Traitement de Flux par un Graphe de Voisinage Incrémental

Ibrahim Louhi*,** Lydia Boudjeloud-Assala*
Thomas Tamisier**

*Université de Lorraine,
Laboratoire d'Informatique Théorique et Appliquée, LITA-EA 3097,
Metz, F-57045, France
{ibrahim.louhi, lydia.boudjeloud-assala}@univ-lorraine.fr
**Luxembourg Institute of Science and Technology,
41, rue du Brill, L-4422 Belvaux, Luxembourg
{ibrahim.louhi, thomas.tamisier}@list.lu

Résumé. Cet article s'intéresse au traitement et de la visualisation des flux de données en temps réel. Pour traiter les flux, nous proposons une nouvelle approche utilisant un clustering basé sur le voisinage. Au lieu de traiter les nouveaux éléments un par un, nous choisissons de traiter chaque groupe de nouveaux éléments simultanément. Un clustering est appliqué sur les éléments de chaque nouveau groupe en utilisant des graphes de voisinage. Les clusters obtenus sont ensuite utilisés dans la construction incrémentale d'un graphe représentant le flux de données. Le graphe du flux est visualisé en temps réel à l'aide de visualisations spécifiques reflétant le processus du traitement. En vue de valider l'approche, nous l'appliquons sur plusieurs jeux de données et la comparons avec divers algorithmes de clustering de flux.

1 Introduction

Ces dernières années ont connu de très grandes avancées technologiques, ce qui a contribué à l'augmentation du volume des données disponibles. Afin d'exploiter ces données brutes il est nécessaire d'en d'extraire des connaissances facilement compréhensibles. La fouille de données est l'étape la plus importante dans le processus d'extraction de connaissances. Plusieurs méthodes de fouille de données ont été proposées depuis que les chercheurs se sont intéressés à ce domaine. Le clustering est une tâche de fouille de données dont le principe est de diviser les données en sous-ensembles appelés *clusters*. Cette méthode considère que les éléments qui se ressemblent (selon certains critères) doivent être regroupés dans le même cluster (Berkhin, 2006). L'ensemble des clusters est une description de l'ensemble des données. Le clustering est généralement utilisé quand aucune information sur les classes n'est disponible et donc qu'aucune technique d'apprentissage ne peut être utilisée.

Dans plusieurs domaines les données sont produites d'une façon continue et souvent à une très grande vitesse. A titre d'exemple, Le suivi de la consommation énergétique, des opérations financières la géolocalisation des smartphones ou les capteurs météorologiques produisent une

série de données à une fréquence stable ou variable. Ce type de données est connu sous le nom de flux de données.

Les flux de données sont caractérisés principalement par l'aspect temporel et par leur grande taille, ces derniers rendent l'extraction de connaissances malaisée. Cependant, le traitement des flux de données est primordial dans de nombreux domaines comme la sécurité des réseaux informatiques, la détection de fraudes, l'analyse des articles de presse, ou l'analyse de réseaux sociaux. Il est donc nécessaire d'adapter les méthodes classiques de fouille de données aux flux de données.

Les méthodes classiques de clustering supposent que les données ont un volume connu permettant de les stocker sur une machine et de les traiter avec la possibilité d'effectuer plusieurs parcours. Le traitement des flux de données doit prendre en considération certaines caractéristiques :

— Les données sont produites d'une façon continue à une fréquence stable ou variable, et généralement à une grande vitesse.
— Nous ne contrôlons pas l'ordre d'arrivée des éléments.
— Le comportement des données peut changer au cours du temps.

Un processus de clustering de flux de données doit être capable de traiter continuellement les données sans aucune restriction sur la mémoire ni sur le temps (Gama, 2010). Afin de respecter ces conditions, le processus du clustering doit idéalement (Silva et al., 2013) :

— être rapide et incrémental,
— s'adapter rapidement aux changements du flux,
— s'adapter à la fréquence d'arrivée des éléments,
— générer des résultats d'une taille qui n'augmente pas très rapidement,
— détecter rapidement les outliers et les prendre en considération dans le traitement,
— pouvoir traiter plusieurs types de données.

Dans ce papier nous présentons dans l'état de l'art quelques techniques de clustering de flux de données. Ensuite, nous présentons notre propre approche NG-Stream pour le clustering de flux basée sur les graphes de voisinage. Nous testons NG-Stream sur plusieurs jeux de données que nous évaluons avec des critères externes et internes. Nous comparons également nos résultats avec ceux de quelques algorithmes de la littérature.

2 État de l'art

Le but des techniques de clustering est d'obtenir à partir d'un ensemble de données des partitions homogènes selon un critère (la similarité par exemple). Les méthodes classiques de clustering, qui traitent des données de taille connue, utilisent généralement les techniques de *k-means* (Forgy, 1965) ou *k-medians* (MacQueen et al., 1967). Elles sont itératives et nécessitent plusieurs itérations sur les données pour sélectionner des représentants de clusters. Un représentant est un élément à partir duquel un cluster est construit.

Plusieurs approches ont essayé d'adapter les méthodes classiques du clustering basées sur les représentants de partitions aux flux de données. L'algorithme STREAM (Guha et al., 2000) est basé sur la technique des *k-medians*. L'idée consiste à découper le flux en fenêtres, ensuite l'algorithme de clustering *k-medians* est appliqué sur chacune d'elles. Les k médians sont choisis, et chaque élément est affecté à son médian le plus proche (le représentant). Le but est de choisir les représentants en minimisant la *Sum of Squared Error (SSQ)* entre les éléments et les

médians. Après le traitement de la première fenêtre, un ensemble de *k* représentants auxquels des poids sont associés est obtenu. Un poids est le nombre d'éléments affectés au représentant. La fenêtre suivante est traitée séparément avec le même processus, ce qui augmente le nombre de représentants à *2*k*. Ainsi de suite, quand le nombre des représentants atteint un seuil prédéfini *m*, *k-medians* est appliqué sur l'ensemble des représentants. Les médians obtenus sont des représentants de niveau 2 (des représentants de représentants). D'une manière recursive, à chaque fois que le seuil *m* est atteint sur un niveau *n*, des représentants sont désignés au niveau *n+1*. A la fin du flux, *k-medians* est appliqué une dernière fois sur tous les médians du dernier niveau. La limite de l'algorithme STREAM réside dans son insensibilité face à l'évolution du flux dans le temps. Les clusters ne prennent pas en considération l'aspect temporel.

L'algorithme CLUSTREAM (Aggarwal et al., 2003) apporte une solution à cette problématique. Afin de permettre de garder une trace des clusters obtenus antérieurement dans le flux, il utilise deux types de clustering. Un micro-clustering online (en temps réel) et un macro-clustering offline (en temps différé). Le micro-clustering stocke un résumé statistique sur le flux de données (les clusters et leur position temporelle dans le flux), le macro-clustering utilise ce résumé de données pour fournir le résultat obtenu par le clustering à n'importe quel point temporel du flux. Quand de nouveaux éléments arrivent, les micro-clusters sont mis-à-jour afin de refléter les changements : la distance entre chaque nouvel élément et les centres des micro-clusters est calculée, ensuite le nouvel élément peut soit intégrer un micro-cluster, soit créer un nouveau cluster. CLUSTREE (Kranen et al., 2009) est un algorithme basé sur le micro-clustering qui s'adapte à la vitesse du flux. Le processus d'affectation des nouveaux éléments aux micro-clusters change suivant la vitesse du flux. Pour les flux rapides, il affecte les nouveaux éléments par groupes au micro-clusters au lieu élément par élément. CLUSTREE peut maintenir le même nombre de micro-clusters à n'importe quelle vitesse du flux. Ce type d'algorithmes (CLUSTREAM et CLUSTREE) dépend fortement de la fréquence avec laquelle ils stockent le résumé statistique. Les anciennes données sont stockées avec un minimum de détails, ce qui peut impliquer une imprécision sur l'état des anciens clusters.

