UNIVERSIDADE FEDERAL DE MINAS GERAIS
Escola de Engenharia
Programa de Pós-Graduação em Engenharia Elétrica

Nayron Morais Almeida

Sistemas Inteligentes Evolutivos: novas abordagens para detecção e classificação de eventos

Belo Horizonte

2025

Nayron Morais Almeida

Sistemas Inteligentes Evolutivos: novas abordagens para detecção e classificação de eventos

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Doutor em Engenharia Elétrica.

Orientador: Walmir Matos Caminhas

Belo Horizonte
2025

A447s

Almeida, Nayron Morais.
Sistemas inteligentes evolutivos [recurso eletrônico] : novas abordagens para detecção e classificação de eventos / Nayron Morais Almeida. - 2025.
1 recurso online (124 f. : il., color.) : pdf.

Orientador: Walmir Matos Caminhas.

Tese (doutorado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Inclui bibliografia.

1. Engenharia elétrica - Teses. 2. Aprendizado do computador - Teses. 3. Processamento eletrônico de dados - Teses. 4. Sistemas especialistas (Computação) - Teses. I. Caminhas, Walmir Matos. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 621.3(043)

Ficha catalográfica elaborada pelo bibliotecário Reginaldo César Vital dos Santos CRB/6 2165
Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS

Escola de Engenharia

COLEGIADO DO CURSO DE GRADUAÇÃO / PÓS-GRADUAÇÃO EM Engenharia Elétrica

FOLHA DE APROVAÇÃO

"Sistemas Inteligentes Evolutivos: Novas Abordagens Para Detecção e Classificação de Eventos"

Nayron Morais Almeida

Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Doutor em Engenharia Elétrica.

Aprovada em 25 de abril de 2025.

Por:

Prof. Dr. Walmir Matos Caminhas
DELT (UFMG) - Orientador

Prof. Dr. Frederico Gadelha Guimarães
DCC (UFMG)

Prof. Dr. Luiz Affonso Henderson Guedes de Oliveira
DCA (UFRN)

Prof. Dr. Daniel Furtado Leite
DCC (University of Paderborn)

Prof. Dr. Anderson da Silva Soares
Instituto de Informática (UFG)

Prof. Dr. Marcos Flávio S. V. D'Angelo
DCC (Unimontes)

Documento assinado eletronicamente por **Walmir Matos Caminhas**, **Presidente de comissão**, em 25/04/2025, às 17:37, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Frederico Gadelha Guimaraes**, **Professor do Magistério Superior**, em 26/04/2025, às 09:53, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Marcos Flávio Silveira Vasconcelos DAngelo**, **Usuário Externo**, em 06/05/2025, às 14:36, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Daniel Furtado Leite**, **Usuário Externo**, em 06/05/2025, às 19:57, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Anderson da Silva Soares**, **Usuário Externo**, em 06/05/2025, às 23:15, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Luiz Affonso Guedes**, **Usuário Externo**, em 07/05/2025, às 13:41, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **4157576** e o código CRC **6DBFA840**.

Referência: Processo nº 23072.223488/2025-60 SEI nº 4157576

Agradecimentos

A jornada até a conclusão deste trabalho foi repleta de desafios, aprendizados e, acima de tudo, apoio inestimável de pessoas especiais.

Agradeço primeiramente ao Prof. Walmir Matos Caminhas pelo suporte, compreensão e incentivo contínuo, fundamentais para que este trabalho se tornasse realidade. Ao Prof. Reinaldo Palhares, Prof. Carlos Bomfim e Murilo pela grande ajuda fornecida ao longo do desenvolvimento dessa pesquisa.

À minha família, por todo amor, paciência e compreensão durante essa caminhada. Seu apoio incondicional foi essencial nos momentos mais difíceis.

Aos colegas e amigos, pelas trocas de conhecimento, discussões enriquecedoras e pelo suporte mútuo ao longo dessa trajetória.

Aos meus professores e amigos da UFMG, pelas valiosas orientações e pela inspiração constante para seguir sempre em busca do conhecimento, e também pelos momentos de alegrias.

Por fim, sou grato(a) a todos que, de alguma forma, contribuíram para este trabalho. Sem vocês, esta conquista não teria o mesmo significado.

Muito obrigado!

"Não são os anos de vida que contam, mas a vida nos anos"

Abraham Lincoln

Resumo

O aprendizado de máquina evolutivo, particularmente quando baseado em fluxo de dados, tem se mostrado uma ferramenta de grande relevância em diversas áreas. Essa abordagem permite que modelos computacionais de sistemas físicos incorporem padrões de comportamento observados nos fluxos de dados, adotando uma perspectiva evolutiva. Dessa forma, os modelos sujeitos à evolução são capazes de iniciar com pouco ou nenhum conhecimento prévio do domínio de aplicação e, por meio do monitoramento contínuo, detectar, internalizar e refinar novos padrões e informações ao longo do tempo. Diante da importância desse tipo de aprendizado, a literatura de sistemas inteligentes apresenta uma variedade de abordagens voltadas ao processamento contínuo de dados ao longo da vida útil dos sistemas. No entanto, ainda há desafios significativos a serem superados, especialmente no que diz respeito à perda de desempenho em função da crescente complexidade dos sistemas modernos. Nesse contexto, este trabalho tem como objetivo propor novas abordagens evolutivas e formulações para a tarefa de classificação de eventos em fluxos de dados, com foco em técnicas de clusterização e avaliação de resíduos de aproximação. Foram desenvolvidas três novas abordagens, baseadas em quantização vetorial e avaliação de resíduos, que se destacam por seu desempenho competitivo, simplicidade e velocidade, graças a procedimentos de aprendizado eficientes e de baixa complexidade. Para validar as propostas, foram conduzidos experimentos em três *benchmarks* públicos da área de processos industriais, com o intuito de avaliar o desempenho e comparar os resultados com os métodos mais avançados disponíveis na literatura. Os resultados obtidos demonstram que, de maneira geral, as abordagens propostas são competitivas em relação às técnicas de referência, apresentando um equilíbrio entre precisão, eficiência e praticidade. Em síntese, este trabalho contribui para o avanço do aprendizado de máquina evolutivo, oferecendo soluções inovadoras e promissoras para o processamento de fluxos de dados em cenários complexos e dinâmicos.

Palavras-chave: sistemas inteligentes evolutivos. fluxo de dados. aprendizado de máquina.

Abstract

Evolving Machine Learning, particularly when based on data streams, has proven to be a highly relevant tool in various fields. This approach enables computational models of physical systems to incorporate behavioral patterns observed in data streams, adopting an evolving perspective. Thus, models subject to evolution can start with little or no prior knowledge of the application domain and, through continuous monitoring, detect, internalize, and refine new patterns and information over time. Given the importance of this type of learning, the intelligent systems literature presents a variety of approaches aimed at the continuous processing of data throughout the system's lifespan. However, there are still significant challenges to overcome, especially regarding performance degradation due to the increasing complexity of modern systems. In this context, this work aims to propose new evolving approaches and formulations for the task of event classification in data streams, focusing on clustering techniques and approximation residual evaluation. Three new approaches were developed, based on vector quantization and residual evaluation, which stand out for their competitive performance, simplicity, and speed, thanks to efficient and low-complexity learning procedures. To validate the proposals, experiments were conducted on three public benchmarks in the field of industrial processes, aiming to assess performance and compare the results with the most advanced methods available in the literature. The results demonstrate that, in general, the proposed approaches are competitive with reference techniques, presenting a balance between accuracy, efficiency, and practicality. In summary, this work contributes to the advancement of evolving machine learning, offering innovative and promising solutions for data stream processing in complex and dynamic scenarios.

Keywords: evolving intelligent systems. data streams. machine learning.

Lista de Ilustrações

Lista de Tabelas

Lista de Algoritmos

Sumário

Capítulo 1

Introdução

A capacidade de detectar e incorporar novos eventos a sua base de conhecimento é uma habilidade importante para qualquer modelo computacional inteligente baseado em dados. Muito frequentemente, é impossível treinar um modelo computacional com todos os possíveis padrões de dados [Markou and Singh, 2003]. Veículos autônomos [Wiseman, 2022] que navegam em ambiente desconhecido e incerto são um exemplo dessa situação. Esses tipos de veículos necessitam de sistemas inteligentes altamente especializados e ao mesmo tempo adaptáveis e que garantam níveis mínimos de segurança, até mesmo em situações não observadas previamente.

Muito além de veículos autônomos, sistemas com essas propriedades são úteis e necessários nas mais diversas áreas. Por exemplo, Sistemas de Recomendação necessitam dessa habilidade para acompanhar a dinâmica de interesses dos usuários ao longo do tempo e sugerir opções ou serviços que são mais verossímeis aos seus interesses atuais. No ambiente hospitalar, por exemplo, essa habilidade embutida em softwares é vital no monitoramento do quadro clínico dos pacientes, já que podem ser bem particulares (medicina personalizada). Em controle de tráfego rodoviário e aéreo, sistemas inteligentes são importantes para acompanhar as necessidades de fluxo ao longo do dia, semana, mês ou ano. Importantes decisões logísticas podem ser tomadas com base na informação capturada por modelos inteligentes. No ambiente industrial, se faz necessário tais propriedades de aprendizado para modelar incrementalmente a dinâmica dos processos, visto que dada sua alta complexidade e variância temporal é impossível conhecer todos os possíveis modos operacionais de máquinas e robôs previamente. Por fim, é possível concluir que existe um vasto campo de aplicações para sistemas inteligentes com propriedades evolutivas.

Na literatura o termo *Detecção de Novidade* é comumente utilizado para designar a tarefa de classificar novas entradas de dados em *Conhecida* ou *Desconhecida* tomando como referência limiares definidos para atributos durante o treinamento[Pimentel et al., 2014]. Em outras palavras, a questão é tratada como um problema de classificação binária, em que se faz necessário retreinar o sistema sempre que houver a necessidade de incorporar

novos padrões de dados ao que é dito ser *usual*.

Uma segunda perspectiva, que tem sido adotada especialmente na última década, é o chamado aprendizado evolutivo ou aprendizado online [Angelov et al., 2010]. Essa visão trata de forma mais apropriada as demandas da classe dos problemas exemplificados. A mesma assume que o sistema está sempre *incompleto*, e, como tal, necessita permanecer em um estado constante de aprendizado (ou treinamento), logo, seu processo de inferência sempre traz consigo o prefixo "a partir do que observei até o momento...", tornando explícito que sua resposta é consequência do histórico observado, e que diante de novas evidências, pode alterar sua "opinião". Além disso, nessa perspectiva, não há reapresentação de dados. Cada informação é apresentada ao sistema uma única vez, sendo de responsabilidade do mesmo efetuar os devidos procedimentos para garantir que o conhecimento associado a informação seja internalizado. Na literatura, essa classe de sistema é chamada de Sistemas Evolutivos ou de Métodos com Aprendizado Evolutivo.

Muito embora a comunidade científica da área venha dedicando-se de forma efetiva a pelo menos duas décadas no desenvolvimento de novos sistemas evolutivos, percebe-se que ainda existe muito a ser elaborado, especialmente em relação aos desafios que geralmente as aplicações impõem, tais como [Angelov et al., 2010, Škrjanc et al., 2019, Batool and Abbas, 2021, Al-Amri et al., 2021]:

D1) Acompanhar a dinâmica dos dados: Os métodos precisam considerar que a dinâmica dos dados está em constante mudança. Consequentemente, o método deve ser capaz de detectar novos padrões de dados, extrair o conhecimento associado, e, em paralelo, manter um nível mínimo de conhecimento do passado.

D2) Gerenciar falsas novidades: Dependendo da formulação do método, falsas novidades, formalmente conhecida por *outliers*, podem acarretar em um grande impacto na estrutura interna do modelo associado, e, consequentemente, prejudicar seu desempenho. Dessa forma, na tentativa de manter a estrutura íntegra e reduzir falsos negativos e falsos positivos (erros do Tipo I e II, [Bishop and Nasrabadi, 2006]), *outliers* devem ser tratados adequadamente.

D3) Ser escalável em dimensionalidade: Devido a alta complexidade de algumas aplicações, ter de lidar com dados de alta dimensionalidade é algo inevitável, o que leva a chamada maldição da dimensionalidade [Bishop and Nasrabadi, 2006]. Logo, os métodos devem idealmente ser escaláveis, de forma a utilizar mecanismos que preservem sua performance em aplicações com alta dimensionalidade.

D4) Responder em tempo-real: O tempo máximo de resposta dependerá de cada aplicação, do quão crítica é. Em geral, costuma-se operar no máximo na escala de milissegundos.

D5) Operar com pouca memória: Desde que não há um tempo predeterminado de execução, em que teoricamente os dados são infinitos, os métodos devem considerar que não há memória suficiente para armazenar todo dado apresentado. Logo, operar com pouca memória é um requisito indispensável.

D6) Gerenciar complexidade computacional: Dada a necessidade de processamento em tempo-real (ou próximo disso), os métodos devem gerenciar seu custo computacional em função do crescimento da dimensionalidade e expansão de sua estrutura interna. O número de parâmetros de modelos e/ou quantidade de modelos locais devem ser minimizados.

