (2002). Pour se faire, l'exploitation de toute source d'informations sur les utilisateurs est nécessaire pour bien les comprendre. Cependant, pour des raisons de disponibilité et de coût, nous nous contentons dans cette étude d'une seule source d'informations enregistrées dans un fichier fourni par le serveur du site. Ce fichier appelé Fichier Log, affiche plusieurs types de caractéristiques (temps, adresse IP, Url des pages visitées, etc). Ce fichier permet de reconstituer des sessions des utilisateurs ayant visité le site associé.

Le travail présenté dans ce papier a deux objectifs : d'une part, de trouver une méthode de codage pour les données basées sur les séquences en prenant en compte la dépendance entre les composantes de ces séquences ; et d'autre part de trouver une méthode de classification incrémentale capable de traiter de nouveaux fichiers Log tout en préservant le modèle initial.

En termes d'organisation, nous présenterons dans la section suivante les données de navigation et la méthode de codage développée pour la constitution de l'espace de travail. Ensuite, la section 3 sera consacrée à la classification non supervisée par les cartes topologiques de Kohonen. Une version évolutive de cette méthode sera décrite (qui consiste à rendre évolutive la découverte de l'espace topologique). Enfin, nous présenterons dans la section 4 l'application de notre démarche de la fouille sur un site Web commercial ainsi que des comparaisons entre la version évolutive proposée et d'autres méthodes connexionnistes de classification incrémentale.

2 Données de navigation et codage

Une des importantes sources d'informations sur la navigation dans un site Web est le fichier Log. Connu aussi sous le nom du « access log » où sont enregistrées toutes les transactions entre le serveur et le navigateur.

Il existe plusieurs formats de représentation d'un fichier Log. La méthode standard par laquelle les serveurs enregistrent les accès par les navigateurs est intitulée : « *Common Log Applications* » créée par NCSA (National *Center for Supercomputing Applications*).

Le fichier Log regroupe plusieurs sessions de plusieurs utilisateurs d'un site. Une session d'utilisateur est définie comme une séquence temporelle d'accès aux pages du site par un seul utilisateur. L'identificateur de cet utilisateur est rarement fourni par le serveur. Pour cette raison, nous définissons une session utilisateur comme un accès de la même adresse IP, pourvu que le temps entre deux pages consécutives ne dépasse un seuil de visualisation.

Chaque URL dans le site est assignée par un index $j \in \{1, \ldots\ldots, N\}$

Où $\boldsymbol{N}$ représente le nombre total de pages.

Globalement, une session peut être codée de la manière binaire suivante :

$$S_j^{(i)} = \begin{cases} 1 & \text{si l'utilisateur demande la } j^{\text{ème}} \text{ URL durant la } i^{\text{ème}} \text{ session} \\ 0 & \text{sinon} \end{cases}$$

Ce codage ne traite pas les mouvements dans le site et n'explicite pas l'intérêt de l'utilisateur devant les pages. Il existe d'autres méthodes de codage Trousse (2000) qui con-

sistent à pondérer des mesures associées aux adresses hiérarchiques des pages et à leurs contenus. Cependant, ces méthodes ne sont pas adéquates à la plupart des sites conçus aujourd'hui à cause de leurs structures aléatoires dues à l'aspect dynamique (accès aux bases de données, adressage aléatoire, contenu personnalisé, etc.). D'autres auteurs, Cooley et al. (1999) proposent de coder les sessions par triplets (@IP, Identificateur, User Agent). Or l'identificateur et User Agent sont rarement communiqués dans les fichiers Log. Il existe par ailleurs, des travaux dans lesquels les sessions sont représentées comme des séquences et pour lesquelles des mesures de similarités adaptées sont définies, Zaïane et al. (1998). Néanmoins, nous ne pouvions pas utiliser ces mesures qui auraient pu causer un problème de complexité avec une matrice de dissimilarités importante vue la quantité d'informations contenues dans les fichiers log. Pour toutes ces raisons, nous avons développé une autre méthode de codage à partir du fichier Log, qui consiste à caractériser une page par son importance de passage. En d'autres termes, par son poids de précédence et de succession par rapport aux autres pages du site apparues dans le fichier Log.

Le principe de cette méthode consiste à calculer pour chaque page sa fréquence de précédence et de succession par rapport à toutes les autres pages et de regrouper ces fréquences dans un seul tableau de données de taille égale au *Nombre de pages × (Nombre de pages précédentes + Nombre de pages suivantes)*.

