Data stream clustering using complex networks

Danilo A. Nunes[1], **Murillo G. Carneiro** [1]

[1]Faculdade de Computação – Universidade Federal de Uberlândia (UFU)

danilo.gusnunes@gmail.com, mgcarneiro@ufu.br

Abstract. *Data stream clustering is a crucial machine learning task for many systems that generate data continuously and require uninterrupted analysis. By adopting resources offered by the MOA (Massive Online Analysis) platform, the objective of this research is to apply learning models based on complex networks in the offline phase of CluStream, in which micro-clusters are grouped using the k-Means algorithm. For the experiments in this project, three databases and several performance measures specific to the problem were considered. The results showed that the use of complex networks has competitive performance, outperform traditional k-Means method in many scenarios.*

Resumo. *O agrupamento em fluxo de dados é uma tarefa de aprendizado de máquina crucial para vários sistemas que geram dados de maneira contínua e carecem de analisá-los ininterruptamente. Através de recursos oferecidos pela plataforma MOA (Massive Online Analysis), a proposta desta pesquisa consiste em aplicar modelos de aprendizado baseados em redes complexas na fase offline do CluStream, na qual micro-grupos são agrupados através do algoritmo k-Means. Para os experimentos desse projeto, foram consideradas três bases de dados e várias medidas de desempenho específicas ao problema. Os resultados mostraram que o uso de redes complexas apresenta desempenho competitivo, chegando a superar o método tradicional k-Means em diversos cenários.*

1. Introdução

Em decorrência do avanço contínuo da tecnologia, uma quantidade enorme de dados é produzida diariamente, com uma perspectiva de crescimento ainda maior. De acordo com [Gantz and Reinsel 2011], as informações digitais geradas em 2010 alcançaram o patamar de zettabytes (trilhões de gigabytes). Essa realidade é impulsionada por diversos sistemas que produzem sequências ilimitadas de dados, conhecidas como Fluxo de Dados.

Nesse sentido, o Aprendizado de Máquina é uma das principais áreas relacionadas à análise e extração de informações importantes a partir desse grande amontoado de dados [Gama et al. 2014]. O aprendizado de máquina possibilita que algoritmos computacionais aperfeiçoem a resolução de determinadas tarefas por meio da experiência [Mitchell 1997], resultando em sistemas cada vez mais inteligentes, capazes de descobrir conhecimentos de forma automática.

O agrupamento é uma das mais relevantes tarefas do aprendizado de máquina. Nessa técnica, os dados não possuem nenhuma informação de rótulo ou classe, tendo como objetivo descobrir a relação entre eles. No processo de agrupamento, os dados são separados de tal forma que aqueles pertencentes ao mesmo grupo devem ser mais parecidos entre si do que em comparação àqueles em grupos diferentes [Berkhin 2006].

Em abordagens tradicionais, um modelo de agrupamento é criado assumindo-se que todo o conjunto de dados para treinamento está disponível. No entanto, em fluxos de dados, essas abordagens são ineficientes, já que as informações chegam constantemente, e guardar esse extenso volume é inviável [Aggarwal et al. 2003]. Além disso, o ambiente dos fluxos de dados é dinâmico, exigindo atualizações contínuas do modelo à medida que os dados chegam, pois as suas propriedades podem mudar no decorrer do tempo.

Especificamente sobre a tarefa de agrupamento, a maioria das técnicas consideram os dados no formato de vetor de atributos, e os grupos são formados com base em alguma medida de similaridade [Berkhin 2006]. Nos últimos anos, o uso de Redes Complexas para caracterizar a formação de grupos tem se tornado um tópico de grande interesse. Redes Complexas são uma forma de representação de sistemas reais por meio de grafos e é utilizada em diversas áreas do conhecimento como Sociologia, Biologia e Computação [Fortunato 2010]. O processo de organizar a rede em grupos é denominado detecção de comunidades.

1.1. Objetivo geral

O objetivo geral deste trabalho de pesquisa consiste em aplicar modelos de aprendizado baseados em redes complexas no contexto de agrupamento em fluxo de dados, a fim de se obter resultados relevantes a partir da introdução dessa abordagem. Para isso, foi selecionado o algoritmo de agrupamento em fluxo de dados CluStream, em que sua fase offline determinada pelo método convencional de agrupamento k-Means será substituída pelas técnicas de detecção de comunidades Fastgreedy e Louvain.

A motivação para o desenvolvimento dessa pesquisa reside no fato de que métodos tradicionais de agrupamento consideram somente a estrutura física dos dados, enquanto redes complexas possibilitam também a análise da organização estrutural, topológica e funcional dos dados. Inclusive, trabalhos recentes da literatura mostraram que essa é uma abordagem de aprendizado saliente, capaz de alcançar resultados superiores aos algoritmos tradicionais, como no problema de agrupamento de objetos invariantes [Freitas and Carneiro 2019].

1.2. Objetivos específicos

- Geração de um conjunto de bases de dados com diferentes características.
- Reprodução de técnicas de agrupamento e das principais medidas de desempenho da literatura.
- Desenvolvimento de estratégias específicas de formação da rede para o contexto de agrupamento em fluxos de dados.
- Comparação entre os resultados dos algoritmos de detecção de comunidades com os do método de agrupamento tradicional k-Means.

2. Fundamentação Teórica

2.1. Fluxo de dados

De acordo com [Aggarwal et al. 2003], um fluxo de dados consiste em um conjunto de dados multidimensionais $\overline{X_1},\ \overline{X_2},\ \ldots\ ,\overline{X_k},\ \ldots$, que chegam em marcadores de tempo $\overline{T_1},\ \overline{T_2},\ \ldots\ ,\overline{T_k},\ \ldots$. Cada $\overline{X_i}$ é um registro multidimensional contendo d dimensões, denotado por $\overline{X_i} = ({x_i}^1,\ {x_i}^2, \ldots\ , {x_i}^d)$.