Este trabalho tem como objetivo contribuir para o desenvolvimento de novas abordagens evolutivas capazes de enfrentar os desafios mencionados, oferecendo soluções mais robustas a ruídos e *outliers*, adaptativas e potencialmente escaláveis para a detecção e classificação de eventos em processos não estacionários. Para isso, propõe-se o desenvolvimento de abordagens modernas que integram aprendizado incremental baseado em quantização vetorial, modelagem granular e análise de resíduos de reconstrução. Essas abordagens permitirão não apenas a detecção precoce de eventos, mas também a construção contínua de uma base de conhecimento, tornando o suporte à tomada de decisão mais eficiente e fundamentado no domínio da aplicação. Dessa forma, espera-se que as soluções desenvolvidas sejam aplicáveis a uma ampla variedade de processos, independentemente do domínio específico, promovendo avanços significativos na área de monitoramento inteligente de sistemas complexos.

1.1 Objetivos

O foco deste trabalho está no desenvolvimento e aplicação de novas abordagens na modelagem evolutiva de eventos em fluxos de dados. Para tanto, é possível destrinchá-lo nas seguintes metas:

- Revisar o estado da arte na literatura de métodos evolutivos direcionados a ambientes não-estacionários e baseados em clusterização e autoencoders;
- Desenvolver novas abordagens evolutivas baseadas em quantização vetorial e autoecoders;
- Aplicar as abordagens desenvolvidas em problemas *benchmark* e sistemas dinâmicos reais;
- Comparar as abordagens desenvolvidas com aquelas que compõem o estado-da-arte utilizando índices de validação apropriados.

1.2 Publicações

Esta seção apresenta as publicações produzidas no decorrer desta pesquisa e as que ainda estão em desenvolvimento, a saber:

1. ALMEIDA, N. M.; CAMARGOS, M. O.; MARIANO, D. G. B.; BOMFIM, C. H. M.; PALHARES, R. M.; CAMINHAS, W. M. **An Evolving Approach to the Similarity-Based Modeling for Online Clustering in Non-Stationary Environments**. Evolving Systems, v. 16, n. 1, p. 16, 2024. (Publicado).
2. ALMEIDA, N. M.; CAMINHAS, W. M. **An Evolving Granular Autoencoder for Incremental Learning of Latent Representations**. (Em desenvolvimento).
3. ALMEIDA, N. M.; CAMINHAS, W. M. **An Evolving Framework for Process Monitoring Using Residual Information**. (Em desenvolvimento).

1.3 Organização do Trabalho

Este trabalho está constituído de seis capítulos. Neste capítulo, foi introduzido o tema e a problemática considerada para o desenvolvimento da presente pesquisa. No Capítulo 2 é apresentado os fundamentos teóricos da detecção e classificação de eventos sob a ótica probabilística e sob a perspectiva evolutiva, bem como a proposta de uma arquitetura geral para modelagem evolutiva de eventos. O Capítulo 3 é dedicado a apresentação de AutoEncoder, uma arquitetura de rede neural que busca aproximar os dados por meio de sua transformação para um novo espaço de dados, chamado *espaço latente*; bem como de proposições de novas abordagens que compartilham dessa metodologia. Em geral o problema de detecção e classificação de eventos em sistemas é tratado sob um esquema de aprendizado não-supervisionado, especialmente clusterização. No Capítulo 4 é apresentado o tema, algumas das principais abordagens que compõem o estado da arte, e, além disso, também é apresentada uma nova abordagem baseada em similaridade de modelos. No Capítulo 5 são apresentados os resultados da aplicação das abordagens propostas, eGAE, eRBM, e eSBM4Clus, comparando-as com algumas outras do estado da arte. Por fim, no Capítulo 6, são apresentadas as conclusões e propostas de trabalhos futuros da pesquisa.

Capítulo 2

Detecção e Classificação de Eventos

Com o crescimento expressivo da complexidade dos sistemas modernos, conhecer previamente todas as possíveis situações a que o sistema está suscetível é algo praticamente impossível. A abordagem mais viável é catalogá-las à medida que surgem, e idealmente incorporar os procedimentos técnicos que devem ser executados para normalizar o sistema no caso da situação representar uma falha operacional.

Por exemplo, considere um processo industrial qualquer que vem operando normalmente, e em um dado momento ocorre um evento de falha pela primeira vez. Após algum tempo desde seu surgimento, a equipe técnica o percebe e observa que está evoluindo lentamente para um estágio mais grave. A falha inicial apresentará grande risco de segurança ao ambiente quando evoluir a níveis próximos à falha funcional. Após um intervalo de tempo aceitável, a equipe interrompe o processo para corrigir o problema. Realiza-se o procedimento de documentar sua origem bem como as ações tomadas. A reincidência da falha agora representa um menor risco em comparação a primeira ocorrência, visto que agora se sabe onde e como atuar para que o processo retorne à normalidade.

Este exemplo ilustra bem o fluxo ideal de aprendizado adotado nesta tese. Nessa perspectiva, considera-se que o modelo computacional possui pouco ou nenhum conhecimento da dinâmica do sistema, e que a partir de seu monitoramento contínuo consegue detectar e incorporar (aprender) os eventos a sua base de conhecimento. Logo, assim como no exemplo anterior, *evento* dentro dessa perspectiva representa um estado físico do sistema, ou estágios dele. Assim, recorrências de estados do sistema são facilmente detectados e identificados, pois o método já possui sua representação internamente.

O restante deste capítulo aprofunda o tema introduzido, descrevendo os fundamentos comuns aos principais métodos que compõem o estado da arte na área, sob uma perspectiva probabilística e evolutiva. Ressalta-se que, ao longo desta tese, os termos *fluxo de dados*, *sistema dinâmico*, *sistema não estacionário* e, de forma mais geral, *sistema* serão utilizados de maneira intercambiável.

2.1 Detecção e Classificação de Eventos sob uma Perspectiva Probabilística

Em geral, a detecção de eventos sob a ótica probabilística está associada ao fenômeno conhecido como *mudança de conceito* (no inglês, *concept drift*) [Gama et al., 2014, Ditzler et al., 2015, Lu et al., 2019], frequentemente tratado como um problema de aprendizado supervisionado. Mais especificamente, considere um método previamente treinado para modelar um determinado sistema. Quando ocorre uma mudança de conceito, a distribuição dos dados pode sofrer alterações, fazendo com que a performance do modelo se degrade significativamente. Isso ocorre porque o modelo foi otimizado para um contexto estatístico anterior e pode não ser capaz de generalizar adequadamente para a nova realidade.

Nesse cenário, a detecção dessa mudança e a adaptação do modelo tornam-se essenciais para manter um desempenho adequado. Estratégias comuns incluem atualização incremental, re-treinamento periódico ou a adoção de métodos adaptativos que permitam ao modelo ajustar-se continuamente às novas condições do sistema. Normalmente, a tarefa de *detectar novos eventos* e *modelar os eventos conhecidos* é dividida entre diferentes métodos, isto é, o método que detecta novos eventos (uma mudança de conceito) não é o mesmo que modela o fluxo de dados. Como métodos clássicos direcionados à detecção é possível citar o DDM [Gama et al., 2004] (do inglês, *Drift Detection Method*) e o EDDM [Baena-Garcıa et al., 2006] (do inglês, *Early Drift Detection Method*). A modelagem pode ser realizada por qualquer método que o usuário julgar apropriado, mas idealmente deve utilizar um esquema de aprendizado online ou ao menos incremental, principalmente para evitar retreinamentos ao surgirem novos eventos.

Considerando um problema de classificação binária, para cada entrada $X \in \mathbb{R}^n$, em que $n \in \mathbb{N}$ denota a quantidade de variáveis de entrada, é fornecido o respectivo rótulo $y \in \{1, -1\}$, com 1 denotando uma classe `Normal` e -1 uma `Novidade`. Formalmente, uma mudança de conceito caracteriza-se pela mudança da distribuição de probabilidade conjunta tendo como referência dois instantes distintos de tempo, t_0 e t_1, sendo formalmente definida como [Gama et al., 2014]:

$$\exists X : P_{t_0}(X, y) \neq P_{t_1}(X, y), \tag{2.1}$$

em que $P_{t_0}(X, y)$ denota a distribuição de probabilidade conjunta no instante t_0 e $P_{t_1}(X, y)$ no instante t_1 entre as variáveis X e y. Para simplificar, é possível ainda, por meio da Teoria de Decisão Bayesiana [Murphy, 2012], descrever $P(X, y)$ em função das probabilidades *a priori* e das distribuições de probabilidade condicionada em cada classe, de tal forma que a região de decisão é construída de acordo com as probabilidades *a posteriori* das classes [Gama et al., 2014],

$$P(y|X) = \frac{P(y)P(X|y)}{P(X)}, \tag{2.2}$$

com $P(X) = \sum_{y \in \{-1,1\}} P(y)P(X|y)$, assumindo mesmo custo para cada classe. É importante ressaltar que as mesmas definições podem ser utilizadas para problemas multiclasse, somente para fins didáticos limitou-se aqui a um problema binário.

Com base em (2.2) é possível então definir as três possíveis origens de mudanças que podem vir a ocorrer em $P(X, y)$ e que caracterizam uma mudança de conceito [Lu et al., 2019]:

- Origem I: $P_{t_0}(X) \neq P_{t_1}(X)$ enquanto $P_{t_0}(y|X) = P_{t_1}(y|X)$, ou seja, a região de decisão não se altera, como ilustrado na Figura 1a. Note que embora ocorra um deslocamento da distribuição de probabilidade (azul claro para vermelho), o limiar de decisão representado pela reta permanece o mesmo, não afetando o desempenho do modelo.

- Origem II: $P_{t_0}(y|X) \neq P_{t_1}(y|X)$ enquanto $P_{t_0}(X) = P_{t_1}(X)$, ou seja, nessa situação há uma mudança na região de decisão, o que leva a uma perda de desempenho do modelo, havendo a necessidade imediata de atualizá-lo. Uma ilustração dessa situação é apresentada na Figura 1b. Por meio da ilustração é possível observar que a distribuição de probabilidade das variáveis de entrada X, permanece a mesma, enquanto que sua relação com as classes muda, logo, o antigo limiar de decisão não mais representa essa relação.

- Origem III: $P_{t_0}(y|X) \neq P_{t_1}(y|X)$ e $P_{t_0}(X) \neq P_{t_1}(X)$, ou seja, ocorre a Origem I e II ao mesmo tempo. Como é possível observar por meio da ilustração na Figura 1c; tanto as variáveis de entrada, quanto sua relação com cada classe muda, logo, nem a distribuição de probabilidade, nem o limiar de decisão representam mais o novo contexto.

Além das origens, é possível ainda destacar como essas mudanças podem ocorrer. Na literatura, é comum utilizar quatro principais categorias para discriminá-las[Gama et al., 2014]:

- Abrupta: O novo conceito surge em um curto período, de forma inesperada, como ilustrado na Figura 2a.

- Gradual: Nesta categoria, a mudança para o novo conceito ocorre gradualmente, isto é, vai se demonstrando indícios ao longo do tempo de que o antigo conceito está perdendo representatividade, intercalando-se com o novo por um determinado

período de tempo até que a mudança se concretize. Uma ilustração dessa situação é dada na Figura 2b.

- Incremental: Aqui o conceito antigo sede lugar para o novo de forma lenta. Em geral, a mudança só é perceptível observando um grande período de tempo. O desgaste natural das peças de uma máquina é um bom exemplo dessa situação. Uma ilustração é dada na Figura 2c.
- Recorrente: Esta última categoria se refere a situação em que um conceito antigo sede lugar para um novo, mas após um certo período de tempo o antigo volta a ocorrer, como observado na ilustração apresentada na Figura 2d.

Como observado, mudanças de conceito refere-se a mudanças inesperadas nos padrões dos dados, fazendo com que um modelo previamente treinado se torne menos eficaz. A mesma exige mecanismos de detecção e adaptação, como re-treinamento periódico ou idealmente de métodos que ajustam o modelo em tempo real. O foco aqui está na adaptação de modelos previamente treinados para lidar com novos padrões, sem esquecimento catastrófico [Gama et al., 2014].

2.2 Sistemas Evolutivos

Os sistemas evolutivos, ao contrário de métodos clássicos, são projetados para aprender e se adaptar continuamente, incorporando novos dados sem a necessidade do re-treinamento tradicional. E sob uma perspetiva geral, são caracterizados por seguir uma filosofia de aprendizado mais próxima da praticada pelos seres vivos, isto é, adota-se uma postura em que o método evolutivo é um ser vivo que está num processo constante de aprendizagem, sempre aprimorando o conhecimento que possui e expandido-o tão logo que surjam novas informações e novas experiências [Angelov et al., 2010].