Nous avons remarqué que ce codage tel qu'il est décrit cause un problème d'effectif. En effet, n'ayant pas d'assez d'URLs ''significatives'' dans le site, nous ne pouvons pas avoir un tableau avec un nombre de données suffisant pour l'apprentissage.

Pour remédier à ce problème, nous avons proposé de glisser la matrice sur le mois. En d'autres termes, nous avons calculé une matrice de codage par jour, voire même par semaine ou généralement par pourcentage de partitionnement sur la base. Cette méthode nous permet de multiplier le nombre d'échantillons et d'avoir plusieurs exemples pour chaque URL.

URLs	Jour	E	URL_1.... URL_j.... URL_N	URL_1.... URL_k.... URL_N	S
URL_1					
.....					
URL_i					
.....					
URL_N					
URL_1					
.....					
URL_i	*k*	*a….*	*b….*	*c….*	*d*
.....					
URL_k					
….					
URL_1					
.....					
URL_i					
.....					
URL_N					

TAB. 1 – *Codage des URLs par rapport aux sessions dans le fichier Log.*

Dans le tableau ci-dessus, E représente une variable caractérisant le nombre de fois qu'une URL donnée apparaît en entrée (respectivement, S en sortie).

A titre d'exemple, *Dans le* $k^{iéme}$ *jour du fichier l'*URL_i *est apparue a* fois comme page d'entrée, précédée *b* fois par la page URL_j, suivie *c* fois par la page URL_k et apparue *d* fois comme page de sortie.

Par ce codage, Non seulement, on caractérise les pages par des variables comportementales i.e. tel que les internautes perçoivent le site. Mais on multiplie aussi les informations sur les pages en codant leurs intérêts sur les pages jour par jour.

3 Classification incrémentale et visualisation

Les techniques numériques de reconnaissance de formes sollicitent une méthode de classification améliorée pour comprendre, interpréter et simplifier les grandes quantités de données multidimensionnelles. La plupart du temps, K-Means et d'autres méthodes de partionement sont utilisées dans les applications industrielles Ribert et al. (1999). Cependant, elles présentent l'inconvénient majeur de déterminer une solution finale indépendante des conditions initiales, spécialement concernant le nombre des classes. Cette connaissance préalable est rarement disponible pour les utilisateurs. En effet, s'ils utilisent une technique de classification c'est parce qu'ils ignorent la structure de leurs données. Par conséquent, cette contrainte est généralement intraitable. Par ailleurs, les techniques de classification considèrent que les bases de données sont complètement représentatives aux problèmes. Or c'est rarement le cas quand il s'agit de traiter des problèmes complexes. En d'autres termes, d'un point de vue pratique, un problème complexe est souvent incrémental. Par conséquent, il devient très important de concevoir des systèmes de classification incrémentale capables de prendre en compte les formes qui ne sont pas disponibles au moment de la constitution du modèle initial par les données initiales. Introduire l'aspect d'incrémentalité n'est pas nouveau. Il existe plusieurs méthodes qui mettent à jour dynamiquement des modèles de classification. Nous nous limitons ici aux approches connexionnistes de type SOM. Cependant, ces méthodes ont quelques limites. Par exemple dans IGG (Incremental grid growing) Blackmore (1995), bien que le processus soit incrémental, il utilise toujours la même base à chaque adaptation de la grille courante. L'aspect incrémental concerne donc, la topologie sans s'occuper de l'arrivée dynamique des données. La méthode GNG (Growing Neural Gas) dans Fritz (1994) construit quant à elle, le modèle en tenant compte des nouvelles formes. Cependant cette méthode souffre d'un problème de lissage. Cela veut dire que, après sa création, la carte est représentée par des sous-ensembles de neurones indépendants. En effet, entre chaque paire de sous-ensembles, des neurones ne possédant pas de connexions sont définitivement supprimés. Cela dit, ces neurones n'auront jamais la possibilité d'être activés par un éventuel ensemble de données qui risquent d'arriver ultérieurement.

3.1 Un bref rappel sur SOM

L'algorithme SOM de Kohonen, Kohonen (1995) représente un véritable outil de visualisation des données multidimensionnelles. Il permet de convertir les relations statistiques complexes et non linéaires en relations géométriques simples dans une carte bidimensionnelle.