Algoritmos de aprendizado de máquina que lidam com fluxo de dados devem seguir algumas restrições que introduzem limite de tempo, memória e adaptação a mudanças, a fim de que se mantenham sempre atualizados [Nguyen et al. 2014]:

- **Passada única:** as amostras fluem uma seguida das outras, uma vez que não ficam armazenadas devido ao grande volume, cada uma deve ser analisada na ordem que chega, no máximo apenas uma vez e não pode ser recuperada.
- **Resposta em tempo real:** muitos sistemas necessitam de resposta em tempo real, dessa forma, o modelo deve ser capaz de processar as informações e fornecer resultados de forma ágil.
- **Memória limitada:** em razão da quantidade de dados gerados ser superior àquela suportada pela memória, o que se faz é computar e armazenar sumários do fluxo, descartando assim, o restante das informações.
- **Detecção de mudança de conceito:** mudança de conceito se refere a mudanças sofridas nos padrões dos dados ao longo do tempo.

2.2. Agrupamento em fluxo de dados

Diversas técnicas foram desenvolvidas com o propósito de lidar com informações geradas em larga escala, como é o caso do Vetor de Atributos do Grupo (*Cluster Feature Vector* $-$ CF-Vetor). Nesse conceito [Zhang et al. 1996], somente uma representação compacta dos dados é guardada.

Considerando N dados de d dimensões em um grupo $\{X_i\}$ com $i = 1, 2, 3, ..., N$, o vetor de atributos do grupo é definido como uma tripla CF-Vetor $= (N, LS, SS)$, em que N é o número de exemplos no grupo, LS é a soma linear dos N exemplos e SS é a soma dos quadrados dos N exemplos.

Entre os principais algoritmos de agrupamento em fluxo de dados, destaca-se o CluStream [Aggarwal et al. 2003]. Nessa técnica, os dados são agrupados através de um processo dividido em duas fases: online de micro-grupos e offline de macro-grupos. Na fase online, é armazenada uma sumarização estatística dos dados, mantida através de micro-grupos: extensões temporais dos vetores de atributos do grupo. Além dos três componentes do CF-Vetor, um micro-grupo contém também a soma dos marcadores de tempo ($CF1^t$) e a soma dos quadrados dos marcadores de tempo ($CF2^t$). Já no estágio offline, os micro-grupos são agrupados utilizando-se o método k-Means [Lloyd 1982].

Primeiramente, o k-Means é executado sobre um conjunto inicial de objetos do fluxo, gerando um total q de micro-grupos, definido previamente. Assim que um novo exemplo chega, há duas possibilidades, ser incluído em um micro-grupo já existente ou criar um novo micro-grupo para integrar inicialmente esse dado. Essa avaliação é feita com base na distância entre o exemplo e um valor limite (chamado de raio) calculado para o micro-grupo. Depois de encontrado o micro-grupo mais próximo, se a distância entre ele e o exemplo estiver dentro do limite, o micro-grupo absorve o dado, caso contrário, um novo micro-grupo deve ser criado.

Um exemplo que não é incorporado a um micro-grupo corresponde a um ruído, ou o começo de um novo grupo, ou até mesmo o começo de um novo movimento nos grupos. Nesse caso, com um novo micro-grupo criado, a fim de se manter a quantidade q, um micro-grupo mais antigo deve ser removido ou outros dois devem ser unidos. Dado um limiar de tempo, se um micro-grupo excede esse valor sem ter sido atualizado, esse

micro-grupo deve ser removido. Se nenhum micro-grupo ultrapassou esse limite, são escolhidos outros dois mais próximos entre si para serem fundidos.

2.3. Redes complexas

Uma rede é uma abstração que permite relacionar pares de objetos que interagem entre si. Assim, por se tratar de um princípio simples capaz de capturar características essenciais de sistemas, as redes têm grande relevância na compreensão complexa do mundo real [Bornholdt and Schuster 2003].

Redes que surgem naturalmente a partir de sistemas reais são chamadas de redes complexas, cuja estrutura é irregular, complexa e evolui dinamicamente com o tempo. Esse tipo de rede descreve uma grande variedade de sistemas, como a World Wide Web, redes neurais, redes metabólicas, redes organizacionais, redes de citações entre artigos e muitas outras [Newman 2003].

Em termos matemáticos, uma rede é representada por um grafo. Um grafo $G = (V, E)$ é formado por um par de conjuntos finitos V e E, em que os n elementos de V são os vértices do grafo G e os m elementos de E são suas arestas [Diestel 2005]. Cada aresta consiste em um subconjunto de dois elementos de V. Então, tem-se que as entidades da rede correspondem aos vértices e as interações entre elas são estabelecidas pelas arestas.

2.4. Comunidades em redes complexas

A estrutura de comunidade em redes complexas é caracterizada pela divisão dos vértices em grupos densamente conectados em seu interior, porém, com uma baixa concentração de conexões externas. Dessa forma, as comunidades (ou ainda, grupos ou módulos) são conjuntos de vértices que provavelmente compartilham propriedades comuns ou têm funções semelhantes dentro do grafo [Fortunato 2010].

A detecção de comunidades consiste em identificar grupos da rede com base apenas nas informações presentes na topologia do grafo. Entre os principais métodos de detecção de comunidades, destacam-se o Fastgreedy [Clauset et al. 2004] e Louvain [Blondel et al. 2008], que utilizam a modularidade como medida para avaliar a qualidade de uma determinada divisão da rede.