Sob uma visão mais técnica, essa classe de métodos não costuma adotar a visão probabilística apresentada anteriormente e realizam por si só a tarefa de detecção, classificação e modelagem dos eventos; utilizando limiares genéricos predefinidos indiretamente por meio da escolha de hiperparâmetros, esses tipos de métodos conseguem identificar o grau de ineditismo de uma nova informação, e assim, realizar os procedimentos apropriados. Ademais, os hiperparâmetros iniciais também podem ser adaptativos conforme o fluxo de dados em abordagens mais sofisticada [Lughofer and Sayed-Mouchaweh, 2015, Leite et al., 2012a, 2023]. Na literatura da área é comum utilizar a expressão *atualizar os parâmetros* para indicar a situação em que o método identificou informação similar em um certo grau àquelas previamente consolidadas no modelo, e como tal realiza um refinamento do modelo (dos parâmetros que o caracterizam um padrão de comportamento). Em contrapartida, quando a informação representa um novo evento, utiliza-se a expressão *atualizar a estrutura*

para indicar que uma mudança mais significativa deve ser realizada para incorporar a nova informação à base de conhecimento [Angelov and Filev, 2004, Angelov et al., 2010].

É fundamental destacar mais claramente a distinção entre a visão probabilística discutida anteriormente e a abordagem evolutiva. Na visão probabilística, o foco está em identificar quando os dados se desviam da distribuição probabilística conhecida durante o treinamento. Quando isso ocorre, o modelo é atualizado para se ajustar ao novo contexto, geralmente perdendo o foco estatístico relativo aos parâmetros e dados que representavam o evento ou fenômeno anterior. Por outro lado, na abordagem evolutiva, a ênfase está em quantificar o grau de novidade da nova informação em relação ao conhecimento prévio, utilizando-a para aprimorar o modelo [Lemos, 2011, Inacio, 2014, da Silva, 2014]. Esse aprimoramento pode envolver o refinamento dos parâmetros, a expansão da estrutura para acomodar novas informações ou, inversamente, a contração do modelo para descartar elementos menos relevantes ou ser mais específico em uma representação. Em resumo, enquanto a visão probabilística busca representar com máxima fidelidade a relação atual entre as variáveis por meio da densidade de probabilidade ou outras propriedades estatísticas acumuladas, a abordagem evolutiva prioriza a manutenção de modelos compactos e adaptáveis, algumas vezes com foco na interpretabilidade e apoio à tomada de decisão humana, que incorporem de forma eficiente toda a informação relevante observada ao longo do tempo.

A diferença citada é mais facilmente percebida por meio da ilustração apresentada na Figura 3. A figura apresenta uma situação em que ocorre uma mudança incremental entre dois contextos ou eventos (Figura 3a)). Note que na abordagem probabilística (Figura 3b), o foco está em fazer o modelo/distribuição acompanhar a relação atual dos dados; na ilustração o modelo é representando pelo circulo, de forma que as circunferências tracejadas e a contínua representam os estados antigos e o atual do modelo, respectivamente. Vale ressaltar também que nessa perspectiva os contextos antigos foram esquecidos. Em contrapartida, na abordagem evolutiva a prioridade está em aprimorar o conhecimento, refinando-o e expandindo-o quando necessário. Na ilustração (Figura 3c), inicialmente a estrutura do modelo caracterizava-se apenas pelo círculo rosa, contudo, após observar as novas informações, sua estrutura foi expandida (círculos pretos) para incorporá-las a base de conhecimento.

2.3 Arquitetura Geral para Detecção e Classificação de Eventos

Com base na fundamentação evolutiva, esta tese propõe um arquitetura geral que faz uso de métodos evolutivos com um esquema de aprendizado não supervisionado para a detecção e classificação de eventos. O termo *evento*, dentro da arquitetura, é empregado de

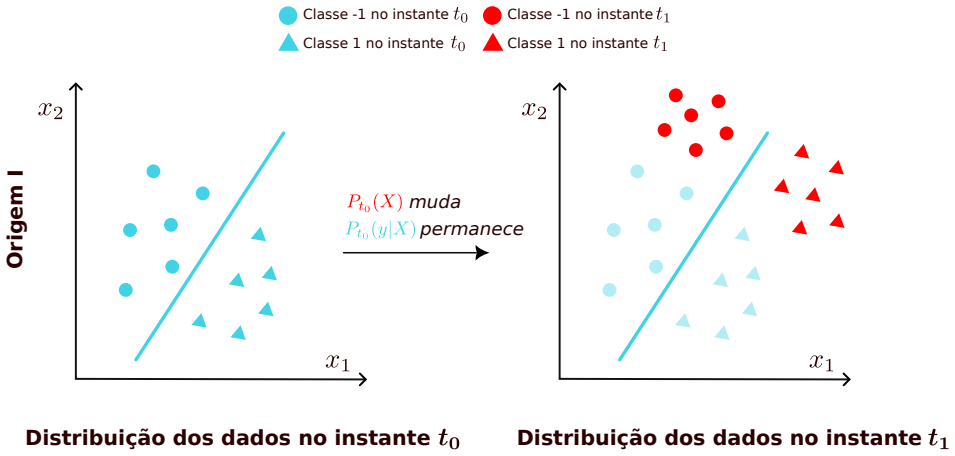
forma genérica para se referir a qualquer novo padrão de dados identificado pelo método evolutivo e armazenado em sua base de conhecimento, ou seja, em sua estrutura interna.

A arquitetura proposta é caracterizada por: (i) processamento contínuo das leituras do sistema, garantindo uma adaptação dinâmica às mudanças no ambiente; (ii) construção incremental de uma base de conhecimento, na qual cada evento registrado representa um comportamento distinto do sistema, permitindo que essa base seja constantemente atualizada à medida que novas amostras são processadas; (iii) flexibilidade na rotulagem, possibilitando que o especialista humano substitua o rótulo genérico *evento* por uma nomenclatura mais descritiva, alinhada ao comportamento real do sistema; e (iv) utilização dessa base de conhecimento como uma ferramenta essencial para a supervisão do sistema, fornecendo suporte à tomada de decisão. Uma visão geral dessa arquitetura é ilustrada na Figura 4.

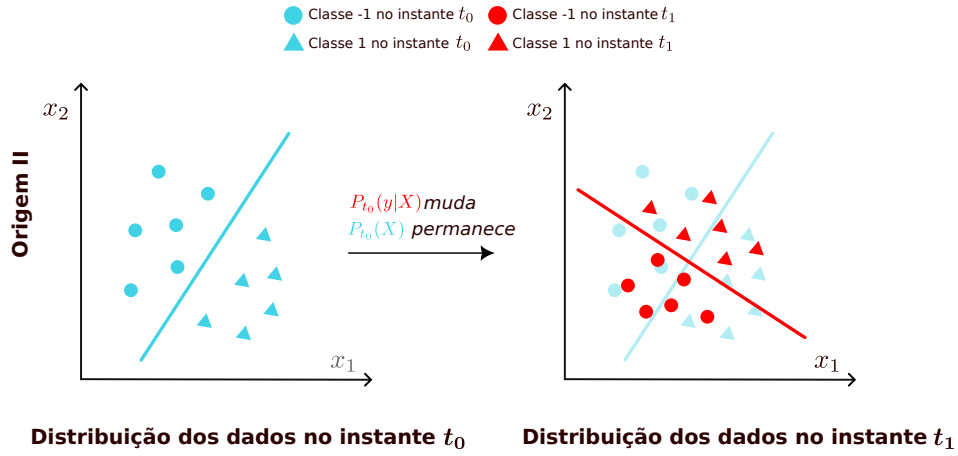
2.4 Resumo do Capítulo

Este capítulo contrastou de forma genérica as ideias básicas que sustentam modelos probabilísticos adaptativos e modelos inteligentes evolutivos que sustentam os principais métodos de sucesso na literatura, destacando a abordagem evolutiva como a filosofia adotada em todas as propostas desta tese. Além disso, foi introduzida uma arquitetura geral para modelagem evolutiva, na qual a participação do especialista humano é fundamental para o aprimoramento do conhecimento construído pelo método evolutivo.

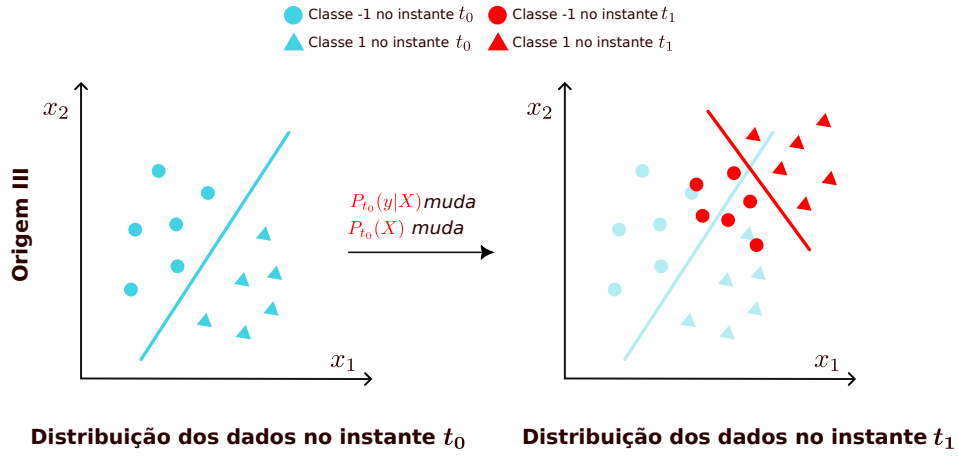
Nessa arquitetura, os métodos evolutivos seguem um esquema de aprendizado não supervisionado, cujo objetivo é incorporar todos os comportamentos distintos do sistema em uma base de conhecimento. Essa base permite que um especialista humano caracterize e rotule os eventos observados, assegurando uma representação quantitativa e qualitativa abrangente, adaptativa e continuamente atualizada da dinâmica do sistema ao longo do tempo.



(a)



(b)



(c)

Figura 1 – Ilustração das mudanças de distribuição de probabilidade conjunta considerando as três possíveis origens de uma mudança de conceito. As retas indicam a fronteira de decisão tomando como referência um modelo linear hipotético.

Fonte: Adaptado de Lu et al. [2019].

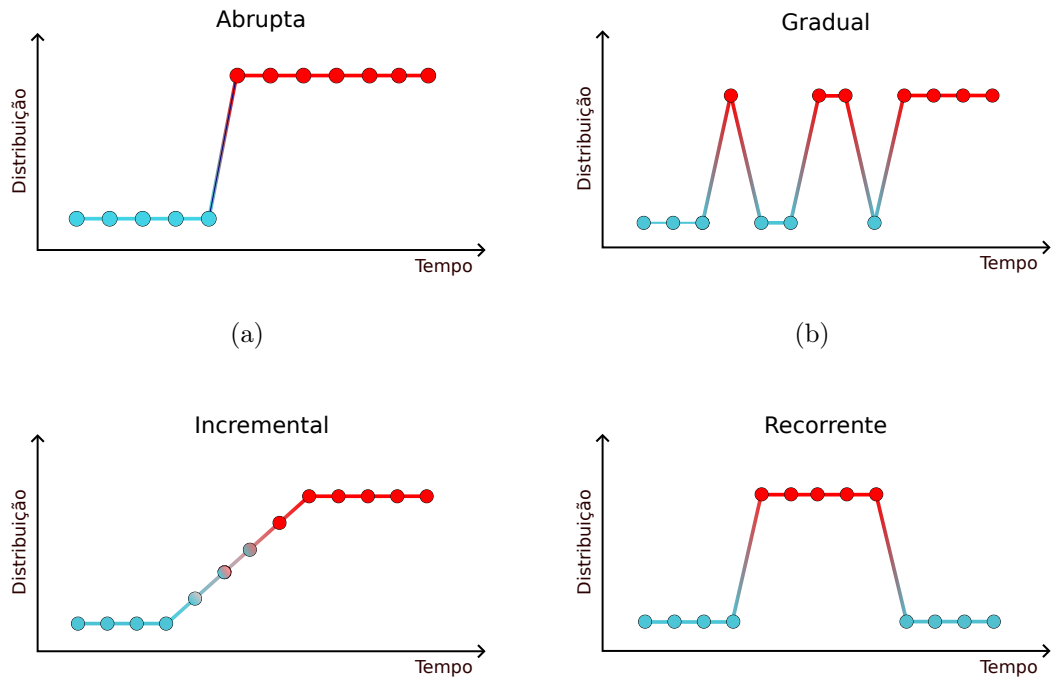


Figura 2 – Ilustração dos tipos de mudanças que podem ocorrer durante uma mudança de conceito.

Fonte: Adaptado de Gama et al. [2014], Lu et al. [2019].