Les approches basées sur la densité essayent de construire un profil de densité de l'ensemble de données afin d'obtenir des régions denses. Ces dernières peuvent avoir des formes et des tailles différentes, contrairement au calcul de la distance euclidienne entre les éléments, par exemple, où les clusters ont une forme sphérique. Cependant un seuil de densité est fixé pour déterminer les régions denses. Vu que la construction d'un profil de densité nécessite de multiples itérations sur les données, le challenge dans le cas des flux de données est d'estimer la densité avec un seul parcours (ou quelques parcours au plus). L'algorithme DENSTREAM (Cao et al., 2006) combine le micro-clustering avec l'estimation de densité. La première étape consiste à définir un élément de départ. Ce dernier est un élément dont le poids de son voisinage (le nombre de ses voisins) est supérieur à un seuil. Une région dense est constituée de l'ensemble des voisins de cet élément. Contrairement au micro-clustering standard, DENSTREAM ne fixe pas le nombre des micro-clusters. Cependant, dans le cas où les éléments de l'ensemble de données sont très éloignés entre eux, la construction d'un profil de densité n'est pas une tâche facile. De ce fait, les méthodes basées sur la densité peuvent rencontrer des difficultés à construire des clusters.

Les méthodes de clustering basées sur les grilles sont une sous-classe des approches basées sur la densité. Une structure sous la forme d'une grille est utilisée pour estimer la densité autour de chaque élément. Les données sont représentées par leurs dimensions (variables),

chaque dimension est divisée en plusieurs partitions. L'espace de données est donc divisé en des cellules $dimension$ x $partition$ où la densité d'une cellule est définie par le nombre de ses points. Une région dense est constituée de l'union des cellules denses connectées. L'algorithme DSTREAM (Chen et Tu, 2007) cherche à déterminer la densité des données, mais au lieu d'utiliser des micro-clusters il utilise une grille. La différence entre la densité basée sur le micro-clustering et la densité basée sur les grilles est que dans cette dernière, la valeur de la densité est mise à jour à chaque fois qu'un nouvel élément arrive, ce qui fait qu'une cellule peut changer de statut à chaque instant (dense ou à faible densité). DSTREAM suppose qu'il n'est pas nécessaire de garder les cellules vides. Il utilise des méthodes pour identifier et supprimer de telles cellules. La faiblesse de cet algorithme réside dans la suppression des cellules vides adjacentes aux régions denses, ce qui peut détériorer la qualité et les résultats du clustering.

Dans la littérature, il existe plusieurs algorithmes de clustering basés sur le voisinage qui utilisent une représentation sous forme de graphes. Un graphe de voisinage est une structure géométrique qui se base sur la distance pour déterminer le voisinage de chaque élément (Toussaint, 1991). La notion de voisinage peut être reflétée sous plusieurs formes dans un graphe : les notions classiques de voisinage comme le Nearest Neighbor Graph qui relie chaque élément avec son plus proche voisin, le ϵ-Neighbor graph où chaque couple de noeuds dont la distance est inférieure à ϵ sont reliés, et le Complet Neighbor Graph où tous ses noeuds sont reliés deux à deux avec des arêtes pondérées représentant la distance entre chaque couple d'éléments. Ainsi que d'autres notions de voisinage telles que le Minimal Spanning Tree (Yao, 1982) qui regroupe les noeuds d'un graphe pondéré de telle sorte que la somme des poids des arêtes au sein du même groupe soit minimale. Le Relative Neighborhood Graph (Toussaint, 1980) qui relie deux noeuds si et seulement si chacun d'eux est le noeud le plus proche de l'autre. Le Gabriel Graph (Gabriel et Sokal, 1969) qui relie deux noeuds si et seulement si le cercle défini par l'arête entre ces deux noeuds ne contient aucun autre noeud à l'intérieur.

Parmi les algorithmes de clustering basés sur les graphes de voisinage nous pouvons citer ROCK (Guha et al., 1999) qui traite les données catégorielles, ROCK considère deux éléments comme étant voisins si leur distance est supérieure à un seuil. Ensuite, pour chaque couple de voisins, il calcule *link* qui représente le nombre de leurs voisins en commun. Un cluster est constitué d'un ensemble de voisins avec un grand degré de connectivité (une moyenne de *link* assez grande). CHAMELEON (Karypis et al., 1999) est basé sur le *k-Nearest Neighbor Graph*. CHAMELEON utilise un algorithme de partitionnent de graphes (coupe minimum) pour trouver des sous-clusters, ensuite pour constituer les clusters finaux de l'ensemble de données, il fusionne les sous-clusters en se basant sur l'interconnectivité relative (obtenue par la normalisation des poids des arêtes connectant deux clusters). Le Spectral Clustering (Ng et al., 2002) se base aussi sur le principe de coupe minimum. Afin de résoudre le problème d'optimisation de la coupe minimum, le Spectral Clustering utilise le spectre (les vecteurs propres) d'une matrice Laplacienne construite à partir de la matrice de similarité. Chaque ligne de la matrice des vecteurs propres représente un élément, un clustering est ensuite appliqué sur les lignes de la matrice. AntGraph (Lavergne et al., 2007) s'inspire du comportement des fourmis pour proposer une construction incrémentale d'un graphe de voisinage, chaque nouvel élément est introduit dans le graphe et il suit son propre chemin (basé sur la similarité) afin de trouver l'élément avec lequel il sera relié.

Plusieurs travaux ont essayé d'utiliser des algorithmes de clustering basés sur les graphes de voisinage dans le contexte des flux de données. REPSTREAM (Lühr et Lazarescu, 2009)

est un clustering hiérarchique basé sur les graphes pour le traitement des flux de données. Pour identifier les clusters, REPSTREAM utilise des graphes construits en reliant chaque élément à ses plus proches voisins. Le premier graphe identifie les représentants des éléments parmi les nouvelles données, ces représentants sont utilisés pour construire le deuxième graphe. REPSTREAM utilise ce deuxième graphe pour prendre des décisions sur la fusion ou la séparation des éléments. Aggarwal (Aggarwal et Subbian, 2012) utilise un algorithme basé sur les graphes pour détecter les évènements en ligne, chaque noeud représente un utilisateur et chaque lien une activité entre deux utilisateurs. L'algorithme prend en considération la structure et le contenu du réseau, si un objet constitue un nouveau cluster il est considéré comme étant un nouvel élément. G-Stream (Ghesmoune et al., 2015) est basé sur une carte auto-adaptative (Kranen et al., 2009), plus précisément la Growing Neural Gas (Fritzke et al., 1995) qui est une approche auto-adaptative incrémentale. G-Stream cherche à trouver une structure topologique qui correspond à la structure des données. La carte auto-adaptative est représentée par un graphe où chaque noeud représente un cluster. Pour chaque nouvel élément le plus proche et le deuxième plus proche noeud sont identifiés et sont reliés par une arête, ensuite le noeud le plus proche est déplacé vers le nouvel élément.

Dans ce papier, nous présentons notre approche NG-Stream pour le traitement et la visualisation des flux de données. NG-Stream est une combinaison entre les représentants de partition et les graphes de voisinage. Un graphe est construit incrémentalement pour représenter le profil de voisinage du flux de données en temps réel. Chercher à identifier le voisin de chaque nouvel élément peut être très coûteux, pour cela, au lieu de traiter chaque élément dès son arrivée, nous traitons un ensemble de nouveaux éléments simultanément afin d'identifier les nouveaux clusters. Ensuite, nous essayons de trouver le voisin du représentant de chaque nouveaux cluster afin de mettre à jour le graphe du voisinage du flux. NG-Stream construit donc le graphe de voisinage tout en réduisant le temps et les calculs nécessaires pour affecter les nouveaux éléments aux clusters. NG-Stream permet également une visualisation continue des résultats du clustering sous la forme d'un graphe de voisinage.

3 NG-Stream et le traitement des flux de données

Le but de NG-Stream est de construire un graphe de voisinage permettant de traiter et de visualiser les flux de données d'une façon continue. Dans notre approche, plutôt que de traiter chaque nouvel élément individuellement dès son arrivée, nous traitons un ensemble de nouveaux éléments simultanément. Cela permet de prendre en compte les caractéristiques d'un groupe de données arrivant dans la même période temporelle.