De plus cet algorithme permet de compresser l'information tout en gardant les relations topologiques et métriques les plus pertinentes dans l'espace de données réel.

L'apprentissage dans les cartes topologiques se fait avec une fonction de voisinage. Il procède en 3 étapes :

- ***Initialisation*** : il s'agit de l'initialisation des poids.
A l'instant t =0, initialiser les poids W_j,à des valeurs triées aléatoirement

- ***Compétition*** : à chaque entrée d'un exemple au réseau, un calcul de distance est effectué pour activer un neurone dit gagnant, c'est celui dont le potentiel d'activation est le plus fort en fonction de l'entrée.
A l'instant $t > 0$, présenter la forme z à la carte et choisir le neurone gagnant W_c^t :

$$\left\|z - W_c^t\right\|^2 = \min_{j \in C} \left\|z - w_j^t\right\|^2$$

- ***Adaptation*** : le choix d'un nœud particulier permet alors d'ajuster les poids localement, en minimisant la différence qui existe encore entre les poids et le vecteur d'entrée. Cet ajustement se fait suivant une forme de voisinage qui peut être carrée, ronde ou hexagonale.

$$W_j^t = W_j^{t-1} + \varepsilon(t)K(\delta(c,j))(W_j^{t-1} - z)$$

Où :

- $\varepsilon(t)$ est le pas d'apprentissage. Il est défini comme fonction décroissante du temps t.
- $\delta(i, j)$ est la distance sur la carte
- $K(.)$ représente la fonction de voisinage

$t = t+1$, présenter une autre forme z et répéter les deux étapes de compétition et d'adaptation.

Le processus est donc constitué de ces trois phases qui sont itérées jusqu'à la minimisation d'une erreur globale calculée sur l'ensemble du réseau ou sur un nombre de cycles d'apprentissage fixé empiriquement.

3.2 Une version incrémentale de SOM : e-SOM

Dans cette section, nous proposons une nouvelle version des cartes topologiques. Cette version possède un aspect incrémental « complet ». En effet, il s'agit de créer la topologie en fonction de l'arrivée des données. Nous appelons cette version e-SOM.

Notre problématique est due à l'arrivée de plusieurs bases de données après la création du modèle initial. Au lieu de refaire tout le modèle, nous proposons de garder l'existant est de l'incrémenter (le mettre à jour), en fonction des nouvelles vagues de données qui arrivent.

3.2.1 Création et gestion de nouveaux neurones

Nous proposons tout d'abord d'étudier la distribution des nouvelles données sur la carte initiale. Pour cela nous appliquons la procédure de *compétition* de SOM (i.e. le classement des données dans la carte). Nous trouvons, par exemple, 10% de données sur le premier neurone, 24% sur le i^{émé} neurones…etc

De manière statistique, nous marquons les neurones qui appartiennent au périmètre de la grille et qui possèdent une densité (effectif) supérieure ou égal à *N/C*. Tels que *C* représente le nombre des neurones de la carte et *N* le nombre d'exemples dans la nouvelle base en cours.

Intuitivement, nous nous intéressons qu'aux neurones du périmètre, car nous estimons que les neurones situés à l'intérieur de la carte, sont déjà enfermés par leur voisinage, alors que ceux des frontières possèdent une possibilité de voisinage en dehors de la carte. Par exemple, le neurone en haut à gauche, possède deux possibilités de voisinage (un au dessous et un autre à gauche)

De plus, l'ajout des neurones dépend du voisinage du neurone marqué (un neurone est dit marqué, s'il est actif et autorisé à un voisinage).

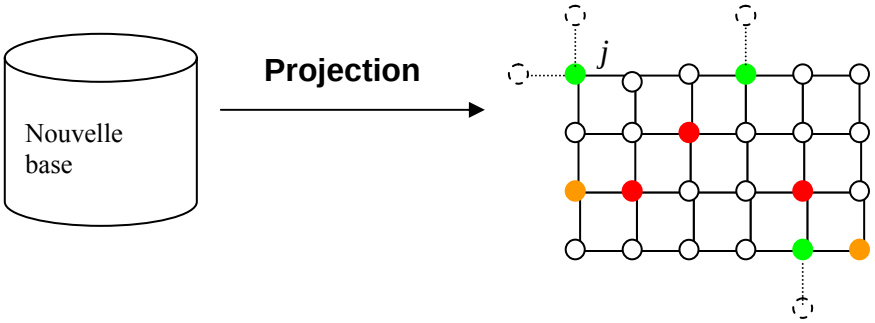

FIG. 1. *Gestion des nouveaux neurones crées pour l'apprentissage incrémental*

- Neurone actif autorisé à avoir de nouveaux voisins
- Neurone actif non marqué, car sa densité n'est pas importante
- Neurone actif mais non autorisé à un nouveau voisinage à cause de sa position dans la carte
- Nouveau neurone « provisoirement » créé.