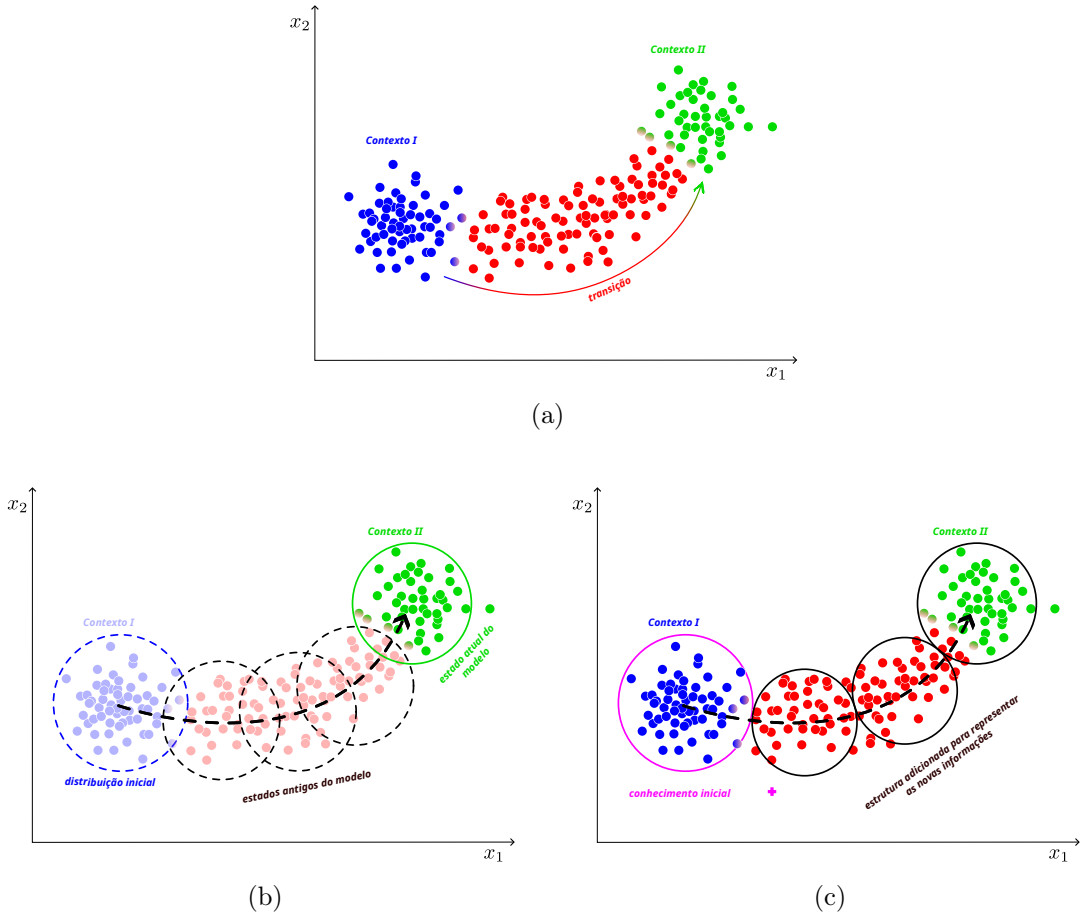


Figura 3 – Ilustração da modelagem de eventos utilizando a perspectiva probabilística e a evolutiva. (a) Na situação em que ocorre uma mudança do Contexto I para o Contexto II em um dado sistema. (b) Seguindo as fundamentações probabilísticas relativas a mudança de conceito, o modelo deve esquecer o Contexto I e ser re-treinado para o Contexto II; no entanto, (c) seguindo a visão evolutiva, os dois contextos, bem como a transição, são mantidos na base de conhecimento do método, pois todos representam comportamentos distintos do sistema.

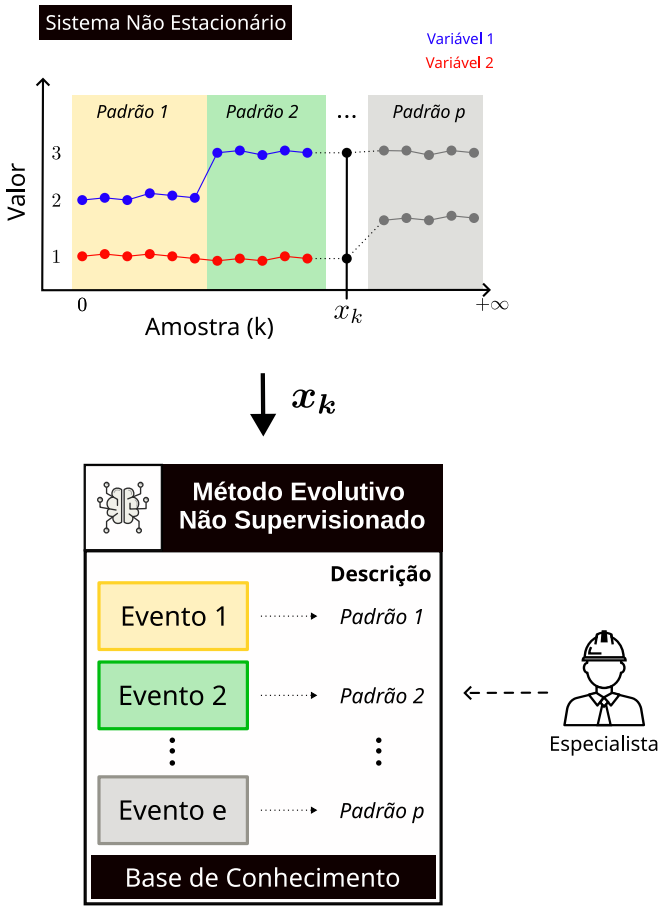


Figura 4 – Arquitetura geral para detecção e classificação de eventos. O método evolutivo processa continuamente as amostras de dados, identificando novos eventos e incorporando-os dinamicamente à sua base de conhecimento. O especialista, por sua vez, pode atribuir descrições aos eventos detectados, alinhando-os ao comportamento real do sistema.

Capítulo 3

Modelagem de Eventos Baseada em AutoEncoders Evolutivos

Métodos baseados em dados são usados há décadas para criar representações mais simplificadas da dinâmica de sistemas complexos, sobretudo os de alta dimensionalidade e comportamento não linear [Chiu, 1994, Ribeiro, 1999, Babuška and Verbruggen, 2003, Angelov and Filev, 2004, Ghani et al., 2023a, Li et al., 2024b, He et al., 2025].

Um exemplo clássico é a Análise de Componentes Principais (PCA) (do inglês, Principal Component Analysis) [Bishop and Nasrabadi, 2006], um método amplamente utilizado para redução de dimensionalidade [Nhat and Lee, 2005, Mishra and Motaung, 2015, Bueso et al., 2018, Elshenawy and Mahmoud, 2018, Rehman et al., 2020, Zheng and Zheng, 2022]. Nessa abordagem, os dados históricos representativos do comportamento normal do sistema, são utilizados como referência para projetar as amostras em um espaço de menor dimensionalidade, preservando a maior parte da variabilidade. A estrutura estatística desse comportamento normal, refletida nesse novo espaço, permite então a definição de limiares para a detecção de novos comportamentos ou anomalias.

Formalmente, considerando as variáveis de entrada $\mathrm{X} \in \mathbb{N}^n$, em que n representa o número de variáveis do sistema, a PCA aplica uma transformação linear que gera uma nova representação $\mathrm{Z} \in \mathbb{R}^l$, com $l \leq n$. Essa nova representação, conhecida como *representação latente*, reduz a dimensionalidade dos dados ao projetá-los em um espaço de menor dimensão, preservando ao máximo sua variabilidade e tornando a informação mais compacta e simplificada. Embora a PCA esteja restrita à captura de padrões lineares e dependa de algumas suposições sobre a distribuição dos dados (como normalidade), ela frequentemente fornece uma representação útil, sendo eficaz em diversas aplicações práticas.

Não obstante, este capítulo trata do aprendizado de representação não supervisionado, com foco no uso de AutoEncoders, uma arquitetura de rede neural artificial que supera as limitações da PCA ao permitir a extração de informações latentes não lineares,

de formar a capturar padrões mais complexos nos dados. O objetivo geral é utilizar essa nova representação como uma forma mais simples de visualizar as dinâmicas do sistema, e a modelagem em si como uma referência dos comportamentos observados, utilizando-o como métrica para julgar se uma nova entrada representa um novo comportamento.

O capítulo inicia apresentando os fundamentos dos autoencoders e trabalhos relacionados encontrados na literatura. Em seguida, são apresentados alguns principais conceitos sobre granulação de informação - conceitos esses utilizados na proposta de um autoencoder evolutivo granular. Por fim, um *framework* geral é proposto para realizar a modelagem com base nos resíduos de reconstrução das entradas.

3.1 AutoEconders

AutoEncoders são um tipo específico de rede neural artificial que busca extrair a essência da informação ou os graus de variabilidade fundamentais que geraram os dados do espaço de entrada original $X \in \mathbb{R}^n$ e projetá-la, normalmente, em um espaço com menor dimensão $Z \in \mathbb{R}^l$. Para isso, a rede é dividida em duas partes. A primeira é chamada *Encoder*. O encoder é responsável por receber a representação original e transformá-la na representação latente. A segunda é chamada de *Decoder*. Este componente é responsável por receber a representação latente e transformá-la de volta na representação original Hinton and Salakhutdinov [2006], Goodfellow et al. [2016]. A arquitetura geral de um AutoEncoder é ilustrada na Figura 5.

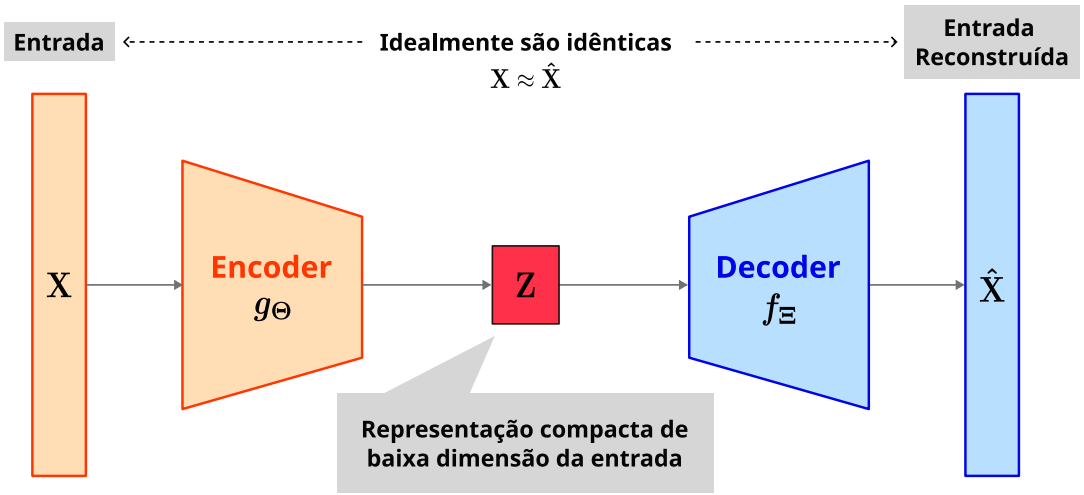


Figura 5 – Estrutura geral de um AutoEncoder.

Fonte: Adaptado de Weng [2018].

Formalmente, uma rede AutoEncoder é definida como:

$$\hat{\mathrm{X}} = f_{\Xi}(g_{\Theta}(\mathrm{X})) \tag{3.1}$$

em que $g_{\Theta} : \mathbb{R}^n \rightarrow \mathbb{R}^l$ representa o Encoder, $f_{\Xi} : \mathbb{R}^l \rightarrow \mathbb{R}^n$ o Decoder, Θ e Ξ representam respectivamente os parâmetros do Encoder e Decoder que são ajustados durante o treinamento da rede, e $\hat{\mathrm{X}} \in \mathbb{R}^n$ é a reconstrução obtida para uma dada entrada X.

Sob uma perspectiva geral, nesse tipo de rede o processo de aprendizado consiste em, a partir de uma estrutura predefinida, ajustar iterativamente seus pesos, Θ e Ξ, de forma a melhorar a reconstrução obtida com o decoder. Definido esse objetivo, o processo de aprendizado tende a minimizar o erro de reconstrução, incentivando o encoder a preservar, no espaço latente, a estrutura topológica dos dados no espaço de entrada. Ou seja, pontos próximos no espaço de entrada também permanecerão próximos no espaço latente [Hinton and Salakhutdinov, 2006, Goodfellow et al., 2016].

Embora existam diversos tipos de autoencoders, cada um direcionado a aplicações específicas, todos baseiam-se nessa definição, mudando apenas a estrutura utilizada para representar cada parte e/ou a função de custo. Os tipos mais comuns são:

- *Denoising Autoencoder* (DAE)[Vincent et al., 2008, 2010]: Voltado especialmente a problemas com alta incidência de ruídos, esta arquitetura busca aprender a representação latente com um versão corrompida da entrada, isto é, a entrada fornecida ao Encoder é uma versão corrompida/ruidosa da entrada original, mas ao Decoder é fornecida a entrada limpa. Com isso, para conseguir reconstruir a saída perfeitamente a rede precisa aprender uma representação latente que desconsidera a parcela de corrupção/ruídos na entrada original, de forma a realizar também a ação de um filtro.

- *Sparse Autoencoder* [Ng et al., 2011, Erhan et al., 2010]: Nessa arquitetura, um termo de penalidade esparsa é adicionado aos pesos do Encoder por meio da função de custo. Essa restrição força o modelo a ativar apenas um pequeno número de neurônios no espaço latente para cada entrada, o que pode levar a representações mais eficientes e simplificadas.