Soit $E = \{e_1, e_2, ...\}$ l'ensemble des éléments du flux F où la cardinalité $|E|$ (la taille du flux) est inconnue. Soit $G = (V, E, p)$ un graphe pondéré non orienté qui représente les clusters du flux en temps réel, chaque noeud représente un élément, chaque arête représente une relation de voisinage entre deux éléments, et le poids d'une arête représente la valeur de la distance entre les deux éléments.

En premier lieu, nous attendons l'arrivée du premier groupe d'éléments. Ces éléments constituent la première fenêtre $f_1 = \{e_{1.1}, e_{1.2}, ..., e_{1.n}\}$. La taille des fenêtres $|f| = n$ est fixée par l'utilisateur suivant son expertise et ses préférences. Nous appliquons un clustering basé sur le voisinage sur les éléments de la première fenêtre f_1 : Nous calculons la distance entre chaque couple d'éléments de f_1 : $(e_{1.i}, e_{1.j})$ où $i \in \{1, ..., n\}$, $j \in \{1, ..., n\}$ et $i \neq j$.

Nous considérons deux éléments $e_{1.i}$, $e_{1.j}$ comme voisins si la distance entre ces deux éléments est inférieure à un seuil *s*, ce dernier est également choisi par l'utilisateur. Nous considérons que chaque ensemble de voisins constitue un cluster, un ensemble de voisins est un groupe d'éléments où chaque élément a une relation de voisinage avec au moins un autre élément du même groupe. $C = \{c_1, c_2, ..., c_p\}$ est l'ensemble des clusters obtenus, p est le nombre des clusters de f_1. Nous déterminons ensuite le centroid de chaque cluster : nous calculons le centre de gravité de chaque cluster, ensuite l'élément le plus proche du centre de gravité est considéré comme étant le centroid du cluster. Les clusters obtenus sont représentés dans le graphe G comme expliqué précédemment. Un cluster est représenté par ces éléments et par la distance entre ses éléments et le centroid du cluster.

Les éléments du groupe suivant $f_2 = \{e_{2.1}, e_{2.2}, ..., e_{2.n}\}$ sont, dans un premier temps, traités indépendamment et avec le même processus que les éléments de f_1. De la même manière nous obtenons les clusters $C_{new} = \{c_1, c_2, ..., c_q\}$; où q est le nombre de clusters du deuxième groupe ; et nous identifions également les centroids des nouveaux clusters. L'ensemble des clusters C_{new} est utilisé pour mettre à jour le graphe G. Pour un nouveau cluster, nous cherchons parmi les précédents clusters le plus proche : nous calculons la distance minimale entre son centroid et les centroids des clusters précédents, si la distance minimale entre deux centroids est inférieure au seuil s, les deux clusters sont reliés. Cela se traduit par la création d'une arête entre les noeuds représentant les deux centroids représentant la distance entre eux. Dans le cas ou un nouveau cluster n'est proche d'aucun des clusters précédents, il est rajouté au graphe sans qu'il ne soit relié avec un autre cluster. Représentant l'apparition d'un nouveau cluster dans le flux de données.

Ainsi de suite, chaque groupe d'éléments qui arrive participe, d'une façon continue et incrémentale, à la construction du graphe G. Affecter un groupe de nouveaux éléments aux clusters, en se basant seulement sur des éléments représentatifs des clusters (les centroids), évite de comparer les éléments un à un et d'augmenter la complexité.

Dans le but d'éviter que le nombre d'éléments dans le graphe augmente au point de saturer la visualisation, nous fixons un nombre maximal d'éléments à visualiser. Au delà de ce nombre, nous supprimons les plus anciens clusters qui ont disparu du flux. Un cluster disparu du flux est un cluster qui n'a pas été mis-à-jour depuis une longue période. Pour chaque cluster nous sauvegardons la date de sa création (ou de sa dernière mise-à-jour), afin que cette information puisse être utilisée pour vérifier sa présence ou sa disparition du flux. Le graphe et ses changements sont visualisés par l'utilisateur en temps réel.

NG-Stream peut également traiter des données mixtes (constituées de variables numériques et catégorielles). Cela est possible grâce à la quantification des valeurs des variables catégorielles en utilisant la mesure de Shih (Shih et al., 2010). La mesure de Shih est une fonction qui, en se basant sur la co-occurrence de valeurs des variables, remplace les valeurs des variables catégorielles par des valeurs numériques. Une co-occurrence est la présence simultanée d'au moins deux valeurs d'attributs dans la même donnée. La mesure de Shih suppose que si deux valeurs d'attributs apparaissent souvent simultanément, ça implique qu'il y a une grande ressemblance entre ces valeurs.

L'algorithme 1 ci-dessous décrit le déroulement des étapes de la construction incrémentale du graphe de voisinage.

Algorithm 1 NG-Stream

ENTRÉES : $E = \{e_1, e_2, ...\}$; $G = (V, E, p)$; n ; $seuil$
SORTIES : Des clusters.
DEBUT
Attendre l'arrivée des n premiers éléments.
Calculer la distance entre chaque couple d'éléments.
Si $distance(e_1, e_2) < seuil$ **Alors**
e_1, e_2 sont voisins
Finsi
Chaque ensemble de voisins (un ensemble de voisins reliés par une relation de voisinage) = Un cluster.
Trouver le centroid de chaque cluster
Représenter chaque cluster par ses éléments et les arêtes entre le centroid et ses voisins dans le graphe G
Tantque le flux continue **Faire**
Attendre l'arrivée de n éléments.
Calculer la distance entre chaque couple d'éléments.
Trouver le centroid de chaque nouveau cluster.
Mettre à jour G : Calculer la $min\_distance$ entre chaque nouveau centroid et les centroids des clusters précédents.
Si $min\_distance < seuil$ **Alors**
Relier les centroids du nouveau et de l'ancien cluster
Sinon
Ajouter le nouveau cluster à G sans le relier
Finsi
Fin tantque
FIN.

4 Expérimentations

Nous avons testé NG-Stream avec différents jeux de données labellisés. Ces jeux de données ont des tailles différentes et un nombre différent d'attributs :

— Parkinson (Little et al., 2007) : des enregistrements vocaux décrits par 26 attributs de quarante personnes dont vingt sont atteintes de la maladie de Parkinson (197 instances numériques).
— QSAR (Mansouri et al., 2013) : les valeurs de 41 attributs (déscripteurs moléculaires) utilisés pour définir si un produit est biodégradable ou non (1055 instances numériques).
— Banknote (Lichman(a), 2013) : des données d'images prises pour l'évaluation d'une procédure d'identification de billets de banque. L'ensemble de données est décrit par 5 attributs (1372 instances numériques).
— Knowledge (Kranen et al., 2009) : contient des informations sur les niveaux de connaissances d'utilisateurs (classifiées par une machine de learning). 5 attributs décrivent ce jeu de données (403 instances numériques).

— EEG (Roesler, 2013) : des données d'une électroencéphalographie qui mesure l'activité électrique du cerveau. Les données contiennent 14 attributs (14980 instances numériques).
— Gas (Fonollosa et al., 2015) : des séries temporelles de 16 capteurs chimiques. Les données sont décrites par 19 attributs (4178504 instances numériques).
— KDD99 (Lichman(b), 2013) : des données de pare-feu réseau décrites par 41 attributs (4000000 instances mixtes contenant des variables numériques et catégorielles).

Pour nos expérimentations, nous utilisons la totalité des instances des quatre premiers jeux de données, et une partie des trois derniers jeux de données (10000 instances). Deux types d'évaluations sont utilisés : une évaluation externe et une évaluation interne. Dans l'évaluation externe nous comparons les résultats obtenus par NG-Stream et par les algorithmes CluStream (Aggarwal et al., 2003), ClusTree (Kranen et al., 2009) et DStream (Chen et Tu, 2007), avec les résultats attendus. Les critères d'évaluation externe utilisés (table 1) mesurent la similarité entre les clusters obtenus et les classes réelles des jeux de données. Les valeurs de la similarité sont comprises entre 0 et 1, où 1 signifie que les résultats sont identiques (Desgraupes, 2013).