A modularidade [Newman 2004] realiza uma comparação entre a real densidade de arestas de uma comunidade e a densidade esperada de arestas para o mesmo grupo de vértices em um grafo com conexões geradas ao acaso. O cálculo da modularidade é baseado na hipótese de que grafos aleatórios não possuem uma estrutura de comunidade inerente [Barabási and Pósfai 2016]

Nesse sentido, uma comunidade em potencial deve conter um número maior de arestas internas que o esperado em um modelo de grafo aleatório. Logo, a equação da modularidade [Fortunato 2010] é definida como

$$Q = \sum_{c=1}^{n_c} \left[\frac{l_c}{m} - \left(\frac{d_c}{2m} \right)^2 \right], \tag{1}$$

em que n_c é o número de comunidades, l_c é o número total de arestas entre os vértices da comunidade c e d_c é o somatório dos graus dos vértices de c. O primeiro termo de cada soma representa a fração de arestas do grafo dentro da comunidade, enquanto o segundo

termo é a fração esperada de arestas em um grafo aleatório no qual a sequência de grau esperada é mesma que do grafo original.

2.4.1. Fastgreedy

Nesse método, cada vértice é inicialmente atribuído a uma comunidade diferente. Dessa maneira, em cada iteração, pares de comunidades são unidos de forma a maximizar o aumento (ou minimizar a diminuição) da modularidade. O algoritmo é finalizado quando todos os vértices estiverem agrupados em uma única comunidade.

Portanto, o Fastgreedy é identificado como um método hierárquico aglomerativo, em que as comunidades que antes estavam separadas, são unidas em diferentes níveis de organização. O algoritmo seleciona o melhor nível levando-se em conta aquele em que a divisão da rede produz o maior valor de Q.

2.4.2. Louvain

Essa técnica é dividida em duas fases que são repetidas de forma iterativa. Na primeira fase, os vértices são inicialmente definidos como comunidades separadas. Em seguida, cada vértice é movido para a comunidade de um dos seus vizinhos que resulte no maior aumento da modularidade, mas somente se esse ganho for positivo. Esse processo é aplicado repetidamente a todos os vértices até que nenhuma melhoria seja alcançada, encerrando a primeira fase.

Já na segunda fase, uma nova rede é construída, os vértices agora correspondem às comunidades encontradas na fase anterior. Assim que a segunda fase é concluída, torna-se possível reaplicar a primeira fase nessa nova rede. Dessa forma, até que não haja mais mudanças e um valor máximo de modularidade seja alcançado, a combinação dessas duas fases é repetida.

3. Metodologia

3.1. Algoritmos e ambiente computacional

A fase online do CluStream foi executada através da plataforma MOA [Bifet et al. 2010], em sua versão "2021.07". O MOA é um framework de código aberto utilizado para mineração em fluxo de dados, possui uma coleção de algoritmos de aprendizado de máquina e ferramentas para avaliação de desempenho.

Já para a etapa offline do CluStream, o módulo Igraph [Csardi and Nepusz 2005], em sua versão "0.9.1" com a linguagem Python, foi utilizado tanto na formação da rede de micro-grupos quanto na execução do Fastgreedy e Louvain. O igraph se configura como uma coleção de ferramentas para criação, manipulação e análise de redes, com ênfase na eficiência, portabilidade e facilidade de uso.

A escolha do Fastgreedy se deu em virtude de análises preliminares dessa técnica obterem resultados significativos em relação a métodos convencionais de agrupamento [Freitas and Carneiro 2019]. Enquanto o Louvain foi selecionado principalmente pelo fato desse algoritmo, assim como o Fastgreedy, possuir um desempenho rápido, com tempo de execução quase linear em grafos esparsos [Csardi and Nepusz 2005].

3.2. Formação da rede de micro-grupos

Para aplicar os métodos Fastgreedy e Louvain no estágio offline do CluStream, é preciso transformar os micro-grupos formados na fase online para uma representação em grafo, visto que esses objetos se encontram originalmente no formato de vetor de atributos. Essa conversão é feita utilizando-se o grafo "k vizinhos mais próximos" ($GkNN$), em que cada dado se conecta aos seus k dados mais semelhantes, tendo em vista alguma medida de similaridade ou distância [Carneiro and Zhao 2018, Carneiro et al. 2021].

Para gerar a rede, o algoritmo $GkNN$, primeiramente, define cada micro-grupo x como um vértice v na rede. Após isso, é calculada a similaridade entre todos os pares de micro-grupos x_i e x_j, nesse caso, utilizando-se a distância Euclidiana. Assim, uma conexão e_{ij} é criada entre os vértices v_i e v_j se, e somente se, x_j pertence ao conjunto dos k elementos mais similares a x_i, definido como $GkNN^{x_i}$, isto é:

$$e_{ij} = \begin{cases} 1, & \text{se } x_j \in GkNN^{x_i}, \\ 0, & \text{caso contrário.} \end{cases} \tag{2}$$

A partir dessa definição, o $GkNN$ produz um grafo direcionado (as arestas têm direção de orientação), inviabilizando o uso do Fastgreedy e Louvain no igraph, uma vez que ambas implementações suportam como entrada apenas grafos não direcionados. Então, foram empregadas duas variações que convertem a rede direcionada original em uma rede não direcionada: $GkNN$ Simétrico e $GkNN$ Mútuo [Carneiro et al. 2019].

- No $GkNN$ Simétrico $(SGkNN)$, uma aresta não direcionada é criada para cada par de vértices conectados com pelo menos uma aresta direcionada, resultando em uma matriz de adjacência simétrica A', obtida a partir da matriz de adjacência original A do $GkNN$:
$$A'_{ij} = max(A_{ij}, A^T_{ij}) \tag{3}$$
- O $GkNN$ Mútuo $(MGkNN)$ cria uma aresta não direcionada entre um par de vértices se, e somente se, ambos estiverem entre os k vizinhos mais próximos um do outro. Então, a partir da matriz de adjacência A do $GkNN$, obtém-se a matriz de adjacência simétrica A' conforme:
$$A'_{ij} = min(A_{ij}, A^T_{ij}) \tag{4}$$

É importante ressaltar que a construção do $GkNN$ direcionado apresenta uma complexidade quadrática de tempo e espaço nesse trabalho de pesquisa, já que é calculada a matriz de distâncias euclidianas entre os micro-grupos. Entretanto, a complexidade de criação desse grafo pode ser reduzida utilizando-se estruturas de dados mais otimizadas, por exemplo, uma kD-Tree (Árvore kD).