- *Stacked Autoencoder* (SAE)[Hinton and Salakhutdinov, 2006, Liu et al., 2018]: Possui vários autoencoders empilhados para maximizar a capacidade de modelagem da rede. Cada camada é treinada de forma gulosa e não supervisionada, uma após a outra, para aprender representações hierárquicas e cada vez mais abstratas dos dados. Após o treinamento não supervisionado, a rede pode ser ajustada de forma supervisionada para tarefas específicas, como classificação ou regressão.

- *Variational Autoencoder* (VAE) [Kingma and Welling, 2013]: Transforma a rede em um modelo generativo em que é aprendido a distribuição de probabilidade dos dados.

Diferente de um autoencoder tradicional, que aprende um mapa de vetor real para vetor real, nessa arquitetura se aprende uma distribuição de probabilidade no espaço latente. Isso permite que o modelo seja utilizado para gerar novos dados semelhantes aos dados de treinamento, tornando-o uma ferramenta poderosa para geração de dados e aprendizado de representação a partir de perturbações no espaço latente.

As aplicações práticas desse tipo de rede são diversas [Goodfellow et al., 2016, Bank et al., 2020]: geração de dados; classificação; clusterização; sistemas de recomendação; redução de dimensionalidade; detecção de anomalia; tradução de idiomas, dentre muitas outras. Nesta tese, o uso do autoencoder está sendo empregado na modelagem evolutiva de fluxo de dados, em que a rede aprende incrementalmente e continuamente uma representação latente das dinâmicas observadas do sistema. A próxima seção apresenta as principais abordagens propostas na literatura que utilizam autoencoders na modelagem de processos industriais em geral.

3.1.1 Revisão de autoencoders empregados na modelagem de sistemas

No que tange ao uso de Autoencoders na modelagem de sistemas não-estacionários, os trabalhos encontrados na literatura, em sua maioria, propõem uma metodologia de aprendizado estática, isto é, a partir de um conjunto de dados históricos dos comportamentos conhecidos do sistema, um Autoencoder é treinado para utilizá-lo como detector de anomalias (ou novidades), somente. Alguns empregaram classificadores treinados com as informações residuais geradas pela diferença entre as entradas e as reconstruções geradas pelo Decoder; porém, permaneceram sob uma metodologia estática de aprendizado. Nesse sentido, os trabalhos que serão apresentados nesta seção em sua maioria são apenas detectores de anomalias, sendo o mais próximo da metodologia proposta nesta tese.

Em Gupta et al. [2022], os autores propuseram uma abordagem para detecção e diagnóstico de falhas em sistemas industriais, utilizando Autoencoders combinados com técnicas de Inteligência Artificial (IA) interpretável. O estudo é aplicado a um *chiller* industrial, um sistema crítico para o controle de temperatura em processos industriais, onde a detecção precoce de falhas é essencial para evitar paradas não planejadas e custos elevados. O método proposto consiste em utilizar um Autoencoder treinado com dados normais (livres de falhas) para modelar o comportamento normal do sistema, o erro de reconstrução como um índice de detecção de falhas, e técnicas de IA interpretável, tais como *SHapley Additive exPlanations* (SHAP) [Lundberg and Lee, 2017] e *Local Interpretable Model-agnostic Explanations* (LIME) [Ribeiro et al., 2016] para identificar as variáveis do processo que mais contribuíram para a anomalia. De acordo com os resultados obtidos, a abordagem é eficaz na detecção de anomalias, permitindo a identificação precoce de

falhas antes de causarem impactos significativos no sistema, e além disso a utilização de técnicas de IA interpretável, permite identificar as causas das falhas, facilitando a tomada de decisões para manutenção corretiva. Contudo, o método possui dependência de dados rotulados e treinamento prévio, além de poder demandar grande esforço computacional, especialmente devido aos métodos de interpretabilidade.

Diallo et al. [2024] propõem uma metodologia para diagnóstico de falhas em processos industriais, combinando autoencoders com métodos tradicionais de classificação. O estudo é aplicado a uma linha de galvanização, um processo crítico na indústria siderúrgica, onde a detecção e identificação de falhas são essenciais para evitar paradas de produção e garantir a qualidade do produto. O método proposto substitui a abordagem tradicional de definir um limiar para o índice de detecção de eventos baseado no erro de reconstrução do AE, por um classificador treinado com dados residuais. A metodologia envolve: a) treinamento do AE com dados referentes ao comportamento normal do processo, b) geração resíduos (calcular a diferença entre os dados originais e reconstruídos) para o comportamento normal e demais comportamentos (falhas de processo); e c) avaliação do desempenho do método com métricas de classificação em dados de teste. O método apresentou bom desempenho no ambiente avaliado. Seguindo um caminho similar ao anterior, um ponto crítico do método é a necessidade prévia de uma base de dados rotuladas para os comportamentos de interesse do sistema, i.e., o método não é auto adaptativo.

O método *Variational Discriminative Stacked Auto-Encoder* (VDSAE) proposto por Huang et al. [2024] é voltado a representação latente e monitoramento de processos industriais. A abordagem combina Autoencoders Empilhados (SAEs) com técnicas variacionais (probabilidade) e discriminativas (classificador). O método visa melhorar a capacidade de extração de características e a robustez do modelo em cenários industriais complexos, sendo integrado por três componentes principais: 1) Autoencoder Empilhado (SAE): Utilizado para aprender representações hierárquicas dos dados de entrada, capturando características em múltiplos níveis de abstração; 2) Componente Variacional: Incorpora um Autoencoder Variacional (VAE) para modelar a distribuição probabilística dos dados no espaço latente, permitindo a geração de novas amostras e melhorando a generalização do modelo; e, 3) Discriminador Pré-treinado: Um classificador é pré-treinado para distinguir entre dados normais e anômalos, guiando o treinamento do SAE para melhorar a representação de características relevantes para o monitoramento de processos. O modelo treinado é utilizado para monitorar processos industriais, detectando anomalias com base no erro de reconstrução ou de forma similar ao anterior, utilizando um classificador treinado com a informação residual. Seus resultados superaram o de métodos tradicionais, como PCA, SAE e VAE, em termos de precisão na detecção de anomalias, contudo, sofre das mesmas questões apontadas ao método anterior e, além disso possui alta complexidade computacional devido a integração dos múltiplos componentes (SAE,

VAE e discriminador).

Yao et al. [2024] propuseram uma arquitetura de autoencoder assimétrica denominada SVD-AE. Na abordagem, dois codificadores são empregados para extrair características nas dimensões de tempo e variáveis, respectivamente. Um decodificador compartilhado é utilizado para gerar reconstruções com base nas representações latentes provenientes de ambas as dimensões. Além disso, uma nova regularização baseada na teoria da Decomposição em Valores Singulares (SVD) (do inglês, *Singular Value Decomposition*) é introduzida para forçar cada codificador a aprender características no eixo correspondente, oferecendo suporte matemático para essa abordagem. Adicionalmente, um componente de perda específico é proposto para alinhar os coeficientes de Fourier das entradas e das reconstruções, preservando detalhes dos dados originais e aprimorando a capacidade de aprendizado de características do modelo. Para indicar a ocorrência de anomalias é utilizado um limiar predefinido e dependente do problema para o erro de reconstrução. O SVD-AE foi avaliado utilizando três conjuntos de dados reais, demonstrando desempenho superior em tarefas de detecção de anomalias em séries temporais multivariadas sob cenários altamente desbalanceados quando comparado aos algoritmos utilizados como referência. Contudo, a abordagem é estática, apenas detecta anomalias, não se auto adapta e pode demandar alto esforço computacional no processo de treinamento e inferência.

Mais recentemente, Liu et al. [2025] propuseram uma arquitetura que combina uma rede *Long Short-Term Memory* (LSTM) para extrair características dinâmicas dos dados. Simultaneamente, variáveis de qualidade são incorporadas em paralelo para capturar características relacionadas à qualidade. O erro de reconstrução e o erro quadrático médio da previsão (SPE) (do inglês, *Squared Prediction Error*), além da estatística de monitoramento de Hotelling T^2 são utilizadas para detectar anormalidades. A abordagem permite uma análise mais robusta dos dados, considerando tanto a dinâmica do processo quanto a qualidade do produto sendo produzido. A metodologia proposta é composta por duas fases: 1ª) treinamento offline com dados da situação normal do processo e definição dos limiares das métricas de detecção de anormalidade; e, 2ª) monitoramento online utilizando as métricas de estatísticas. A abordagem foi avaliada no bechmark *Tennessee Eastman Process* (TEP) Downs and Vogel [1993], obtendo boa performance; porém, o método não adapta a novas dinâmicas do processo, e além disso é computacionalmente caro.

Diversas outras abordagens foram propostas na literatura seguindo essa mesma metodologia geral, tais como [Wang et al., 2020, Kaupp et al., 2022, Hu et al., 2022, Zeng et al., 2023, Gao et al., 2023, Wang et al., 2024, Yue et al., 2024, Li et al., 2024a, Jang et al., 2025]. As variações em sua maioria são referentes a arquitetura da rede, de forma que todos operam sob um regime de treinamento offline, e na maioria das vezes necessitam de uma base de dados históricos rotulada, e com tamanho razoável.

Seguindo a linha de aprendizado dessa tese, a evolutiva, poucos trabalhos foram propostos. O encontrado que mais se assemelha ao contexto desta tese é o de Senthilnath et al. [2024]. Nesse trabalho os autores combinaram autoencoders com *Q-learning* [Watkins and Dayan, 1992] para criar um sistema de aprendizado por reforço que evolui automaticamente. O método, chamado *Self-evolving Autoencoder Embedded Q-Network* (SAE-QN), utiliza um autoencoder para aprender representações compactas e significativas dos dados, enquanto o *Q-learning* é responsável por tomar decisões ótimas em um ambiente dinâmico. A principal contribuição do trabalho é a capacidade do sistema de auto-evoluir, adaptando-se a mudanças no ambiente sem a necessidade de intervenção manual. Embora o método consiga aprender de forma evolutiva, requer um alto esforço computacional e está muito suscetível a qualidade dos dados recebidos, podendo ter seu desempenho degradado facilmente devido a certas escolhas feitas pelo *Q-learning*, que por sua vez são advindas dos dados. Além disso, devido a grande quantidade de parâmetros pré-definidos, pode vir a ser trabalhoso obter aquela que melhor direciona o aprendizado em cada ambiente. No entanto, vale salientar que a abordagem representa um grande marco no que diz respeito ao uso de aprendizado evolutivo em redes profundas, sendo uma área ainda pouca explorada ou com poucos resultados positivos.

Nesse sentido, visando contribuir para a evolução da área, na Seção 3.2 é apresentada uma proposta de autoencoder evolutivo que aprende incrementalmente a representação latente.

3.1.2 Granulação de informação

A capacidade humana de processar informações de maneira eficaz está intimamente ligada à habilidade de realizar granulação de informação. Tal qual apresentou Zadeh [1997], a granulação de informação refere-se ao processo de dividir informações complexas em partes menores e mais manejáveis, chamadas de "grânulos". Esses grânulos podem ser conjuntos, classes ou clusters que representam uma agregação de elementos com características ou propriedades semelhantes. Na tarefa de reconhecer uma cabeça humana, por exemplo, primeiramente ocorre uma granulação da informação, a imagem da cabeça, em partes menores: testa, olhos, orelhas, boca, nariz, etc.

No paradigma de Computação Granular (GrC) [Pedrycz, 2007] (do inglês, *Granular Computing*), grânulos de informação são utilizados para resolver problemas complexos de maneira eficiente, sendo que o nível necessário de granularidade depende da aplicação. Essa dependência torna-se evidente ao considerar que o sucesso da interação humana com o ambiente e na resolução de problemas está diretamente associado à capacidade de perceber o mundo real em diferentes níveis de granularidade e alternar livremente entre eles.

Sob uma perspectiva computacional e/ou matemática, a representação de grânulos de informação é dependente do domínio da aplicação, e normalmente são baseados em

conjuntos clássicos, conjuntos nebulosos, conjuntos rough, em clusters e distribuições de probabilidade (grânulos probabilísticos) Pedrycz [2007], Kreinovich [2011], Pedrycz [2011], Leite et al. [2012b], Pedrycz and Homenda [2013], Yao [2016], Cordovil et al. [2020], Leite et al. [2025]. Neste trabalho, grânulos de informação são representados por meio de duas medidas estatísticas, média e dispersão, de subconjuntos de dados. O nível de granularidade, que nesse contexto significa o tamanho do subconjunto de dados, é definido por um hiperparâmetro ajustado no início da execução do método.