Ensuite, nous utilisons des critères d'évaluation internes (table 2) pour évaluer la qualité des clusters obtenus par chaque algorithme en termes de compacité et séparabilité des clusters. Nous comparons également le temps d'exécution des quatre algorithmes. La comparaison avec les autres algorithmes est faite uniquement avec les jeux de données numériques. Nous testons également NG-Stream avec des données mixtes, et nous étudions l'impact des paramètres (la taille des fenêtres et le seuil de distance) sur les résultats et sur le temps d'exécution.

Critère	Fonction
Czekanowski Dice	$CD = \frac{2\|A \cap B\|}{\|A\|+\|B\|}$
Fowlkes-Mallows	$FM = \sqrt{\frac{TP}{TP+FP} \times \frac{TP}{TP+FN}}$
Jaccard	$J = \frac{\|A \cap B\|}{\|A \cup B\|}$
Kulczynski	$K = \frac{2\times\|A\|\cap\|B\|}{\|A\|+\|B\|-2\times\|A\cap B\|}$
Precision	$P = \frac{TP}{TP+FP}$
Rand index	$RI = \frac{a+b}{a+b+c+d}$
Adjusted Rand Index	$ARI = \frac{Index-ExpectedIndex}{MaxIndex-ExpectedIndex}$
Recall	$Recall = \frac{\|A\|\cap B\|}{\|B\|}$
Rogers Tanimoto	$RT = \frac{a+d}{a+2(b+c)+d}$

TAB. 1: Critères d'évaluation externe.

A et B sont les éléments du cluster et les éléments de la classe respectivement.
TP : True Positives, FP : False Positives, FN : False Negatives.
a : le nombre des paires d'éléments qui sont dans le même cluster et la même classe.
b : le nombre des paires d'éléments qui sont dans des clusters différents et des classes différentes.
c : le nombre des paires d'éléments qui sont dans le même cluster et dans des classes différentes.
d : le nombre des paires d'éléments qui sont dans des clusters différents et dans la même classe.

Critère	Fonction
Calinski Harabasz	$CH = \frac{SS_B}{SS_W} \times \frac{N-k}{k-1}$
Davies Bouldin	$DB = \frac{1}{c}\sum_{i=1}^{k} Max_{i \neq j} \frac{d(X_i)+d(X_j)}{d(c_i,c_j)}$
Dunn	$D = min_{1\leq i\leq k}\{min\{\frac{d(c_i,c_j)}{max_{1\leq p\ k(d(X_p))}}\}\}$
Silhouette	$S(i) = \frac{(b(i)-a(i))}{Max\{a(i)-b(i)\}}$

TAB. 2: Critères d'évaluation interne.

k est le nombre de clusters.
$SS_B = \sum_{i=1}^{k} n_j \|m_i - m\|^2$, $SS_W = \sum_{i=1}^{k}\sum_{x\in c_i} \|x - m_i\|^2$ où : m_i est le centre du cluster i, m la moyenne des données, x est un point, c_i est le ième cluster, et $\|v_1 - v_2\|$ est la distance euclidienne entre v_1 et v_2.
$d(X_i)$ est la moyenne de la distance entre chaque élément du cluster i et le centre du cluster.
$d(c_i, c_j)$ est la distance entre les centres des deux clusters i et j
$a(i)$ est la distance moyenne entre un élément i est le reste des éléments du cluster.
$b(i)$ est la distance minimale de l'élément i par rapport aux autres éléments du cluster.

4.1 Evaluation externe

Avant de comparer NG-Stream avec les autres algorithmes, nous testons l'impact du choix du seuil de la distance sur nos résultats. Nous faisons varier le seuil de la distance entre 0 et une très grande valeur. Nous comparons ensuite les résultats obtenus lors de chaque itération en utilisant les critères d'évaluation externes. les figures de 1 à 7 illustrent les résultats obtenus.

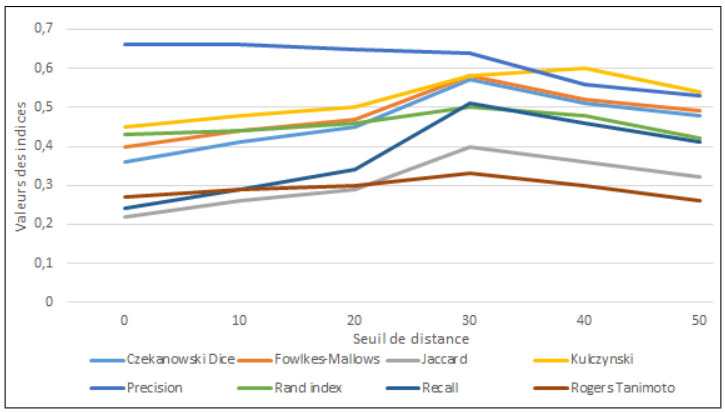

FIG. 1: Indices de l'évaluation externe selon le seuil - jeu de données Parkinson

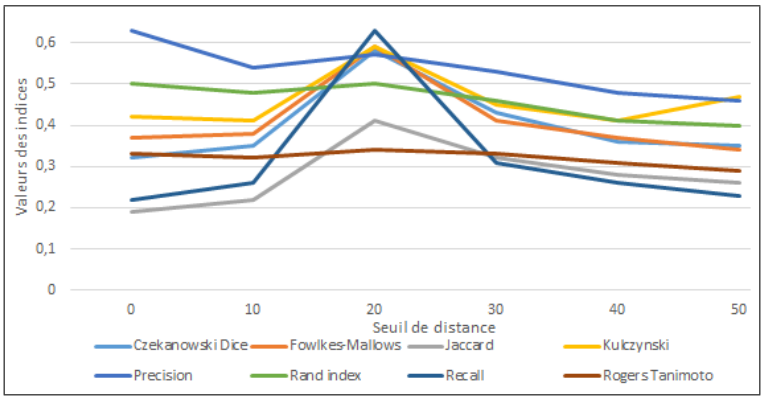

FIG. 2: Indices de l'évaluation externe selon le seuil - jeu de données QSAR

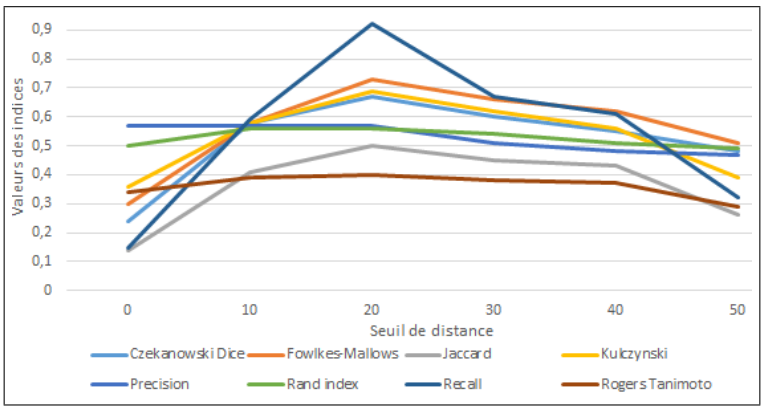

FIG. 3: Indices de l'évaluation externe selon le seuil - jeu de données Banknote

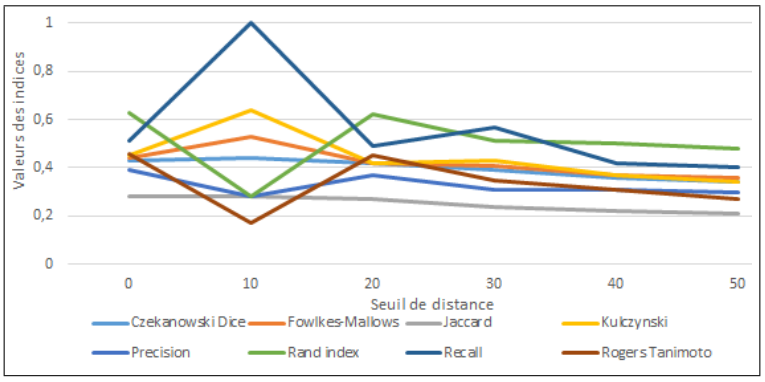

FIG. 4: Indices de l'évaluation externe selon le seuil - jeu de données Knowledge