Les nouveaux neurones créés (*j'*) sont ainsi initialisés par la formule suivante :

$$W_{j'}(k) = \frac{1}{2}\left[\frac{1}{N_j}\sum_{i=1}^{N_j} V_{i,j}(k) + W_j(k)\right] + \varepsilon$$

k : est l'index du vecteur poids *W*, $1 \le k \le p$, *p* étant le nombre de variables d'entrée
$W_{j'}(k)$: est le $k^{ème}$ élément du vecteur poids du nouveau neurone créé au voisinage du neurone *j(j est le neurone marqué)*
Nj = Card(*j*) est l'effectif de la nouvelle base qui active le neurone *j*
$V_{i,j}(k)$: est le $k^{ème}$ élément du $i^{ème}$ vecteur de la nouvelle base activant le neurone *j*
$W_j(k)$: est le $k^{ème}$ élément du vecteur poids du neurone *j* après l'apprentissage de la carte courante.
ε : est un nombre aléatoire tel que : $0 \le \varepsilon \le 1$.

La formule proposée représente la moyenne entre le vecteur poids du neurone marqué *j* (qui est responsable de la création du nouveau neurone) et la combinaison de tous les vecteurs de la nouvelles base qui sont projetés sur le neurone *j*.

3.2.2 Mise à jour de la topologie

Une fois le nouveau voisinage organisé, il ne reste qu'à entraîner la carte avec la nouvelle base. Nous répétons donc, le processus SOM sur la nouvelle base pour l'adaptation générale sur les anciens et les nouveaux neurones. Cela fait référence à un apprentissage par morceau (de neurones de la carte) en fonction des données disponibles.

Le processus incrémental est itératif à chaque arrivée d'une nouvelle base de données, ce qui explique la dynamique de la restructuration de la carte.

3.2.3 Optimisation

Une fois l'apprentissage incrémental fini à chaque arrivée d'une base, on réutilise la procédure de compétition de SOM pour déterminer l'activation des neurones marqués selon le principe de la section 3.2.1. (FIG. 1)

Si après cette compétition, le neurone créé est inactif, il est directement supprimé sinon il est gardé , et il devient donc « réel ».

Nous présentons la formulation algorithmique qui interprète cette nouvelle version :

1- Application de SOM (Création de la première carte: T_0) sur la première base de données (D_0)

-Initialisation des poids
-Compétition des formes par rapport au poids
-Adaptation des poids

2-Mise à jour de la topologie

- i=1
// *Pour une nouvelle base D_i de taille N_i*
-$\forall\ z \in D_i$, j = projection (z,T_{i-1})
-Si $j \in$ périmètre (Ti-1) Alors
-Si $N_j \geq \frac{N_i}{C_i}$ Alors //*C_i est le nombre de neurone de la carte courante*
Création (j', voisinage(j))
Finsi
Finsi

3-Initialisation des nouveaux neurones (j')

$$W_{j'}(k) = \frac{1}{2}\left[\frac{1}{N_j}\sum_{i=1}^{N_j} V_{i,\,j}(k) + W_j(k)\right] + \varepsilon$$

4-Adaptation de T_i , $\forall j$

Adaptation selon SOM de C_i , $\forall j$
// $T_i = T_{i-1} \cup \{j'\}$//

5-Remise à jour
Éliminer tout *j'* , tel que Card (*j'*) = 0 // tout nouveau neurone créé vide
i = *i*+1, si *i* < *M* Alors Retour à 2 sinon Arrêt
M : représente le nombre de sous – bases.