Dois conceitos essenciais no que se refere a granularidade de informação são i)*cobertura* e ii)*especificidade* [Yager, 2008, Pedrycz and Homenda, 2013]. Este conceitos estabelecem objetivos conflitantes e estão atrelados a tarefa de criar grânulos com um grau mínimo de representação global (todo o conjunto de dados) e individual (cada dado do conjunto), isto é, considerando um grânulo $\boldsymbol{\mathcal{G}}$ e um subconjunto de dados $\mathbf{D}$ (evidência experimental), tem-se, conforme abaixo.

i) *Evidência experimental*: A evidência acumulada dentro dos limites de $\boldsymbol{\mathcal{G}}$ tem que ser tão alta quanto possível. Definindo isso, pode-se dizer que a existência desse grânulo é bem motivada (justificada), pois o mesmo irá representar (*cobrir*) todos dados experimentais observados naquela região do espaço. Por exemplo, se $\boldsymbol{\mathcal{G}}$ é definido como um conjunto intervalar, quanto mais dados são observados dentro desse grânulo (nesse intervalo), maior é a legitimidade de sua existência.

ii) *Significado semântico*: Ao mesmo tempo, o grânulo de informação deve ser tão *específico* quanto possível. Este requisito implica que a informação granular resultante deve possuir um significado semântico bem definido. Em outras palavras, deseja-se que $\boldsymbol{\mathcal{G}}$ tenha informação altamente detalhada, especificando exatamente o que o grânulo representa. Isso significa que, quanto mais compacto for o grânulo de informação, melhor será sua qualidade.

Por exemplo, a Figura 6, extraída de Pedrycz and Homenda [2013], ilustra o processo de otimização dos intervalos que definem o grânulo $\boldsymbol{\mathcal{G}}$ com base na *cobertura* e *especificidade*. No exemplo, uma representação numérica sensata dos dados seria a mediana, med($\mathbf{D}$). No processo de otimização do grânulo de informação intervalar, observe que quanto maior os limites a e b, maior a medida de *cobertura* e menor a de *especifidade*, em contrapartida, quanto menor os limites a e b em torno da mediana, maior a medida de *especifidade* e menor a de *cobertura*.

Nesta tese, optou-se pode definir o grau mínimo de significância de um grânulo considerandos essas duas medidas através de dois hiperparâmetros. O primeiro, denotado por $\tau \in \mathbb{N}$ está relacionado a *cobertura*, e define a quantidade mínima de amostras que devem existir em um grânulo antes de outro ser criado; o segundo, denotado por $\delta \in \mathbb{R}$ está

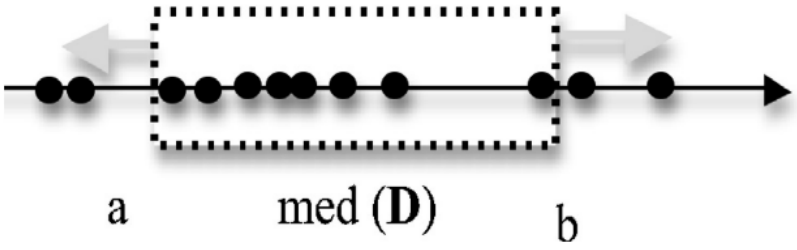


Figura 6 – Ilustração das possibilidades de representação de um grânulo de informação intervalar $\boldsymbol{\mathcal{G}} = [a, b]$ com base nas medidas de *cobertura* e *especificidade*.

relacionado a *especificidade*, e indica a distância mínima que deve existir de um grânulo para outro. Mais detalhes serão fornecidos na próxima seção.

3.2 Proposta de Autoencoder Evolutivo

Para que uma rede neural multicamada preserve algum nível de memória sobre eventos passados, é essencial que os dados relevantes estejam representados em sua estrutura. No entanto, em fluxos de dados, novas leituras são geradas em intervalos curtos de tempo, frequentemente em segundos ou até milissegundos. Nesses cenários, o retreinamento contínuo da rede com grandes volumes de dados torna-se inviável, tanto devido à necessidade de *hardware* especializado e de alto custo, quanto pela ausência de garantias de que a arquitetura originalmente definida será capaz de capturar adequadamente toda a dinâmica do fluxo. Afinal, a definição da arquitetura da rede é, em grande parte, guiada por conhecimento técnico aliado a experiência empírica, o que pode não ser suficiente para abranger as variações dos dados ao longo do tempo.

Nesse sentido, neste trabalho grânulos de informação são empregados com o intuito de compactar conjuntos de informações similares para simplificar a tarefa de aprendizado pela rede. Com isso, ao invés de reapresentar para a rede conjuntos com milhares de amostras de dados referentes ao passado, são entregues apenas algumas dezenas de grânulos de resumem todo esse conjunto de dados (descontadas as devidas perdas geradas com a compactação).

Dito isso, nesta parte da tese é apresentada a proposta de um autoencoder evolutivo em que tanto o encoder quanto o decoder são arquiteturas definidas previamente, podendo variar a depender da aplicação. A capacidade de aprendizado evolutivo foi incorporada por meio de grânulos de informação, que são utilizados pela rede ao longo de seu aprendizado para preservar o conhecimento do passado. A arquitetura proposta foca o aprendizado na representação latente, com isso o decoder pode ser uma rede mais simples em comparação ao encoder, na situação em que a qualidade da reconstrução não é de interesse. Para facilitar futuras referências, essa abordagem será denominada a partir desse ponto de

eGAE(do inglês, *Evolving Granular AutoEncoder*).

A definição formal do eGAE segue a definição geral de autoencoder apresentada em (3.1). O mecanismo de aprendizado evolutivo está associado a função de perda e ao conjunto de grânulos de informação: o conjunto de grânulos é atualizado continuamente e incrementalmente para representar de forma compacta o passado observado, e a função de perda, ao aprender com novos dados utiliza esse conjunto para preservar na rede o respectivo conhecimento. De forma prática, com essa modificação, durante a atualização dos pesos da rede uma regularização implícita é adicionada, evitando que a rede foque apenas no aprendizado dos novos dados. A função de perda proposta é descrita a seguir:

$$\begin{aligned}\mathcal{L} &= \lambda_1 \cdot \frac{1}{w} \left\| f_\Xi \left(g_\Theta(\mathrm{X}_t) \right) - \mathrm{X}_t \right\|^2 + \lambda_2 \cdot \frac{1}{g_t} \left\| f_\Xi \left(g_\Theta(\mathrm{C}_t) \right) - \mathrm{C}_t \right\|^2 \\ &= \lambda_1 \cdot \frac{1}{w} \|\hat{\mathrm{X}}_t - \mathrm{X}_t\|^2 + \lambda_2 \cdot \frac{1}{g_t} \|\hat{\mathrm{C}}_t - \mathrm{C}_t\|^2,\end{aligned} \tag{3.2}$$

em que $\mathrm{X}_t \in \mathbb{R}^{w \times n}$ e $\hat{\mathrm{X}}_t \in \mathbb{R}^{w \times n}$ representa uma janela de dados (ou mini-lote ou *mini-batch* no inglês) de $w \in \mathbb{N}$ amostras do passado do sistema (em referência ao instante atual t) e as reconstruções geradas pelo decoder, respectivamente; $\mathrm{C}_t \in \mathbb{R}^{g_t \times n}$ e $\hat{\mathrm{C}}_t \in \mathbb{R}^{g_t \times n}$ representa os centros dos grânulos de informação e suas estimativas geradas pelo decoder, respectivamente; e, $g_t \in \mathbb{N}$ representa a quantidade de grânulos no instante atual. Cada grânulo é construído a partir do cálculo incremental da média e dispersão de amostras consideradas similares de acordo com alguns critérios pré-estabelecidos, mais detalhes serão dados ao longo desta seção. Por fim, $\lambda_1 \in \mathbb{R}_+$ e $\lambda_2 \in \mathbb{R}_+$ são parâmetros pré-definidos que indicam a importância de cada termo no aprendizado da rede; foi utilizado $\lambda_1 = 0.1$ e $\lambda_2 = 1 - \lambda_1$ ao longo de todo o trabalho, ressalta-se que, embora essas definição seja complementar, não há obrigatoriedade de que o seja.

A seguir são descritos os procedimentos utilizados na manutenção do conjunto de grânulos.

Criação e atualização de grânulos

Criação

Um grânulo $\mathcal{G}_i$, com $i \in \mathbb{N}$ denotando um índice, é representado por uma tripla $(\mathcal{M}_i, \mathcal{D}_i, \mathcal{N}_i)$, em que $\mathcal{M}_i \in \mathbb{R}^n$, $\mathcal{D}_i \in \mathbb{R}$ e $\mathcal{N}_i \in \mathbb{N}$ representa a média ou centro, a dispersão e a quantidade das amostras que foram que o mesmo representa, respectivamente.

A cada nova janela de dados $\mathrm{X}_t \in \mathbb{R}^{w \times n}$, a seguinte condição é avaliada após a etapa de atualização, com toda amostra $\mathrm{x}_k \in \mathrm{X}_t$, para verificar a necessidade da criação de um novo grânulo:

$$\mathcal{N}_i \geq \tau \text{ e } \mathcal{D}_i \neq 0 \text{ e } \|\mathcal{M}_i - \mathrm{x}_k\|^2 > \delta\mathcal{D}_i \qquad \forall \mathcal{G}_i = (\mathcal{M}_i, \mathcal{D}_i, \mathcal{N}_i) \in \mathrm{G}_t, \tag{3.3}$$

em que $\tau \in \mathbb{N}$ e $\delta \in \mathbb{R}$ representam limiares pré-definidos para a quantidade mínima de amostras (medida mínima de *cobertura*) que um grânulo deve representar e a distância mínima (medida máxima de *especificidade*) que cada grânulo deve estar dos demais, respectivamente. $\mathrm{G}_t = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_{g_t}\}$ é o conjunto de grânulos existentes no instante atual, com $g_t \in \mathbb{N}$ indicando quantidade de elementos.

Uma vez que (3.3) tenha sido satisfeita, um novo grânulo $\mathcal{G}_{g_t+1}$ é adicionado a G_t com os seguintes valores iniciais:

$$\mathcal{G}_{g_t+1} = (\mathcal{M}_{g_t+1} = \mathrm{x}_k, \mathcal{D}_{g_t+1} = 0, \mathcal{N}_{g_t+1} = 1). \tag{3.4}$$

Atualização

A atualização dos grânulos ocorre em duas situações e sempre considerando o grânulo mais próximo a x_k, denotado por i^*,

$$i^* = \underset{i \in \{1,2,\ldots,g_t\}}{\operatorname{argmin}} \|\mathrm{x}_k - \mathcal{M}_i\|^2. \tag{3.5}$$

A primeira situação diz respeito aos grânulos que ainda não atingiram uma medida mínima de significância, isto é,

$$\mathcal{D}_{i^*} == 0 \textbf{ ou } \mathcal{N}_{i^*} < \tau, \tag{3.6}$$

e, a segunda diz respeito a situação em que a nova informação é similar ao grânulo i^*, isto é, a distância obtida é menor que $\delta \cdot \mathcal{D}_{i^*}$,

$$\|\mathrm{x}_k - \mathcal{M}_{i^*}\|^2 \leq \delta \cdot \mathcal{D}_{i^*}. \tag{3.7}$$

Na duas situações as seguintes expressões são utilizadas para efetuar a atualização incremental do grânulo $\mathcal{G}_{i^*}$:

$$\begin{aligned}
\mathcal{M}_{i^*}^{t+1} &= \frac{\mathcal{N}_{i^*}^t}{\mathcal{N}_{i^*}^t + 1}\mathcal{M}_{i^*}^t + \frac{1}{\mathcal{N}_{i^*}^t}\mathrm{x}_k, \\
\mathcal{D}_{i^*}^{t+1} &= \frac{\mathcal{N}_{i^*}^t}{\mathcal{N}_{i^*}^t + 1}\mathcal{D}_{i^*}^t + \frac{1}{\mathcal{N}_{i^*}^t}\|\mathcal{M}_{i^*}^{t+1} - \mathrm{x}_k\|^2, \\
\mathcal{N}_{i^*}^{t+1} &= \mathcal{N}_{i^*}^t + 1,
\end{aligned} \tag{3.8}$$

em que $t+1$ indica o atributo após a atualização e t seu valor anterior.

Uma visão geral do método é dada no Algoritmo 1. E um exemplo ilustrativo é dado na Figura 7 considerando uma base de dados sintética com duas espirais entrelaçadas. Executando o eGAE obtêm-se o resultado apresentado na Figura 8, em que, percebe-se claramente que o uso de grânulos de informação ao longo do aprendizado (Figura 8a e Figura 8b) proporciona uma melhor representação latente e qualidade de reconstrução, do que quando considerando apenas a informação da janela atual de dados (Figura 8c e Figura 8d).