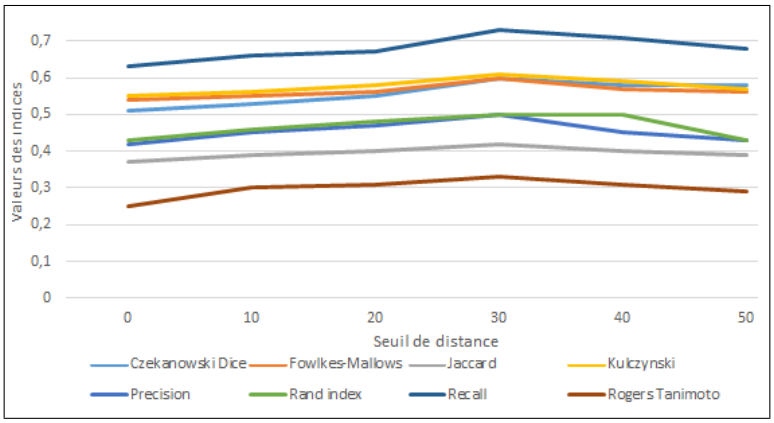

FIG. 5: Indices de l'évaluation externe selon le seuil - jeu de données EEG

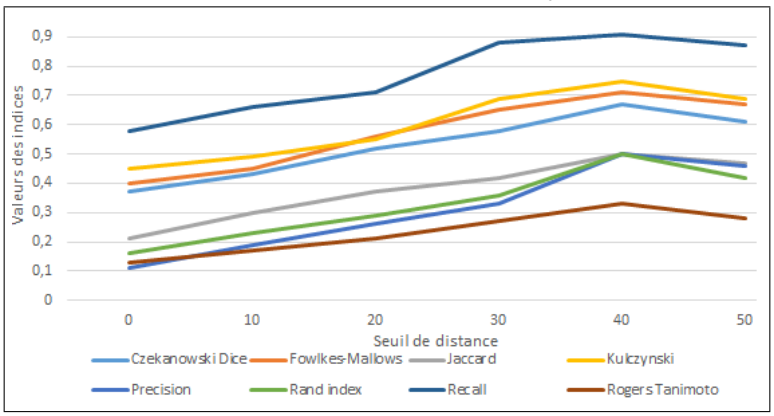

FIG. 6: Indices de l'évaluation externe selon le seuil - jeu de données Gas

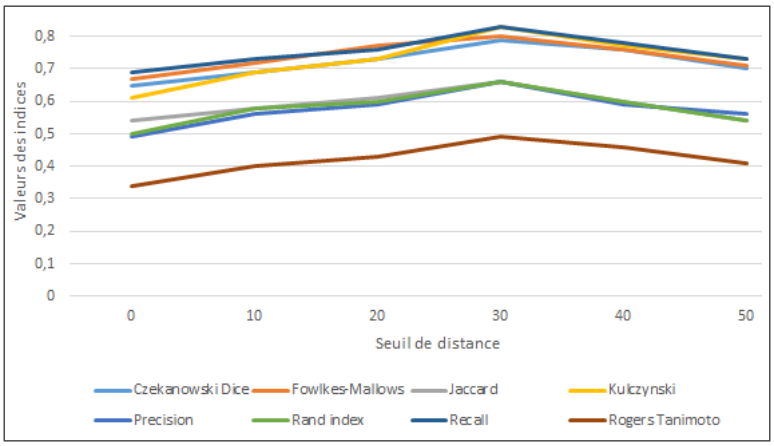

FIG. 7: Indices de l'évaluation externe selon le seuil - jeu de données KDD99

Nous remarquons qu'effectivement le seuil à un impact sur les résultats obtenus, les valeurs des critères externes font un pic presque simultanément, la valeur du seuil à ce moment là représente la valeur optimale pour le jeu de données.

Quand la valeur du seuil de distance est presque nulle, il n'y a que les éléments identiques qui peuvent être dans le même cluster, ce qui nous donne un grand nombre de clusters. Quand la valeur du seuil est très grande, tous les éléments sont regroupés dans le même cluster.

Nous comparons maintenant les résultats obtenus par NG-Stream avec les résultats optimaux obtenus par Clustream, ClusTree et DStream. Les valeurs des critères externes de chaque algorithme sont représentées par les histogrammes ci-dessous (figures de 8 à 13). Comme il est expliqué précédemment, si la valeur d'un de ces critères externes est égale ou se rapproche de 1, cela signifie que les résultats obtenus sont identiques avec ceux attendus. Les résultats de l'évaluation montrent que NG-Stream obtient des résultats aussi bons que les autres algorithmes.