3.3 Post-classification de e-SOM

Comme l'algorithme classique de SOM, e-SOM fournit une topologie de neurones représentés par les vecteurs de poids (souvent appelés référents). Chaque vecteur est comparé par rapport à tous les exemples de la base pour un objectif de groupement (Clustering). Cependant, on peut trouver quelques neurones inactifs à l'intérieur de la carte à cause du nombre fixé a priori dû à la topologie initiale faite par SOM (première étape de e-SOM). Ceci étant, pour optimiser ce nombre, nous proposons une classification des référents par la méthode de la classification ascendante hiérarchique (CAH) Bouroche et al. (1994) (FIG. 2). Nous utilisons le critère de Ward pour l'agrégation des classes. Ce critère consiste à maximiser l'inertie inter-classes et minimiser l'inertie intra-classes. Dans ce cas, si deux exemples X_1, X_2 activent respectivement deux neurones N_1, N_2 et N_1, N_2 appartiennent à la même classe formée par CAH, nous considérons que X_1 et X_2 sont de la même classe dans la cartographie.

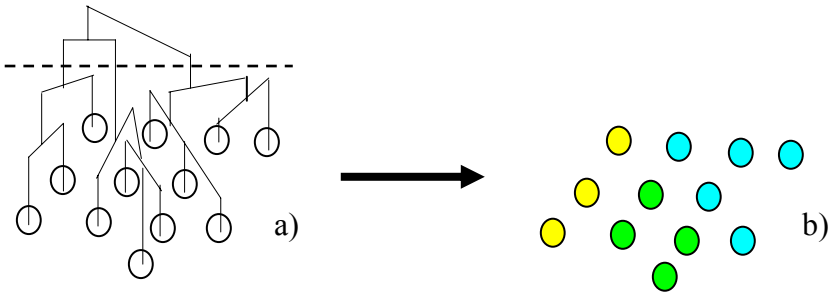

FIG. 2. *Post – classification de e-SOM par CAH. a) Carte faite par e-SOM (12 neurones) .b) Carte optimisée par regroupement de ces neurones (3 classes).*

4 Résultats

4.1 Les données

Pour notre application, nous utilisons un site commercial nommé : www.123credit.com. Il s'agit d'un site qui commercialise un certain nombre de services pour des clients désirant avoir des crédits (Auto, Maison, terrain, etc.). Le fichier Log a été fourni dans son état brut

en 4 temps. Ce fichier dans son intégralité, décrit des transactions de navigation pendant 1 mois. Son volume avoisine les 700 Méga octets.

Après prétraitement et codage selon le principe décrit dans la section 2, nous extrayons un nombre de sessions égal à 37174 dont le nombre d'Urls est égal à 40. Plusieurs Urls ont été supprimées à cause du nombre réduit de fréquentations (<1%). De plus, le site analysé était nouveau à l'époque de l'étude. Il contenait donc un nombre réduit d'Urls. Une session est une succession d'enregistrements ayant la même adresse IP et ne dépassant pas un seuil de visualisation paramétrable (30 secondes pour notre expérience), Benabdeslem (2003)

Le fichier ayant des interactions de 1 mois (soit 30 jours), le tableau de données représente (40 × 30) exemples, caractérisés par ((40×2)+2) Urls. Soit donc, un tableau de 1200 individus sur 82 variables.

Il est bien évident que ce dernier tableau contient des entiers. Nous normalisons le tableau en colonnes pour centrer et réduire les données. Nous proposons aussi, de le normaliser en lignes pour éliminer toute dépendance entre les variables.

4.2 Classification et visualisation

Le fichier étant fourni en 4 morceaux, nous avons commencé avec un premier codage d'une première base qui était initialement disponible et nous l'avons incrémenté au fur et à mesure de l'arrivée des 3 autres (en moyenne 300 exemples obtenus après l'arrivée de chaque nouvelle base). Ces bases sont ainsi présentées de manière itérative à e-SOM pour former des classes de pages selon les interactions des utilisateurs dans le site. Ensuite, l'algorithme CAH est appliqué sur les référents pour optimiser le nombre de neurones de la carte. Nous obtenons donc des classes de neurones. Cependant, après la projection des exemples dans la carte obtenue, chaque Url peut activer plus qu'un seul neurone (selon sa parution pendant les jours du mois). Ceci étant, une Url n'est pas unique, mais peut être représentée par un ensemble d'observations où chaque observation représente l'intérêt de la dite Url sur les utilisateurs dans un jour donné. Nous avons constaté que les neurones activés par la même Url sont généralement voisins dans la carte. Pratiquement, pour étiqueter la carte nous procédons par la technique du vote majoritaire. Cette technique consiste à attribuer des libellées pour chaque zone (ensemble de neurones proches dans la carte). L'Url « étiquette » qui caractérise la carte sera celle qui sera omniprésente dans la zone de la même manière que l'étiquetage d'un neurone (FIG 3).