Algoritmo 1: *AutoEncoder Granular Evolutivo (eGAE).*

Entrada : $\mathrm{x}_t \,|\, t = 1, 2, \ldots, +\infty$.
Saída : $\mathrm{Z}_t, \hat{\mathrm{X}}_t$.
Inicialização
  Defina w, τ, δ, λ_1 e λ_2.
  $\mathrm{X}_0 \leftarrow \emptyset$.
  $\mathrm{G}_0 \leftarrow \emptyset$
  $g_0 \leftarrow 0$.
begin
  $\mathrm{X}_t \leftarrow \mathrm{X}_t \cup \mathrm{x}_t$.
  if X_t *possuir* w *amostras* **then**
    atualize o conjunto de grânulos chamando *granulação*(X_t).
    monte a matriz C_t contendos os centros $\mathcal{M}_{1,\ldots,g_t}$ do conjunto de grânulos atuais.
    atualize os pesos da rede utilizando X_t e C_t, com a função de perda (3.2), com taxa de aprendizado e quantidade de épocas apropriadas.
  end
  $\mathrm{Z}_t \leftarrow$ obtenha a representação latente com o Encoder.
  $\hat{\mathrm{X}}_t \leftarrow$ obtenha a saída reconstruída com o Decoder.
  return $\mathrm{Z}_t, \hat{\mathrm{X}}_t$.
end
function *granulação(*X*)*
  foreach x_k ***in*** X **do**
    if $\mathrm{G}_t == \emptyset$ **then**
      $\mathcal{G}_{g_t+1} \leftarrow$ crie um novo grânulo utilizando (3.4).
    else
      $i^* \leftarrow$ obtenha o grânulo mais similar a x_k utilizando (3.5).
      if i^* *deve ser atualizado de acordo com* (3.6) *ou* (3.7) **then**
        atualize-o com (3.8).
      else
        if *é necessário criar um novo grânulo de acordo com* (3.3) **then**
          $\mathcal{G}_{g_t+1} \leftarrow$ crie um novo grânulo utilizando (3.4).
        end
      end
    end
  end
end

3.2.1 Intuição na definição dos valores dos hiperparâmetros

A abordagem proposta possui um total de quatro novos hiperparâmetros : λ_1, λ_2, τ e δ; taxa de aprendizado, quantidade de épocas de treinamento e tamanho da janela de

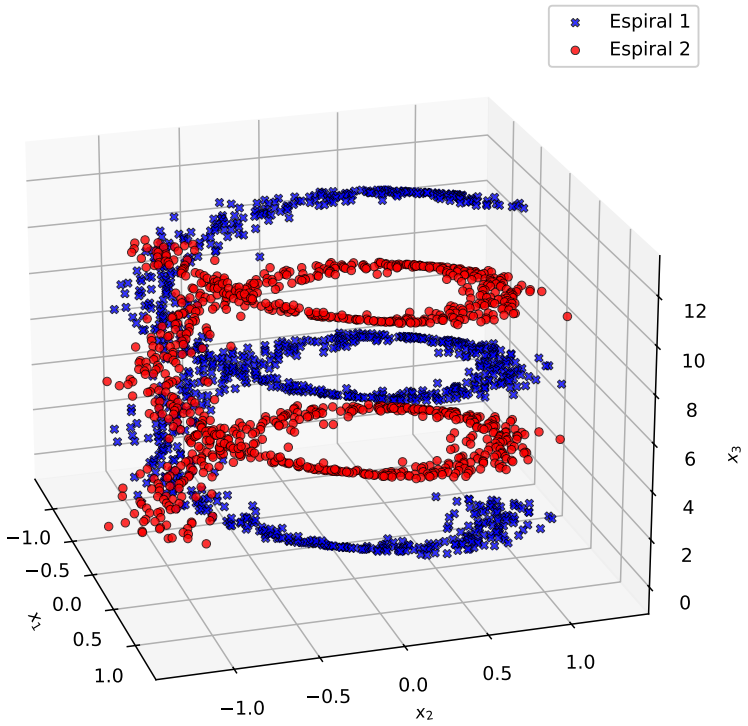


Figura 7 – Base de dados sintética com duas espirais 3D entrelaçadas.

dados, por exemplo, são comuns a qualquer rede neural, por isso foram desconsiderados nesse momento. Nesse sentido, esta subseção dedica-se a explicação da intuição que envolve a definição de cada um. Para tanto, utiliza-se a mesma base de dados apresentada na Figura 7 para ilustrar o comportamento da rede em termos da quantidade de grânulos e erro quadrático médio de reconstrução. Essas duas medidas foram escolhidas por serem a forma mais clara de observar a influencia desses hiperparâmetros. No contexto em que o hiperparâmetro não está sendo variado, utiliza-se por padrão os seguintes valores $w = 100$, $\tau = 5$, $\delta = 2.0$, taxa de aprendizado $= 0.01$ e quantidade de épocas igual a 50. A arquitetura utilizada foi, **Encoder**: `Linear (3, 512) -> ReLU() -> Linear(512, 3)`; e, **Decoder**: `Linear (2, 512) -> ReLU() -> Linear(512, 3)`, em que os valores numéricos em `Linear(·,·)` indicam a quantidade de entradas e saídas na respectiva camada.

3.2.1.1 Pesos da janela de dados atuais e dos grânulos

Os pesos associados ao aprendizado da janela de dados atual e dos grânulos, detonados respectivamente por λ_1 e λ_2, irão ser primordiais na capacidade de generalização do modelo. Cada um irá determinar a prioridade de aprendizado da rede, de forma que ao definir $\lambda_1 > \lambda_2$ indica a rede que a janela de dados atuais é mais importante, e com isso informações do passado podem ser esquecidas. Se o objetivo é fazer a rede aprender tudo aquilo que é observado, a prioridade for generalização do modelo, recomenda-se definir

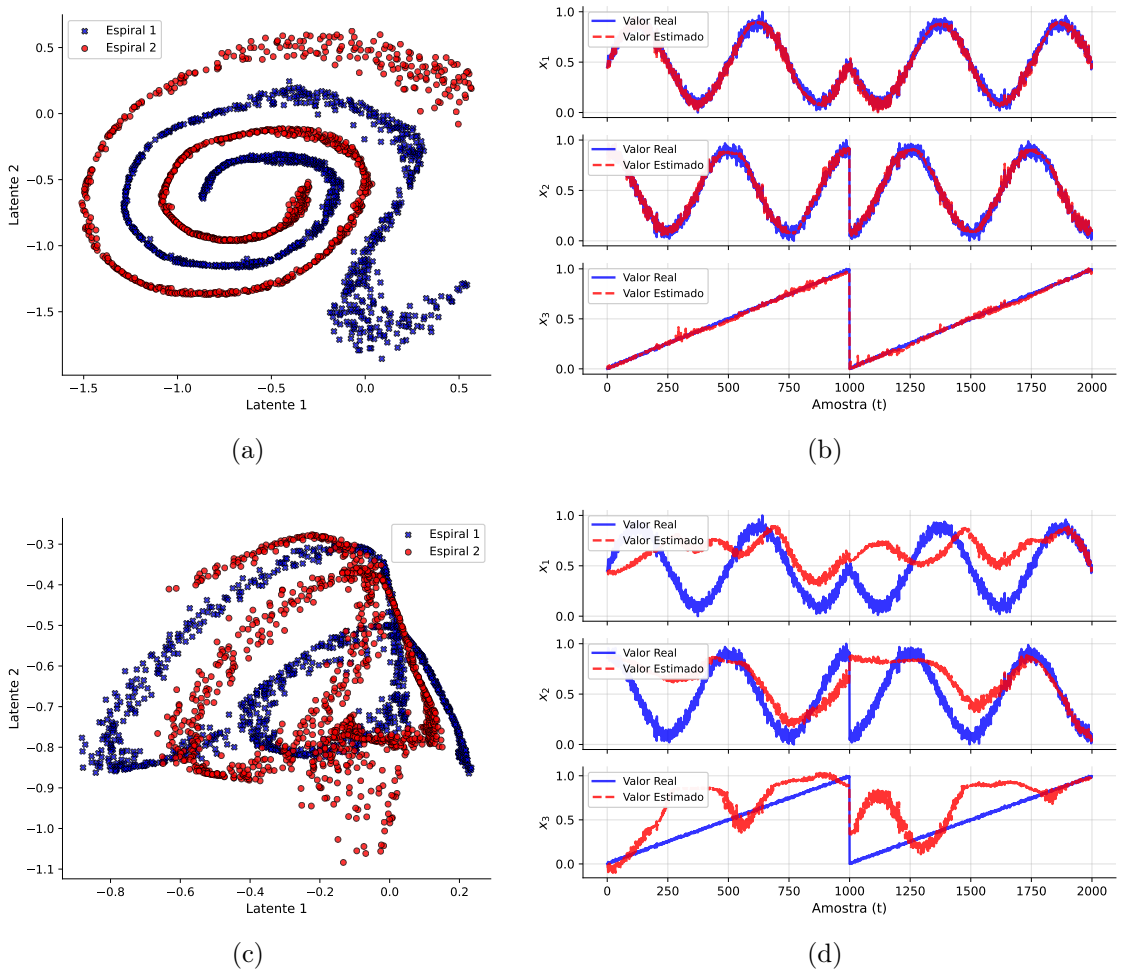


Figura 8 – Resultado do eGAE aplicado a base de dados tridimensional de duas espirais entrelaçadas, com redução de uma dimensão. Vale observar que o uso de grânulos de informação ao longo do aprendizado foi vital para generalização da rede, proporcionando uma a) melhor representação latente, e uma b) melhor reconstrução da entrada. A mesma rede, sem considerar grânulos informação, não consegue boa generalização; obtendo uma c) representação latente bem mais complexa, e, d) boa qualidade de reconstrução apenas na última janela de dados.

$\lambda_1 < \lambda_2$. Uma ilustração desse comportamento é apresentado na Figura 9, em termos do Erro Quadrático Médio (MSE). Na figura, importante observar que a medida que o valor λ_1 aumenta e λ_2 diminui, o MSE cresce, indicando piora na qualidade de reconstrução da rede, perda de generalização.

3.2.1.2 Medida mínima de cobertura e máxima de especificidade

Como apontado anteriormente, τ define a quantidade mínima de amostras que cada grânulo de informação deve possuir para só então criar novos, ou seja, a *medida mínima de cobertura*. Em contraste, δ define o raio máximo que um grânulo possui, indicando até que ponto o respectivo grânulo irá ser utilizado para representar amostras similares, indicando

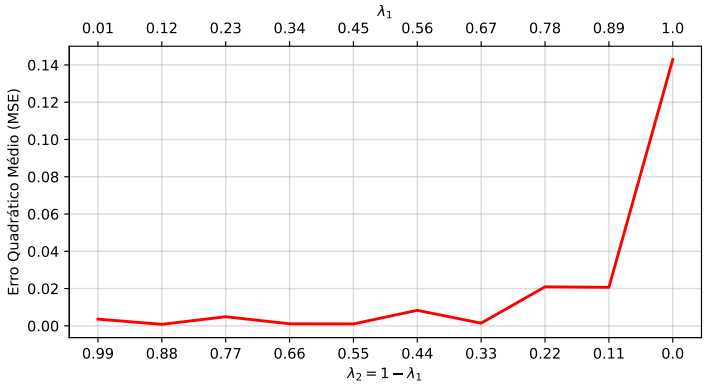
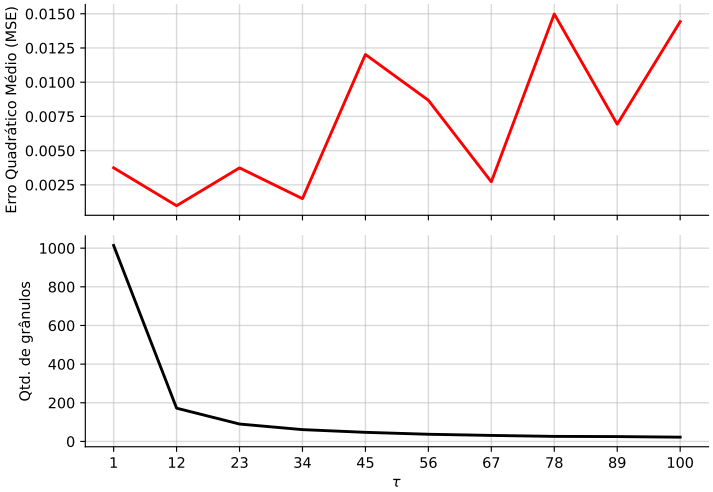


Figura 9 – Qualidade do aprendizado da rede ao variar λ_1 (eixo horizontal superior) e λ_2 (eixo horizontal inferior). Nesse exemplo $\lambda_2 = 1 - \lambda_1$.

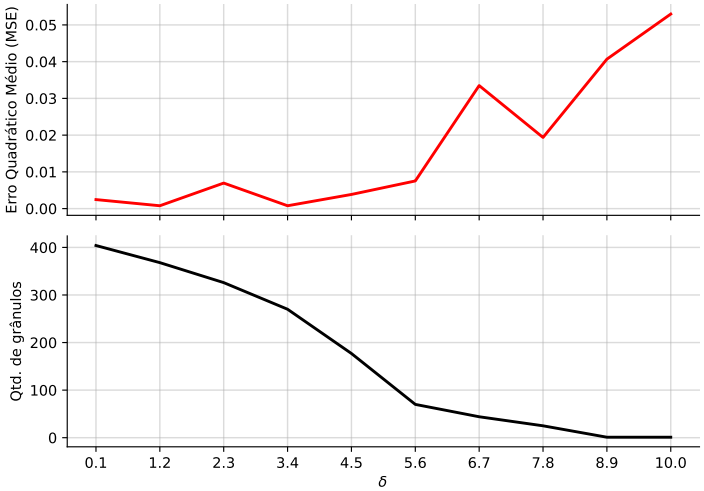
a *medida máxima de especificidade*. Logo, ambos influenciaram diretamente no tamanho do conjunto de grânulos, que por sua vez irá influência no aprendizado da rede. Diminuir a cobertura mínima e especificidade, irá refletir no aumento do conjunto de grânulos, e melhora no aprendizado da rede; porém, também aumentará o custo computacional da rede. Nesse sentido, definir valores que mantenham o conjunto de grânulos estável, é o recomendável. Na Figura 10a e Figura 10b é apresentado o comportamento da rede, em termos do MSE e quantidade de grânulos, ao varia τ e δ, respectivamente. Observa-se que quanto maiores os valores para τ e δ, menor a quantidade de grânulos, porém, pior é a qualidade da reconstrução.