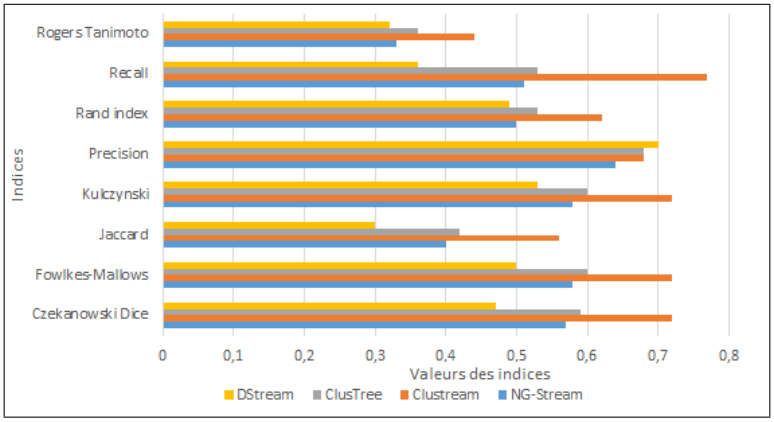

FIG. 8: l'évaluation externe sur le jeu de données Parkinson

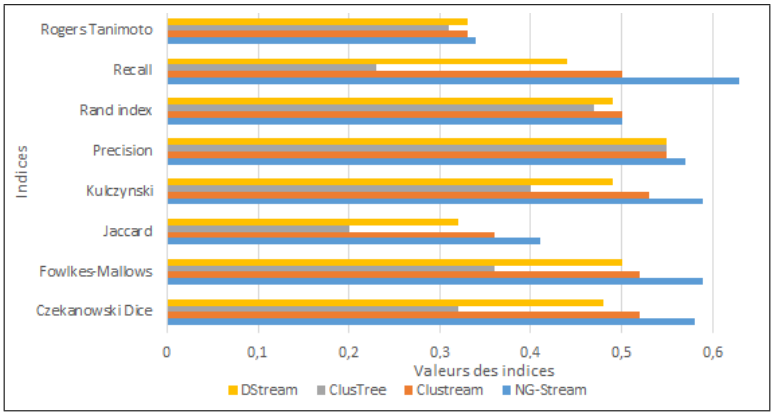

FIG. 9: l'évaluation externe sur le jeu de données QSAR

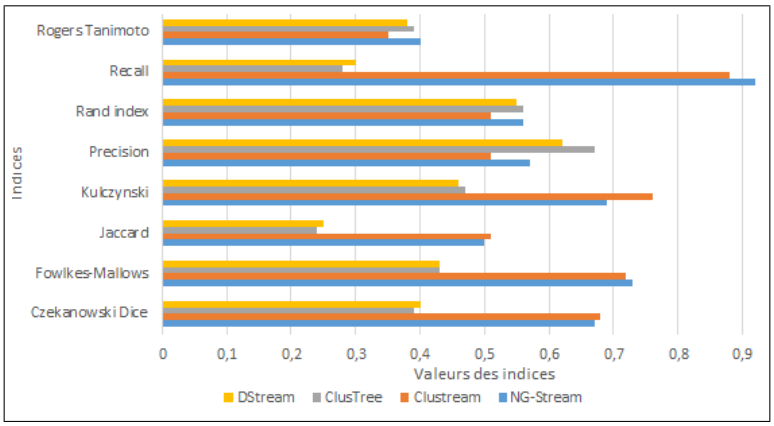

FIG. 10: l'évaluation externe sur le jeu de données Banknote

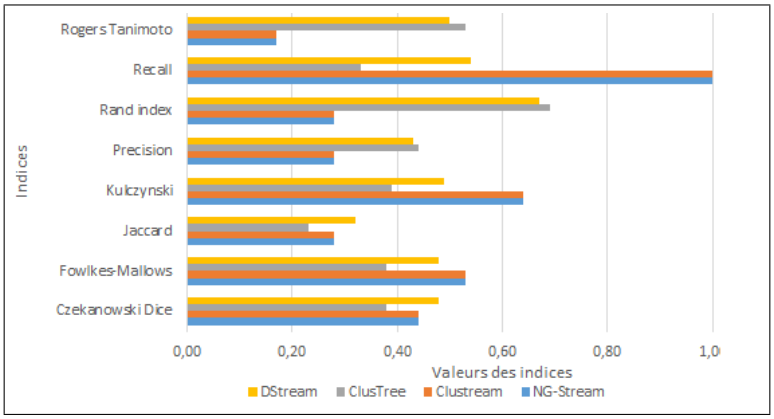

FIG. 11: l'évaluation externe sur le jeu de données Knowledge

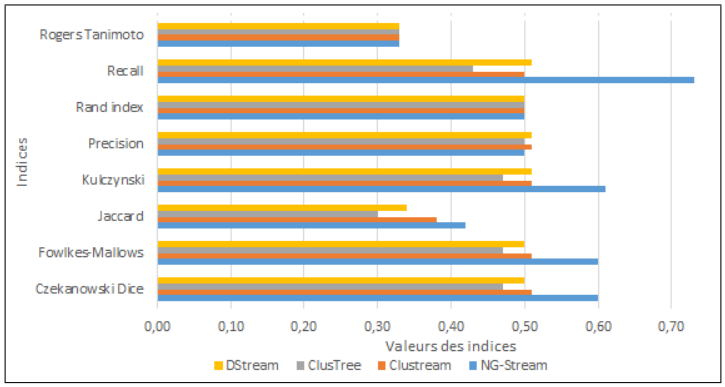

FIG. 12: l'évaluation externe sur le jeu de données EEG

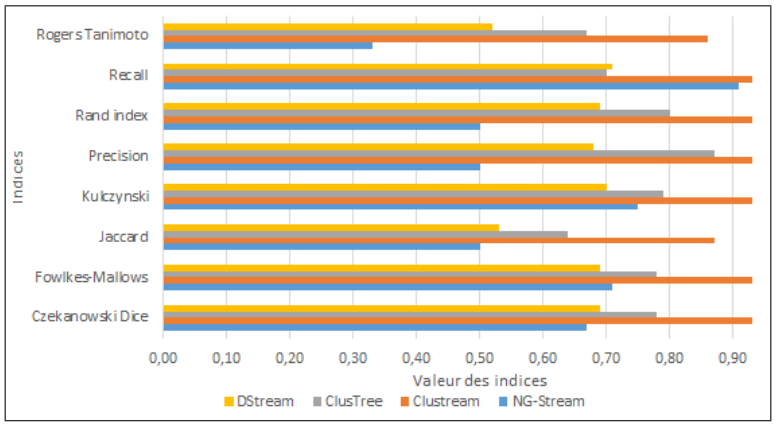

FIG. 13: l'évaluation externe sur le jeu de données Gas

4.2 Evaluation interne

Pour l'évaluation interne nous avons utilisé les mêmes valeurs de paramètres pour tous les algorithmes que ceux utilisés pour l'évaluation précédente. Les résultats obtenus en utilisant les critères d'évaluation internes sont compris dans les tables ci-dessous (tables de 3 à 6).

	NG-Stream	Clustream	ClusTree	DStream
Parkinson	**0,67**	1.12	1.88	1.85
QSAR	**18.32**	41.60	21.30	23.20
Banknote	**1.10**	1.12	1.94	2.95
Knowledge	**1.03**	**1.03**	1.69	1.53
EEG	1.98	2.24	**1.90**	8.29
Gas	2.34	**1.73**	2.40	2.70

TAB. 3: Résultats de l'évaluation selon le critère Davies Bouldin

	NG-Stream	Clustream	ClusTree	DStream
Parkinson	0.05	**0.07**	0.04	0,01
QSAR	**2.08**	1.32	1.12	1.48
Banknote	0.01	**0.02**	0.01	0.01
Knowledge	**1.04**	**1.04**	0.07	0.11
EEG	**9.44**	7.64	1.19	0,01
Gas	6.27	**7.53**	1.26	1.38

TAB. 4: Résultats de l'évaluation selon le critère Dunn

	NG-Stream	Clustream	ClusTree	DStream
Parkinson	0.43	**0.50**	0.31	-0,07
QSAR	**0.55**	0.43	0.41	0.20
Banknote	**0.36**	**0.36**	0.27	0.27
Knowledge	0.15	0.15	**0.18**	**0.18**
EEG	**0.02**	**0.02**	-0,13	0,10
Gas	0.01	**0.03**	-0.19	-0.13

TAB. 5: Résultats de l'évaluation selon le critère Silhouette

	NG-Stream	Clustream	ClusTree	DStream
Parkinson	87.93	104.16	**328.15**	12.80
QSAR	**1.73**	0.48	0,50	1.17
Banknote	117.43	**122.74**	104.16	88.67
Knowledge	37	36	**56**	51
EEG	**06.11**	5.15	4.27	5.10
Gas	0.30	**1.00**	0.48	0.19

TAB. 6: Résultats de l'évaluation selon le critère Calinski Harabasz

Les critères Calinski Harabasz, Dunn et Silhoutte sont à maximiser. Le critère Davies Bouldin est à minimiser (Desgraupes, 2013). La meilleure valeur pour chaque critère est marquée en gras dans les tableaux. Ces résultats montrent que selon le jeu de données, NG-Stream peut trouver des clusters de meilleure qualité que les autres algorithmes, ou des clusters qui se rapprochent de ceux trouver par Clustream.

4.3 NG-Stream et les données mixtes

Comme expliqué précédemment, NG-Stream peut également traiter les données mixtes grâce à la quantification des variables catégorielles en utilisant la mesure de Shih (Shih et al., 2010). Nous testons les performances de NG-Stream sur un jeu de données mixte (KDD99 (Lichman(b), 2013)). Nous utilisons les critères d'évaluation externes pour comparer les clusters obtenus avec les classes réelles des données, et parallèlement nous évaluons l'impact que peut avoir la taille des fenêtres sur les clusters obtenus. Les résultats sont illustrés sur la figure ci-dessous (figure 14).

Les résultats de l'évaluation externe montrent que NG-Stream est très performant sur un jeu de données mixte, les clusters obtenus se rapprochent des classes réelles selon la majorité des indices utilisés. Nous remarquons également que la taille des fenêtres n'impacte pas les résultats obtenus, nous avons varié la taille des fenêtres entre 20 et 100 éléments, et les résultats sont restés stables.