3.3. Medidas de desempenho

Com a finalidade de analisar a qualidade dos grupos encontrados após a execução dos algoritmos, foram selecionados cinco critérios de desempenho: pureza, pureza inversa, medida F, CMM e coeficiente de silhueta. O MOA foi estendido para abrigar as medidas "pureza inversa" e "medida F", possuindo as demais já implementadas.

3.3.1. Pureza

A pureza de cada grupo é calculada pela divisão da quantidade de dados da classe mais frequente em relação ao total de dados do grupo, em outras palavras, a pureza mede a proporção do grupo ocupada pelos dados da classe majoritária [Amigó et al. 2009]:

$$Pureza = \frac{1}{NC} \sum_{i=1}^{NC} \max_{L_j \in L} Precisao\,(C_i, L_j), \tag{5}$$

$$Precisao\,(C_i, L_j) = \frac{|C_i \cap L_j|}{|C_i|}, \tag{6}$$

em que NC é a quantidade de grupos, C é o conjunto de grupos e L, o conjunto de classes.

3.3.2. Pureza Inversa

A pureza inversa resulta da divisão da maior quantidade de dados pertencentes a uma mesma classe agrupados juntos em relação ao total de dados dessa classe, isto é, a pureza inversa avalia para cada classe a maior proporção dos seus dados presentes em um mesmo grupo [Amigó et al. 2009]:

$$Pureza\ Inversa = \frac{1}{NL} \sum_{i=1}^{NL} \max_{C_j \in C} Precisao\,(L_i, C_j), \tag{7}$$

$$Precisao\,(L_i, C_j) = \frac{|L_i \cap C_j|}{|L_i|}, \tag{8}$$

em que NL é a quantidade de classes, C é o conjunto de grupos e L, o conjunto de classes.

3.3.3. Medida F

A medida F é uma combinação entre a pureza e pureza inversa, podendo ser obtida através da média harmônica entre esses dois conceitos. A partir desse cálculo, obtém-se valores no intervalo $[0, 1]$, do qual a maximização representa um melhor resultado:

$$Medida\ F = \frac{2 \times Pureza \times Pureza\ Inversa}{Pureza + Pureza\ Inversa}. \tag{9}$$

3.3.4. Cluster Mapping Measures (CMM)

O CMM [Kremer et al. 2011] foi desenvolvido especificamente para lidar com o contexto dinâmico dos fluxos de dados, essa medida penaliza situações em que ocorrem:

- **perda de dados**: devido ao movimento dos grupos, dados podem ser perdidos.
- **dados mal situados**: a sobreposição de grupos decorrente da evolução, união e divisão dos grupos pode gerar dados mal situados.

- **ruídos**: inclusão de ruídos nos grupos encontrados.

O CMM varia no intervalo [0,1], resultando em 1 caso não ocorra nenhuma falha e 0 para erro máximo:

$$CMM = 1 - \frac{\sum_{o \in F} w(o) \times pen(o, C)}{\sum_{o \in F} w(o) \times con(o, Cl(o))}, \tag{10}$$

se $F = \emptyset$, então $CMM = 1$, ademais:

- $w(o)$: peso de o, definido de acordo com o instante da chegada da instância
- $pen(o, C)$: penalidades gerais para pontos de falha
- $con(o, Cl_i(o))$: conectividade do ponto o para o grupo Cl_i
- F : conjunto erro, isto é, conjunto de objetos mapeados para a classe incorreta
- o : uma única instância de um fluxo S formado por $\{o^1, o^2, ..., o^n\}$ instâncias
- C : conjunto dos grupos
- CL : ground-truth do agrupamento

3.3.5. Coeficiente de Silhueta

O cálculo do coeficiente de silhueta para cada dado é definido pela dissimilaridade média aos dados do seu grupo em relação à dissimilaridade média aos dados de outros grupos [Kaufman and Rousseeuw 1990]. Essa medida varia no intervalo $[-1, 1]$, no qual seu resultado é normalizado para $[0, 1]$:

$$Coeficiente\,de\,Silhueta = \frac{1}{N} \sum_{i=1}^{N} \frac{b(i) - a(i)}{\max\{b(i), a(i)\}}, \tag{11}$$

em que N é a quantidade de objetos, $a(i)$ é a distância média do i-ésimo objeto a todos os objetos do seu grupo e $b(i)$, a distância média do i-ésimo objeto a todos os objetos do seu grupo vizinho mais próximo.

3.4. Bases de dados

Para retratar a natureza específica dos fluxos de dados, foram selecionadas três bases de dados artificiais, em que as instâncias foram produzidas através do Gerador Aleatório de Função de Base Radial (*Random RBF Generator*) disponível na plataforma MOA: ArtifMC, ArtifMCE e ArtifMCER.

- **ArtifMC - artificial com mudança de conceito**: contém 55.000 instâncias e tem como objetivo simular mudanças de conceito presentes em um fluxo de dados. No gerador artificial, mudanças de conceitos são introduzidas movendo os centroides com velocidade constante, nesse caso, com uma distância de 0.01 a cada 500 exemplos.
- **ArtifMCE - artificial com mudança de conceito e eventos**: possui um total de 125.000 instâncias e, além de mudanças de conceitos, incorpora eventos de aparecimento/desaparecimento de classes, que ocorrem a cada 25.000 instâncias ao longo do fluxo.