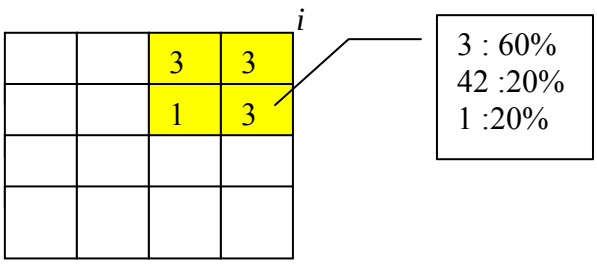

FIG 3. Principe d'étiquetage de la carte par vote majoritaire

Dans la FIG3, Le neurone *i* est étiqueté par l'URL 3 car elle représente 60% de présence par rapport aux autres URLs (42 et 1) et la zone colorée est étiqueteé par cette même URL car elle est présente dans la plus part des neurones. Nous pouvons remarquer que L'Url 3 a activé plusieurs neurones. Cela est dû à sa présence dans la base sous forme de plusieurs exemples. Cette étiquetage est ensuite validé par des experts en modélisation de clients et Marketing relationnel.

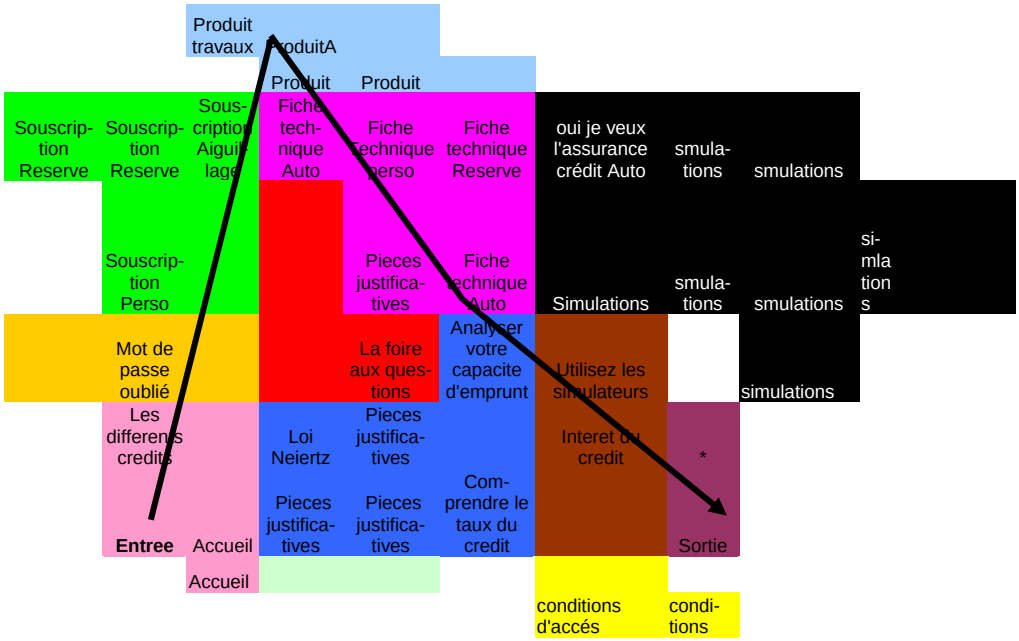

FIG. 4. *Visualisation du site www.123credit.com tel que les internautes le perçoivent*

FIG.4 montre l'application de e-SOM sur les données de navigation enregistrées dans le fichier Log du site. Chaque couleur est spécifique à une classe de neurone regroupant les pages susceptibles d'intéresser un profil donné d'utilisateurs. Nous pouvons aussi remarquer que la carte n'est pas uniforme (rectangulaire ou hexagonale comme dans SOM) à cause des nouveaux neurones qui sont créés dynamiquement selon la disponibilité des bases.