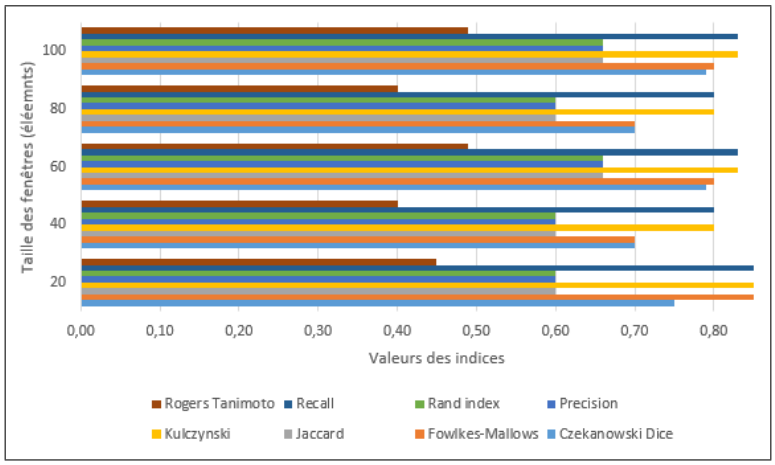

FIG. 14: Les indices d'évaluation externes selon la taille des fenêtres pour les données mixtes

4.4 Le temps d'exécution

Le temps d'exécution de NG-Stream a été aussi comparé avec ceux des trois autres algorithmes (table 7). Nous remarquons que NG-Stream a presque le même temps d'exécution que les autres algorithmes sur les petits jeux de données (Parkinson, QSAR, Banknote and Knowledge). Cependant, NG-Stream est beaucoup plus rapide sur les grands jeux de données (EEG and Gas). NG-Stream est plus rapide parce que il compare uniquement les centroids des nouveaux clusters avec ceux des clusters précédents.

	NG-Stream	Clustream	ClusTree	DStream
Parkinson	**0.01**	**0.01**	**0.01**	**0.01**
QSAR	0.04	**0.03**	0.04	**0.03**
Banknote	**0.01**	0.02	0.03	0.03
Knowledge	**0.01**	0.02	0.02	0.3
EEG	**0.26**	1.84	1.85	1.87
Gas	**0.63**	1.89	1.86	1.87

TAB. 7: Comparaison entre les temps d'exécution des algorithmes (secondes)

L'impact des paramètres de NG-Stream (la taille des fenêtres et le seuil de la distance) est illustré dans les tables 8 et 9 respectivement. Le choix des valeurs des paramètres de NG-Stream (la taille n et le seuil s) a un impact sur le temps d'exécution. Des fenêtres d'une plus grande taille impliquent plus de couples d'éléments à comparer, donc l'algorithme a besoin de plus de temps pour calculer la distance entre chaque couple d'éléments dans la même fenêtre.

Le seuil de la distance peut modifier le nombre de clusters dans une fenêtre de flux, une grande valeur réduit le nombre de clusters. Donc moins de temps est nécessaire pour calculer la distance entre les centroids des nouveaux clusters et des clusters précédents. Quand le seuil

de distance est supérieur ou égal à la valeur de la plus grande distance entre deux éléments, tous les éléments sont regroupés dans un seul cluster.

Taille des fenêtres	20	40	60	80	100
Parkinson	0.01	0.02	0.03	0.04	0.05
QSAR	0.04	0.08	0.12	0.16	0.20
Banknote	0.01	0.02	0.03	0.04	0.05
Knowledge	0.01	0.02	0.03	0.04	0.05
EEG	0.26	0.52	0.78	1.04	1.30
Gas	0.63	1.26	1.89	2.52	3.15
KDD99	0.46	0.92	1.38	1.84	2.30

TAB. 8: Temps d'exécution par rapport à la taille des fenêtres (secondes)

Seuil de distance	0	10	20	30	40	50
Parkinson	0.02	0.01	0.01	0.01	0.01	0.01
QSAR	0.16	0.08	0.05	0.05	0.05	0.05
Banknote	0.06	0.01	0.01	0.01	0.01	0.01
Knowledge	0.02	0.01	0.01	0.01	0.01	0.01
EEG	0.88	0.76	0.40	0.26	0.21	0.21
Gas	1.05	0.96	0.93	0.62	0.41	0.40
KDD99	1.39	1.33	1.05	0.86	0.43	0.41

TAB. 9: Temps d'exécution par rapport au seuil

4.5 Visualisation du flux

Un des principaux objectifs de notre approche est d'avoir un visuel graphique de l'évolution du flux. Nous souhaitons proposer une technique de visualisation qui reflète le processus du traitement, donc notre choix s'est porté naturellement sur une représentation sous la forme de graphes de voisinage (figures 15 à 18). Ce type de visualisation permet de représenter parfaitement les clusters obtenus, le changement du comportement du flux dans le temps et aussi l'apparition d'outliers (des éléments qui diffèrent du reste de l'ensemble de données). Ces derniers sont bien évidemment représentés par des clusters avec un très faible nombre d'éléments. Un outlier peut aussi représenter le début de création d'un nouveau cluster.

Dans les figures ci-dessus (figures de 15 à 18) nous avons effectué des captures à quatre instants de la visualisation du jeu de données Parkinson. Chaque capture est prise juste après la fin du traitement d'un nouveau groupe d'éléments à partir de T1. La taille de chaque groupe est fixée à 10 éléments. Les sous-graphes représentent les clusters détectés par NG-Stream. Les couleurs représentent les vraies classes des éléments (deux noeuds de la même couleur représentent deux éléments ayant la même classe dans l'ensemble de données d'origine). Dans la première capture d'écran (figure 15) nous avons un seul cluster composé des éléments déjà

traités du flux. Dans la deuxième capture (figure 16), un nouveau groupe est arrivé et il est constitué d'un seul cluster. le centroid du nouveau cluster est similaire au centroid du précédent cluster, donc les clusters ont été reliés. Dans la troisième capture (figure 17) le nouveau groupe contient également un seul cluster. Par contre, cette fois-ci le centroid du nouveau cluster ne ressemble pas au centroid du précédent cluster. Donc le cluster est rajouté dans le graphe de voisinage sans qu'il ne soit relié à aucun cluster. Nous remarquons aussi qu'un des nouveaux éléments (en bleu) est affecté au cluster vert selon le critère de la distance. Dans la quatrième capture (figure 18), le nouveau groupe contient deux clusters. Les deux nouveaux clusters sont reliés avec deux différents clusters précédents.

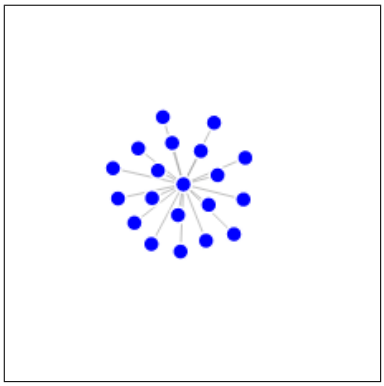

FIG. 15: Capture du graphe à T1

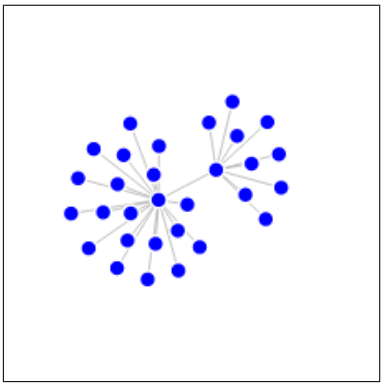

FIG. 16: Capture du graphe à T2

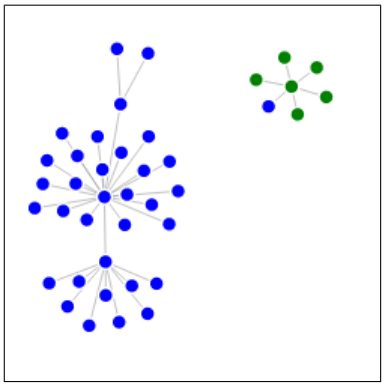

FIG. 17: Capture du graphe à T3

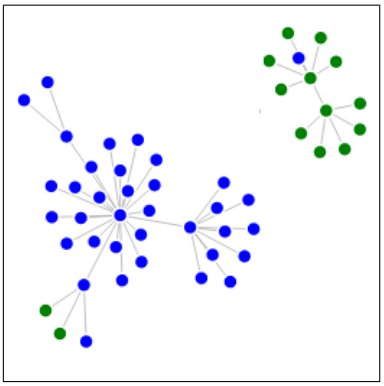

FIG. 18: Capture du graphe à T4

5 Conclusion

Dans ce papier nous avons présenté NG-Stream, une nouvelle approche pour le traitement des flux de données. La principale idée consiste à traiter par groupe les éléments arrivés presque simultanément au lieu de traiter chaque élément séparément. Dès l'arrivée du premier groupe, les clusters de ce groupe et les centroids des clusters sont identifiés. Ces clusters sont

représentés par un graphe où chaque noeud représente un élément et chaque arête représente la distance entre un élément et le centroid du cluster. Ce graphe est continuellement mis-à-jour tout au long du flux de données. Pour chaque nouveau groupe, les éléments sont traités en premier lieu séparément des précédents éléments. Quand les clusters du nouveau groupe sont déterminés, ils sont reliés avec les précédents clusters (en se basant sur la similarité entre les nouveaux et les précédents centroids). La construction incrémentale du graphe est visualisée par l'utilisateur, ceci est possible grâce à une visualisation sous forme de graphes de voisinage.