- **ArtifMCER - artificial com mudança de conceito, eventos e ruídos**: possui um total de 125.000 instâncias e se apresenta como uma combinação das bases ArtifMC e ArtifMCE, com acréscimo da presença de ruídos no decorrer do fluxo.

***Random RBF Generator*:** Inicialmente, um número fixo de centróides é gerado, cada um possuindo uma posição aleatória, um desvio padrão, rótulo de classe e peso. Novas instâncias do fluxo são produzidas deslocando-se valores dos atributos de pontos centrais escolhidos ao acaso. Assim, esse gerador cria uma hiperesfera com distribuição normal de instâncias ao redor de cada ponto central, com densidades que variam. É importante ressaltar que apenas atributos numéricos no intervalo $[0, 1]$ são gerados, e a seleção aleatória do centróide também determina a que classe a instância pertence.

Tabela 1. Resumo das bases de dados

Bases de dados	Instâncias	Atributos	Classes	Mud. de conceito	Eventos	Ruídos
ArtifMC	55.000	2	5	✓		
ArtifMCE	125.000	2	8	✓	✓	
ArtifMCER	125.000	2	8	✓	✓	✓

4. Resultados

O desempenho dos algoritmos na fase offline do CluStream foi avaliado sob diferentes valores dos principais parâmetros das técnicas. O k do $GkNN$, tanto do modelo simétrico quanto do mútuo, foi variado no intervalo de 1 a 20. Já para o k-Means, o seu parâmetro k, que determina a quantidade de grupos finais, foi definido entre 2 e 20. Os métodos Fastgreedy e Louvain permaneceram com o padrão estabelecido no igraph.

Nesse sentido, para analisar conjuntamente os resultados das medidas de avaliação, definiu-se um valor ideal de parâmetro para cada técnica. Em outras palavras, foi determinado um respectivo valor de k para cada técnica que apresente bons resultados em todas as medidas ao mesmo tempo. Esse valor ideal é obtido a partir daquele no qual a média harmônica entre a medida F, CMM e coeficiente de silhueta seja maior.

É importante destacar que os centróides iniciais do k-Means são determinados de forma aleatória, assim como a ordem em que os vértices são considerados na fase de otimização da modularidade do Louvain. Dessa forma, os resultados obtidos por essas duas técnicas, em cada base de dados analisada, foram obtidos através do cálculo da média de 10 execuções.

Ainda, nota-se que foram mantidos os parâmetros predefinidos do CluStream no MOA, em que o fluxo dos dados foi dividido em janelas de tamanho igual a 1000, ou seja, a fase offline, na qual os micro-grupos são agrupados e as medidas de desempenho são calculadas, foi executada a cada 1000 objetos processados. Diante disso, para cada critério de avaliação analisado, os resultados são apresentados como a média dos valores obtidos nas janelas no decorrer do fluxo de dados.

4.1. Base de dados ArtifMC

Para realizar os experimentos, as instâncias da base de dados ArtifMC foram produzidas de acordo com os seguintes valores dos parâmetros do *Random RBF Generator Events*, em que os demais parâmetros permaneceram com a sua definição padrão:

- instanceRandomSeed: 1
- noiseLevel: 0

A Tabela 2 mostra o desempenho dos algoritmos na base de dados ArtifMC. Nesse caso, em cada técnica foram definidos os seguintes valores ideais para o correspondente parâmetro k:

- Fastgreedy com $SGkNN$ e $MGkNN$: 5 e 6, respectivamente.
- Louvain com $SGkNN$ e $MGkNN$: 8 e 12, respectivamente.
- k-Means: 6

A partir dos valores apresentados na Tabela 2, é possível perceber que o Louvain e Fastgreedy, tanto para versão simétrica quanto mútua do $GkNN$, obtiveram para todas as medidas, melhores resultados que os do k-Means. Ainda em vista da Tabela 2, realizou-se uma comparação entre o Louvain com $SGkNN$ e o k-Means, a fim de se observar o comportamento dessas técnicas em cada janela durante todo o fluxo dos dados. O Louvain com $GkNN$ simétrico foi escolhido por exibir um desempenho superior entre os métodos de detecção de comunidades. Dessa forma, na Figura 1, observa-se que para cada medida, o Louvain alcança majoritariamente uma performance melhor que a do k-Means.

Tabela 2. Comparação entre os resultados obtidos pelos algoritmos de detecção de comunidades (variando o $GkNN$ em simétrico e mútuo) e o k-Means na base de dados ArtifMC após a definição do valor ideal de parâmetro para as técnicas

	Fastgreedy		Louvain		k-Means
Medidas de Avaliação	$SGkNN$	$MGkNN$	$SGkNN$	$MGkNN$	
Pureza	0.979	0.984	**0.987**	**0.987**	0.964
Pureza Inversa	0.989	0.985	**0.991**	**0.991**	0.899
Medida F	0.984	0.984	**0.989**	0.988	0.930
CMM	**0.982**	**0.982**	0.981	0.981	0.944
Coeficiente de Silhueta	0.941	0.953	**0.959**	0.958	0.840

4.2. Base de dados ArtifMCE

Para a execução dos testes, as instâncias da base de dados ArtifMCE foram produzidas de acordo com a definição dos seguintes valores dos parâmetros do gerador aleatório, permanecendo os demais sem modificação:

- instanceRandomSeed: 3
- noiseLevel: 0
- eventFrequency: 25000
- eventMergeSplit: habilitado
- eventDeleteCreate: habilitado

A Tabela 3 apresenta o desempenho das técnicas na base de dados ArtifMCE. Em cada método, o valor ideal do correspondente parâmetro k foi determinado como segue:

- Fastgreedy com $SGkNN$ e $MGkNN$: 4 e 6, respectivamente.
- Louvain com $SGkNN$ e $MGkNN$: 8 e 9, respectivamente.