Nous avons aussi comparé e-SOM avec l'algorithme classique de SOM ainsi que d'autres versions incrémentales de SOM comme IGG et GNG (TAB. 2). Cette comparaison est faite sur quatre critères :

Méthode	N	Lissage	TE	GDR
SOM	960	Oui	0.3	0.20
IGG	850	Oui	0.2	0.23
GNG	230	Non	0.15	0.46
e-SOM	240	Oui	0.10	0.38

TAB. 2 – *Comparaison entre différentes versions incrémentale de SOM*

- N : représente le nombre de neurones dans la carte. Pour SOM, ce nombre est calculé selon l'heuristique de Kohonen (N= 5 × (Nombre d'exemples)$^{0.54321}$). Pour les versions incrémentales, ce nombre est calculé dynamiquement chacune selon son algorithme.
- TE : représente l'erreur topologique pour représenter la qualité de la carte. Cette mesure représente le pourcentage des données pour lesquelles le premier neurone activé et le deuxième ne sont pas adjacents dans la carte.
- GDR=$\frac{E_i - E_f}{E_i}$: la décroissance de l'erreur quadratique moyenne sur la carte. E_i et E_f représentent respectivement l'erreur initiale et l'erreur finale de l'apprentissage.

De plus, FIG3 fournit un outil de visualisation des différentes interactions des utilisateurs du site. Ces interactions sont représentées par la projection des sessions du fichier Log vers la carte obtenue. Par exemple, le chemin illustré dans la figure représente 40% des utilisateurs qui (1) entrent dans le site par la page d'accueil (Cela montre que le site est assez bien connu) (2) visitent les produits de la catégorie A (crédits pour voitures) (3) demandent plus d'informations sur ces produits (4) sortent du site sans l'achat d'aucun service. Il faut savoir que ce chemin ne représente pas forcément des utilisateurs qui ont produit exactement la même session, mais qui ont eu dans leurs parcours des Urls similaires (activant les mêmes neurones dans la carte). Par ce type de cheminement, nous pouvons constater que les différentes informations ou les hyperliens dans les pages associés à ces produits doivent être mis en question et donc modifiés car elles ne sont pas attractives pour les utilisateurs. Ce genre d'analyse de profils est ainsi réalisé facilement sur l'ensemble de tous les utilisateurs du site par une simple projection de leurs interactions sur la cartographie obtenue par e-SOM.

Nous pouvons remarquer dans TAB. 2 que e-SOM donne généralement de meilleurs résultats que ses concurrentes. En effet, le nombre final de neurones obtenus est optimal, l'erreur topologique et l'erreur quadratique sont considérablement réduites et la propriété du lissage est respectée grâce à la topologie initiale faite par SOM sur la base initiale. Cette propriété est importante, notamment pour de nouvelles données susceptibles d'activer les neurones à l'intérieur de la carte. Cependant, elle est absente dans la méthode GNG qui est relativement l'autre meilleure approche particulièrement sur N et GDR.

5 Conclusion

Ce travail nous a permis d'analyser les comportements des internautes face à des sites Web. Nous avons tout d'abord, proposé une nouvelle méthode de codage basée les interactions enregistrées dans le fichier Log. Ensuite, Nous avons développé une idée qui consiste à traiter les nouvelles données qui arrivent, en utilisant le même modèle issu des données antérieures. En d'autres termes, nous avons rendu dynamique la création du modèle de la classification (la carte). Il s'agit donc, d'une carte qui change de figure dans le temps.

L'algorithme que nous avons proposé, respecte les mêmes règles de la création des cartes topologiques SOM (initialisation, compétition et adaptation) et permet de rendre la création des neurones dynamique et dépendante de la nature des informations qui arrivent dans le temps.

Nous avons donc, rendu évolutive, la découverte de l'espace topologique : i.e. le nombre de regroupement homogènes (clusters) change dans le temps en fonction des informations complémentaires qui arrivent. Une procédure tout à fait intéressante pour une mise à jour intelligente de la cartographie.

Références

Benabdeslem, K. (2003). *Approches connexionnistes pour la visualisation et la classification des séquences évolutives : Application aux données issues d'usage d'Internet*. Thèse de doctorat de l'université Paris 13.

Benabdeslem, K., Bennani, Y. and Janvier, E. (2001). *Connectionist approach for Website visitors behaviors mining*. In Proceedings of ACS/IEEE International Conference on Computer Systems and Applications, Lebanon.

Benabdeslem, K., Bennani, Y and Janvier, E. (2002). *Visualization and analysis of web navigation data* in LNCS2415 (Springer), pp 486-491, Madrid.