NG-Stream a été testé sur des jeux de données labellisés, et en même temps été comparé à trois algorithmes de clustering de flux de données (Clustream, ClusTree et DStream). Nous avons effectué une évaluation avec des critères externes et des critères internes du clustering. Les critères d'évaluation internes et externes ont montré que NG-Stream a de bons résultats. Notre approche est adaptée pour le traitement de données mixtes, des tests ont été également réalisés sur ce type de données. L'évolution du temps d'exécution a été aussi étudiée par rapport aux deux paramètres de NG-Stream (la taille des fenêtres et le seuil de distance). L'étude du temps d'exécution a montré que NG-Stream est aussi rapide que les autres algorithmes sur de petits jeux de données, et beaucoup plus rapide sur de grands jeux de données. Cela est dû au calcul de la distance entre uniquement les centroids des clusters au lieu de chaque couple d'éléments.

En perspective nous envisageons d'utiliser une technique pour proposer un seuil de distance et une taille de fenêtre à l'utilisateur qui s'adaptent suivant l'évolution du flux de données. Nous envisageons également de proposer une technique de sélection de variables qui soit efficace pour le traitement des flux de données. Notre objectif à long terme est d'améliorer et d'intégrer cette approche dans un environnement interactif de fouille visuelle de flux de données, d'où la nécessité de permettre une plus grande interaction de l'utilisateur avec le processus du traitement, cette interaction se fera principalement à travers la visualisation des résultats du clustering.

Références

Aggarwal, C. C., J. Han, J. Wang, et P. S. Yu (2003). A framework for clustering evolving data streams. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pp. 81–92. VLDB Endowment.

Aggarwal, C. C. et K. Subbian (2012). Event detection in social streams. In *SDM*, Volume 12, pp. 624–635. SIAM.

Berkhin, P. (2006). A survey of clustering data mining techniques. In *Grouping multidimensional data*, pp. 25–71. Springer.

Cao, F., M. Ester, W. Qian, et A. Zhou (2006). Density-based clustering over an evolving data stream with noise. In *Proceedings of 2006 SIAM Conference on Data Mining*, Volume 6, pp. 328–339. SIAM.

Chen, Y. et L. Tu (2007). Density-based clustering for real-time stream data. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 133–142. ACM.

Desgraupes, B. (2013). Clustering indices. *University of Paris Ouest-Lab Modal'X 1*, 34.

Fonollosa, J., S. Sheik, R. Huerta, et S. Marco (2015). Reservoir computing compensates slow response of chemosensor arrays exposed to fast varying gas concentrations in continuous monitoring. *Sensors and Actuators B : Chemical 215*, 618–629.

Forgy, E. W. (1965). Cluster analysis of multivariate data : efficiency versus interpretability of classifications. *Biometrics 21*, 768–769.

Fritzke, B. et al. (1995). A growing neural gas network learns topologies. *Advances in neural information processing systems 7*, 625–632.

Gabriel, K. R. et R. R. Sokal (1969). A new statistical approach to geographic variation analysis. *Systematic Biology 18*(3), 259–278.

Gama, J. (2010). *Knowledge discovery from data streams*. CRC Press.

Ghesmoune, M., M. Lebbah, et H. Azzag (2015). Clustering over data streams based on growing neural gas. In *Advances in Knowledge Discovery and Data Mining*, pp. 134–145. Springer.

Guha, S., N. Mishra, R. Motwani, et L. O'Callaghan (2000). Clustering data streams. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, FOCS '00, Washington, DC, USA, pp. 359–366. IEEE Computer Society.

Guha, S., R. Rastogi, et K. Shim (1999). Rock : A robust clustering algorithm for categorical attributes. In *Data Engineering, 1999. Proceedings., 15th International Conference on*, pp. 512–521. IEEE.

Karypis, G., E.-H. Han, et V. Kumar (1999). Chameleon : Hierarchical clustering using dynamic modeling. *Computer 32*(8), 68–75.

Kranen, P., I. Assent, C. Baldauf, et T. Seidl (2009). Self-adaptive anytime stream clustering. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pp. 249–258. IEEE.

Lavergne, J., H. Azzag, C. Guinot, et G. Venturini (2007). Incremental construction of neighborhood graphs using the ants self-assembly behavior. In *Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE International Conference on*, Volume 1, pp. 399–406. IEEE.

Lichman(a), M. (2013). Uci machine learning repository. `http://archive.ics.uci.edu/ml/datasets/banknote+authentication`. (consulté le : 16.11.2015).

Lichman(b), M. (2013). Uci machine learning repository. `https://archive.ics.uci.edu/ml/datasets/KDD+Cup+1999+Data`. (consulté le : 12.11.2015).

Little, M. A., P. E. McSharry, S. J. Roberts, D. A. Costello, I. M. Moroz, et al. (2007). Exploiting nonlinear recurrence and fractal scaling properties for voice disorder detection. *BioMedical Engineering OnLine 6*(1), 23.

Lühr, S. et M. Lazarescu (2009). Incremental clustering of dynamic data streams using connectivity based representative points. *Data & Knowledge Engineering 68*(1), 1–27.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Volume 1, pp. 281–297. Oakland, CA, USA.

Mansouri, K., T. Ringsted, D. Ballabio, R. Todeschini, et V. Consonni (2013). Quantitative structure–activity relationship models for ready biodegradability of chemicals. *Journal of*

chemical information and modeling 53(4), 867–878.

Ng, A. Y., M. I. Jordan, Y. Weiss, et al. (2002). On spectral clustering : Analysis and an algorithm. *Advances in neural information processing systems 2*, 849–856.

Roesler, O. (2013). Uci machine learning repository. `http://archive.ics.uci.edu/ml/datasets/EEG+Eye+State`. (consulté le : 16.11.2015).

Shih, M.-Y., J.-W. Jheng, et L.-F. Lai (2010). A two-step method for clustering mixed categroical and numeric data. *Tamkang Journal of Science and Engineering 13*(1), 11–19.

Silva, J. A., E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. de Carvalho, et J. Gama (2013). Data stream clustering : A survey. *ACM Computing Surveys (CSUR) 46*(1), 13.

Toussaint, G. T. (1980). The relative neighbourhood graph of a finite planar set. *Pattern recognition 12*(4), 261–268.

Toussaint, G. T. (1991). *Some unsolved problems on proximity graphs*. Citeseer.

Yao, A. C.-C. (1982). On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM Journal on Computing 11*(4), 721–736.

Summary

In this paper we deal with the real time processing and visualization of data streams. In order to deal with data streams we propose a novel approach that uses a neighborhood-based clustering. Instead of processing each new element one by one, we propose to process each group of new elements simultaneously. A clustering is applied on each new group using neighborhood graphs. The obtained clusters are then used to construct incrementally a representing graph of the data stream. The data stream graph is visualized in real time with specific visualizations that reflects the processing algorithm. To validate the approach, we apply it to multiple data sets and we compare it with various stream clustering approaches.