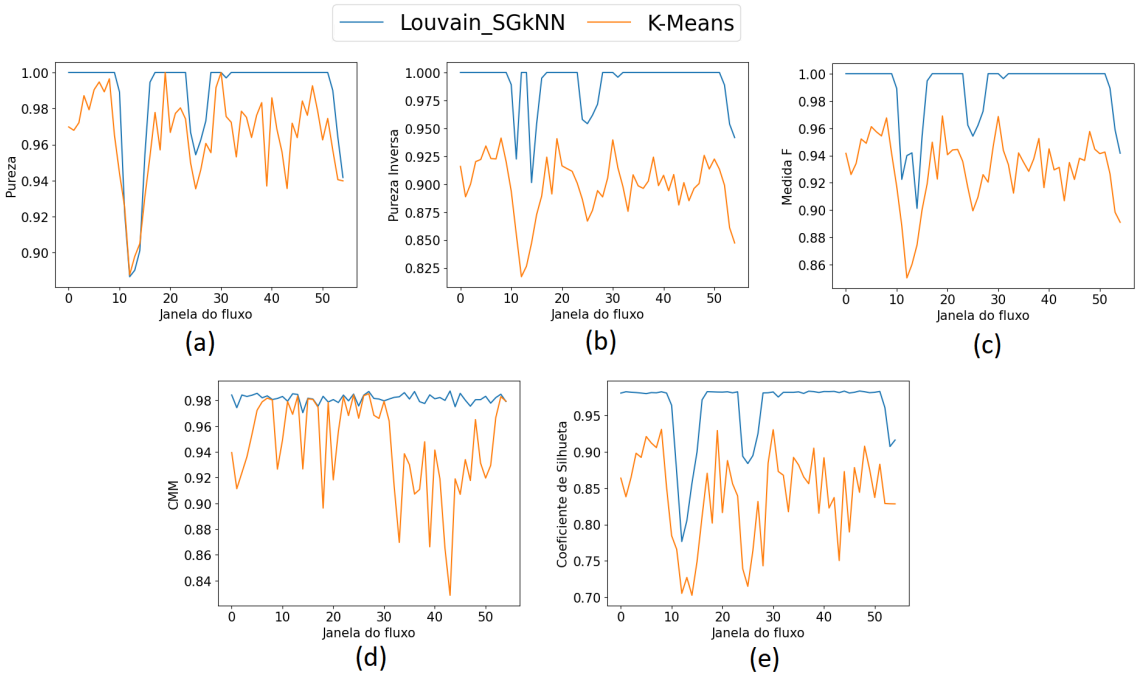

Figura 1. Resultados obtidos pelo Louvain e k-Means em cada janela da base ArtifMC: (a) Pureza, (b) Pureza Inversa, (c) Medida F, (d) CMM e (e) Coeficiente de Silhueta

- k-Means: 7.

Ao analisar a Tabela 3, nota-se que as técnicas de detecção de comunidades, sob as diferentes versões do $GkNN$, obtêm melhores resultados que os do k-Means, para todos os critérios de avaliação. Também com base na Tabela 3, uma vez que o Louvain com $GkNN$ simétrico é melhor que a sua versão mútua e que ao Fastgreedy, foi realizada uma comparação entre sua performance e a do k-Means em cada janela no decorrer do fluxo de dados. Assim, o comportamento dos métodos na Figura 2 demonstra que o Louvain, na maioria dos casos, apresenta um desempenho superior ao do k-Means, alcançando muitas vezes o valor máximo das medidas.

Tabela 3. Comparação entre os resultados obtidos pelos algoritmos de detecção de comunidades (variando o $GkNN$ em simétrico e mútuo) e o k-Means na base de dados ArtifMCE após a definição do valor ideal de parâmetro para as técnicas

	Fastgreedy		Louvain		k-Means
Medidas de Avaliação	$SGkNN$	$MGkNN$	$SGkNN$	$MGkNN$	
Pureza	0.960	0.964	**0.970**	0.969	0.948
Pureza Inversa	0.976	0.983	**0.991**	0.987	0.880
Medida F	0.967	0.973	**0.980**	0.977	0.911
CMM	0.979	0.980	**0.981**	**0.981**	0.944
Coeficiente de Silhueta	0.889	0.902	**0.918**	0.914	0.783