Bouroche, J, M. and Saporta, G, (1994). *L'analyse des données*, Presse universitaire de France.

Bennani, Y. (1994). *Multi-expert and hybrid connectionist approach for pattern recognition: speaker identification task*. International Journal of Neural Systems, Vol.5, No. 3, 207-216.

Blackmore, J. (1995). *Visualizing high dimensional structure with the incremental grid growing neural network*. technical report, departement of computer sciences of the university of Texas, Austin.

Cadez, I., Heckerman, D., Meek, C., Smyth, P., White, S. (2000). *Visualization of Navigation Patterns on a Web Site Using Model Based Clustering*. it In proceedings of the KDD.

Cooley, R., Mobasher, B. and Srivastaval, J. (1999). *Data preparation for mining world mining wide web browsing patterns*. Knowl. Inf. Syst. Vol 1, No 1, 5-32.

Deshpande, M, Karypis, G. (2001). *Selective Markov Models for predicting Web-page accesses*. 1st SIAM conference on Data Mining, Chicago, Illinois.

Forgy, E. (1965). *Cluster analysis of multivariate data : efficiency versus interpretability of classifications*, Biometrics, Vol. 21, 768.

Fraley, C and Raftery,A. (1998); *How many clusters ? Which clustering method ? Answers via model-based cluster analysis*. Computer Journal, 41, 578-588.

Fritz, B. (1994). *Growing cell structures - A self organizing network for unsupervised learning*. Neural Networks, Vol 7, N9, pp 1441,1460.

Hattori, K and Torii, Y. (1993). *Effective algorithms for the nearest neighbour method in the clustering problem*, Pattern Recognition, Vol. 26, N°5, pp. 741-746.

Kohonen, T. (1995). *Self-Organizing Maps*. Series in Information Sciences, Vol. 30. Springer, Heidelberg.

Martinez, T and Schulten, K. (1991). *A neural gas network learns topologies*. Artificial Neural Networks, Elsevier Science Publishers B.V, pp 397,402.

McLachlan, G and Basford, K. (1988). *Mixture Models : Inference and Applications to Clustering*. Marcel Dekker.

Perkowitz, M and Etzioni, O. (2000). *Towards adaptative web sites : conceptual framework and case study*. Artificial Intelligence Journal, 118 ,1-2.

Ribert, A., Ennaji, A. and Lecourtier, Y. (1999). *An Incremental Hierarchical Clustering*. Vision Interface'99, Trois rivières, Canada.

Trousse, B. (2000). *Evaluation of the Prediction Capability of a User behaviour Mining Approach for Adaptative Web Sites*. In RIAO 2000, 6th Conference on "Content-Based Multimedia Information Access", College de France, Paris, France, April pp12-14,.

Zaïane, O., Xin, M., and Han, J. (1998). *Discovering Web access patterns and trends by applying OLAP and data mining technology on Web Logs*. In proceeding of the Advances in digital Libraries Conference, 19-29.

Zeboulon, A. (2001). *Reconnaissance et classification de séquences*. DEA127's internship report, University of Paris 9.

Zeboulon, Z, Bennani, Y and Benabdeslem, K. (2003). *Hybrid connectionist approach for knowledge discovery from web navigation pattern*. In the proceedings of ACS/IEEE International Conference on Computer Systems and Applications, Tunisia.

Summary

In this paper, we present a system of behavioural data mining from navigations of commercial Web sites customers. We show how to develop an unsupervised learning approach for this type of data which stored in the form of traces of navigation in Log files. The first part of this study relates to data coding problem that used later on in the analysis. In fact, the Web sites are dynamics, where the pages cannot be characterized by fixed variables; the hierarchy of URL addresses, the contents, etc. Hence, they represented by numerical identifiers, without having any “sense” which are used as indices addresses to have information in data bases to fill the contents of the pages. Therefore, we propose a new method that coding the ses-

sions from the Log file. This technique consists in characterizing a given page by a weight vector of passage importance i.e by its weights of precedence and successions relating to all the other pages which appear in the Log file. In the second part, we analyze the properties of Kohonen's algorithm where we propose a version adapted to the behavioural data. This stage enables us (1) to build a map of the Web site such as it is seen by the customers (2) to gather the pages for an objective of coding of sessions (3) and to project the interactions of the customers of the Log file on the map in the form of trajectories symbolizing their behaviours.