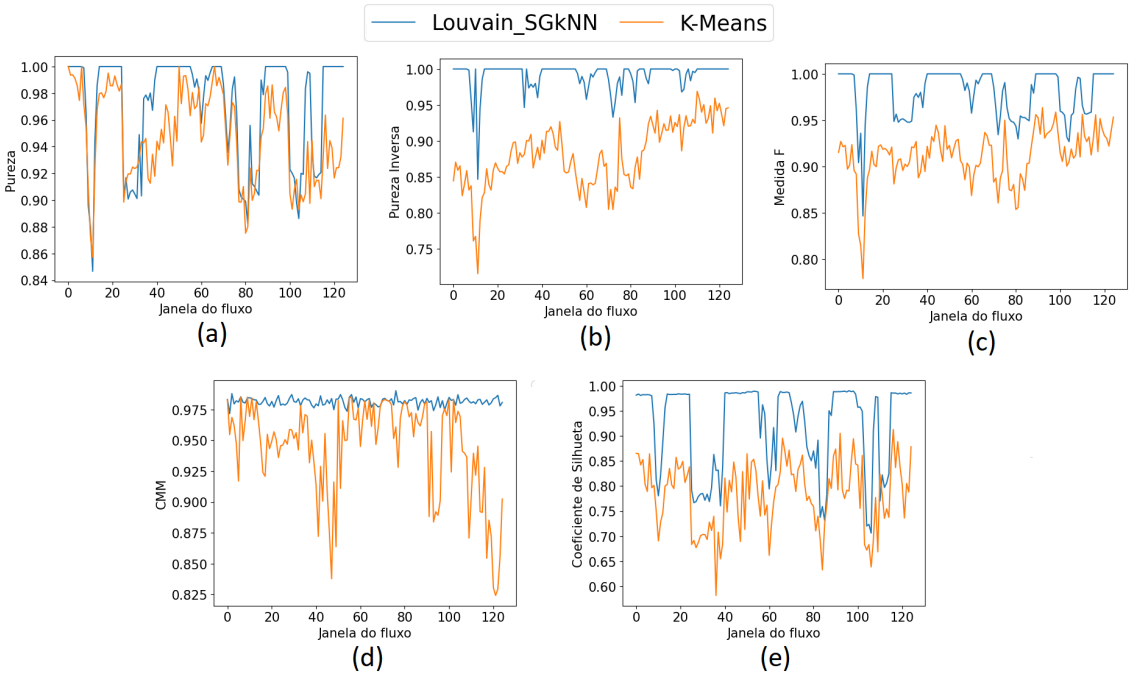

Figura 2. Resultados obtidos pelo Louvain e k-Means em cada janela da base ArtifMCE: (a) Pureza, (b) Pureza Inversa, (c) Medida F, (d) CMM e (e) Coeficiente de Silhueta

4.3. Base de dados ArtifMCER

Para executar os experimentos e simular o comportamento desejado para a base de dados ArtifMCER, suas intâncias foram geradas definindo-se os determinados valores para os seguintes parâmetros do gerador, mantendo os demais inalterados:

- instanceRandomSeed: 3
- noiseLevel: 0.1
- eventFrequency: 25000
- eventMergeSplit: habilitado
- eventDeleteCreate: habilitado

A Tabela 4 exibe o desempenho das técnicas na base de dados ArtifMCER. Para cada método, o valor ideal do correspondente parâmetro k foi determinado conforme:

- Fastgreedy com $SGkNN$ e $MGkNN$: 3 e 4, respectivamente.
- Louvain com $SGkNN$ e $MGkNN$: 7 e 14, respectivamente.
- k-Means: 10

Em relação à Tabela 4, observa-se que o Louvain com o $SGkNN$ tem resultados superiores aos do k-Means, por outro lado, sua versão mútua revela um comportamento contrário. O Fastgreedy também obtém um desempenho melhor que o k-Means somente com o $GkNN$ simétrico, nesse caso, para pureza, CMM e coeficiente de silhueta. Ainda considerando a Tabela 4, e partindo do fato do Louvain com $SGkNN$ se destacar entre os métodos de detecção de comunidades, foi feita uma comparação entre seus resultados e os obtidos pelo k-Means ao longo das janelas do fluxo de dados

A Figura 3 aponta que o Louvain se mostra, no geral, superior ao k-Means para todas as medidas. Além disso, é possível perceber que a presença de ruídos impacta

bastante na performance, já que ambas as técnicas não são eficientes em lidar com esse tipo de dado. Inclusive, diferentemente das outras análises, em alguns casos como a pureza e o coeficiente de silhueta, os resultados do Louvain se aproximam muito aos do k-Means.

Tabela 4. Comparação entre os resultados obtidos pelos algoritmos de detecção de comunidades (variando o $GkNN$ em simétrico e mútuo) e o k-Means na base de dados ArtifMCER após a definição do valor ideal de parâmetro das técnicas

	Fastgreedy		Louvain		K-Means
Medidas de Avaliação	$SGkNN$	$MGkNN$	$SGkNN$	$MGkNN$	
Pureza	0.889	0.855	**0.905**	0.851	0.879
Pureza Inversa	0.917	0.870	**0.972**	0.932	0.934
Medida F	0.899	0.859	**0.935**	0.884	0.901
CMM	0.953	0.918	**0.965**	0.923	0.936
Coeficiente de Silhueta	0.833	0.797	**0.851**	0.792	0.810

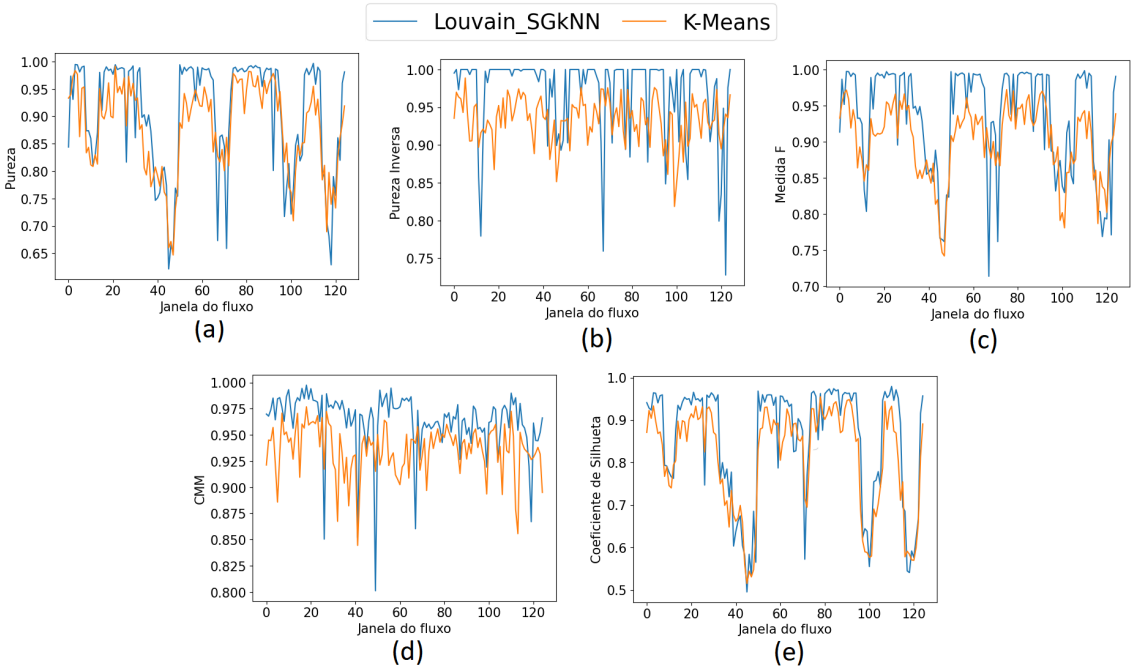

Figura 3. Resultados obtidos pelo Louvain e k-Means em cada janela da base ArtifMCER: (a) Pureza, (b) Pureza Inversa, (c) Medida F, (d) CMM e (e) Coeficiente de Silhueta

5. Conclusão

Nesse trabalho de pesquisa, modelos de aprendizado baseados em redes complexas foram incorporados à etapa offline do CluStream. Dessa forma, os algoritmos de detecção de comunidades Louvain e Fastgreedy foram utilizados ao invés do método tradicional k-Means. Para a execução dos experimentos, foram produzidas três bases de dados com

diferentes propriedades: mudança de conceito, aparecimento e/ou desaparecimento de classes e a presença de ruídos. Além disso, utilizou-se o grafo $GkNN$ (simétrico e mútuo) na formação da rede a partir dos micro-grupos formados no estágio online do CluStream.

Em relação às bases de dados, para aquelas em que não há ruídos, o Fastgreedy e Louvain juntos alcançaram, para todas as medidas de avaliação, um desempenho superior ao obtido pelo k-Means, considerando as duas versões do $GkNN$. Já quando a base contém ruídos, ainda assim, pelo menos um dos métodos de detecção de comunidade tem uma performance melhor que a do k-Means em cada critério de avaliação.

Em suma, conclui-se que a abordagem de aprendizado usando redes complexas se mostrou relevante, mesmo considerando problemas em que as instâncias das bases de dados não se encontram originalmente em formato de grafo. Sendo assim, a pesquisa evidencia que dados na forma de vetor de atributos podem ser beneficiados pela análise em rede e alcançar bons resultados através do processo de detecção de comunidades.

Agradecimentos

Os autores agradecem o suporte financeiro do Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (processo 439556/2018-0). DAN também agradece ao CNPq por sua bolsa de estudos.

Referências

Aggarwal, C. C., Han, J., Wang, J., and Yu, P. S. (2003). A framework for clustering evolving data streams. In *Proceedings of the 29th international conference on Very large data bases - Volume 29*, VLDB '03, pages 81–92, Berlin, Germany. VLDB Endowment.

Amigó, E., Gonzalo, J., Artiles, J., and Verdejo, M. (2009). A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12:461–486.

Barabási, A.-L. and Pósfai, M. (2016). *Network science*. Cambridge University Press, Cambridge, United Kingdom. OCLC: ocn910772793.

Berkhin, P. (2006). Survey of Clustering Data Mining Techniques. page 56.

Bifet, A., Holmes, G., Kirkby, R., and Pfahringer, B. (2010). MOA: massive online analysis. *J. Mach. Learn. Res.*, 11:1601–1604.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. 2008(10):P10008. arXiv: 0803.0476.

Bornholdt, S. and Schuster, H. G., editors (2003). *Handbook of graphs and networks: from the genome to the internet*. Wiley-VCH, Weinheim, 1st ed edition. OCLC: ocm50056112.

Carneiro, M. G., Cheng, R., Zhao, L., and Jin, Y. (2019). Particle swarm optimization for network-based data classification. *Neural Networks*, 110:243–255.

Carneiro, M. G., Gama, B. C., and Ribeiro, O. S. (2021). Complex network measures for data classification. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Carneiro, M. G. and Zhao, L. (2018). Analysis of graph construction methods in supervised data classification. In *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 390–395. IEEE.

Clauset, A., Newman, M. E. J., and Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, 70(6):066111. arXiv:cond-mat/0408187.

Csardi, G. and Nepusz, T. (2005). The Igraph Software Package for Complex Network Research. *InterJournal*, Complex Systems:1695.

Diestel, R. (2005). *Graph theory*. Number 173 in Graduate texts in mathematics. Springer, Berlin ; New York, 3rd ed edition.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486(3-5):75–174.

Freitas, L. M. and Carneiro, M. G. (2019). Community detection to invariant pattern clustering in images. In *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 610–615.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4):1–37.

Gantz, J. and Reinsel, D. (2011). Extracting Value from Chaos. page 12.

Kaufman, L. and Rousseeuw, P. (1990). *Finding Groups in Data: An Introduction To Cluster Analysis*.

Kremer, H., Kranen, P., Jansen, T., Seidl, T., Bifet, A., Holmes, G., and Pfahringer, B. (2011). An effective evaluation measure for clustering on evolving data streams. pages 868–876.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill Science/Engineering/Math, New York, 1ª edição edition.

Newman, M. E. J. (2003). The Structure and Function of Complex Networks. *SIAM Review*, 45(2):167–256.

Newman, M. E. J. (2004). Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6):066133. arXiv: cond-mat/0309508.

Nguyen, H.-L., Woon, Y.-K., and Ng, W. K. (2014). A Survey on Data Stream Clustering and Classification. *Knowledge and Information Systems*, 45.

Zhang, T., Ramakrishnan, R., and Livny, M. (1996). BIRCH: an efficient data clustering method for very large databases. *ACM SIGMOD Record*, 25(2):103–114.