A próxima subseção apresenta a proposta de um *framework* geral, e baseado no eGAE, para modelagem incremental. O *framework* utiliza o eGAE para detectar quando uma dada informação representa um novo comportamento baseado no resíduo de reconstrução, similar as abordagens apresentadas na Subseção 3.1.1. O diferencial da abordagem proposta está no uso do eGAE na etapa de reconstrução, que evita retreinamentos manuais ao surgirem novos eventos, e pelo uso de método evolutivos na identificação dos eventos.

Vale ressaltar ainda, que uma outra abordagem para modelagem baseado no eGAE, além da baseada em resíduos, seria o processamento da informação latente. Contudo, trabalhar com esse espaço impõe um grande desafio, o de ajustar constantemente a modelagem dos métodos de clusterização online para refletir na mudança, também constante, que ocorre no espaço. Para conseguir representar tudo aquilo que foi observado, em seu processo de aprendizado, a rede realiza mudanças constante na representação da informação latente, com isso, a modelagem realizada pelos métodos de clusterização, sempre estaria desatualizada em relação a representação atual de rede. Um exemplo desse comportamento é apresentado na Figura 11, em que é possível observar o espaço latente, dos dados das espirais entrelaçadas, ao longo do aprendizado. Para cada instante de tempo observado, o espaço latente se mostra diferente do instante anterior. Devido a essa dificuldade, essa metodologia não foi explorada nessa tese, listando-a como trabalho futuro.



(a)



(b)

Figura 10 – Qualidade do aprendizado e quantidade de grânulos na rede ao variar (a) τ e (b) δ.

3.3 Modelagem Baseada em Resíduos

Nesta seção será apresentada a proposta de um *framework* que utiliza métodos evolutivos que são capazes de gerar estimativas dos dados de entrada, possibilitando o cálculo de resíduos de reconstrução. A premissa base do *framework* é similar a adotada no processo de inferência da SBM: *se uma dada entrada* x *é relacionada a algum um padrão de dados já conhecido pelo método, então pode ser aproximada com pequeno resíduo de reconstrução*. Com isso, é possível decidir com base no resíduo, se uma nova amostra representa ou não um novo padrão de dados. Para simplificar futuras referências, este *framework* será nomeado de eRBM (do inglês, *Evolving Residual-Based Modeling*).

Para tanto, considere que $\mathrm{X} \in \mathbb{R}^{w \times n}$ representa um conjunto com w leituras/amostras do funcionamento normal de um sistema real; considere também que $\mathrm{x}_t \in \mathbb{R}^n$

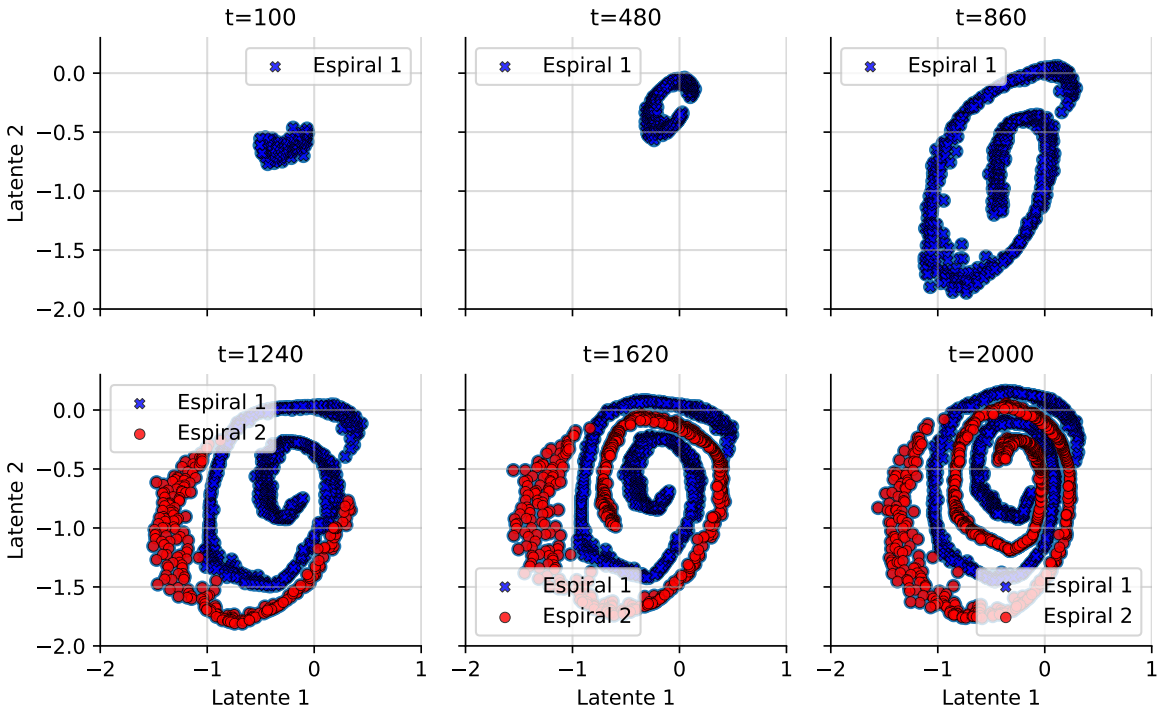


Figura 11 – Espaço latente dos dados das duas espirais entrelaçadas em diferentes momentos do aprendizado com eGAE.

representa uma leitura/amostra da situação atual, em que t indexa tal instante. Nessa abordagem, X é chamado de *evidências empíricas iniciais* e é utilizado para ensinar ao método, antes de utilizá-lo para modelar o sistema efetivamente, como é o comportamento usual do sistema observado até o momento.

Dito isso, a abordagem proposta caracteriza-se por utilizar essa evidência empírica inicial para discriminar x_t, isto é, decidir quando uma nova entrada representa uma novidade (um novo evento), de forma a expandir a base de conhecimento ou a estrutura interna do método. Além disso, a abordagem também utiliza x_t, quando julgar que deve, para refinar a base de conhecimento.

Em termos práticos, antes da abordagem iniciar o processo de aprendizado ou treinamento, X é apresentado ao método, de forma que todos os modelos locais criados internamente irão representar o comportamento normal. O termo *modelo local* no contexto do eRBM faz referência a parte(s) da estrutura interna do método evolutivo, criada(s) especificamente para caracterizar o respectivo comportamento do sistema. Por exemplo, para o caso de um método evolutivo granular, cada grânulo é dito ser um modelo local.

Após essa etapa inicial de aprendizado, o método inicia o processamento efetivamente. Nessa situação, x_t poderá ser utilizado para ajustar os parâmetros dos modelos existentes ou para criar novos. Essa última ação é realizada quando uma certa porção de amostras de um passado recente difere do conhecimento armazenado pelos modelos

existentes. Vale ressaltar que por meio desse procedimento, a abordagem por si só rotula x_t em *normal* ou *novo evento*, apenas. Contudo, o especialista pode (e idealmente deve) alterar esses rótulos para termos mais próximos do significado real segundo o domínio do problema e a intenção de utilização do modelo. Uma visão geral do *framework* é dada na Figura 12. Como observado na figura, na arquitetura do eRBM, o método evolutivo deve obrigatoriamente fornecer como saída do processo de inferência a estimativa para a entrada, $\hat{x}_t$, e o identificador interno do modelo ao qual a entrada foi associada, denotado por c_t^*. A estimativa $\hat{x}_t$ é utilizada para calcular a quantidade de resíduo, e o identificador c_t^* para, na situação em que o resíduo está dentro dos limites aceitáveis, atualizar o respectivo modelo.

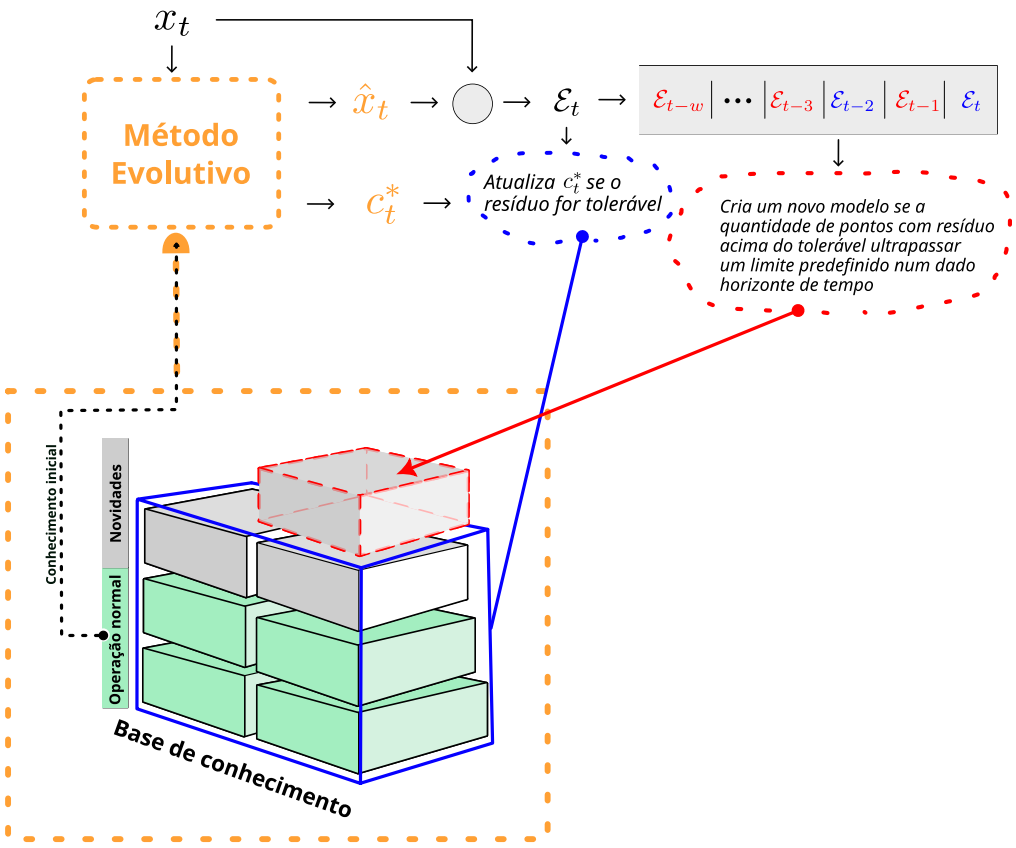


Figura 12 – Visão geral do *framework* eRBM.

Mecanismo de aprendizado

Considerando que o eRBM já foi devidamente inicializado com o *conhecimento inicial*, o passo seguinte consiste em determinar o limiar de resíduo para esse conjunto de modelos iniciais. As seguintes expressões são utilizadas para essa tarefa:

$$\mathcal{E}_{\mathrm{x},\hat{\mathrm{x}}} = \sum_{j=1}^{n} \left(\mathrm{x}^j - \hat{\mathrm{x}}^j\right)^2, \tag{3.9}$$

$$\mathcal{E}_e^m = \frac{1}{w} \sum_{\mathrm{x}\in\mathrm{X}} \mathcal{E}_{\mathrm{x},\hat{\mathrm{x}}}, \tag{3.10}$$

$$\mathcal{E}_e^v = \frac{1}{w} \sum_{\mathrm{x}\in\mathrm{X}} (\mathcal{E}_i^m - \mathcal{E}_{\mathrm{x},\hat{\mathrm{x}}})^2, \tag{3.11}$$

em que $\mathcal{E}_e^m$ e $\mathcal{E}_e^v$ representa a média e variância de resíduos do conhecimento inicial do método, respectivamente; e, $e \in \mathbb{N}$ denota a quantidade de conjuntos de modelos, ou estados, adicionados para representar cada comportamento; o estado inicial do método é denotado com $e = 0$. Na Figura 12 por exemplo, $e = 0$ faz referência aos quatros modelos que representam a operação normal, logo, um estado relaciona o conjunto de modelos adicionados a estrutura (base de conhecimento) em cada alteração, seja um único ou conjuntos de modelos.

Como dito, em tempo de execução, espera-se que o método evolutivo forneça como saída a aproximação da entrada, denotada por $\hat{\mathrm{x}}_t$, e o modelo a qual a entrada foi internamente associada, denotado por c_t^*; com essas informações é julgado se a nova entrada é novidade ou não de acordo com os limiares definidos. Primeiramente é calculado o resíduo entre x_t e $\hat{\mathrm{x}}_t$ utilizando (3.9), após, utiliza-se a seguinte expressão para efetuar o julgamento:

$$\begin{cases} \textit{é novidade}, & \mathcal{E}_{x_t,\hat{x}_t} > \mathcal{P}\left(\mathcal{E}_i^m + \mathcal{E}_i^v\right) \\ \textit{não é novidade}, & \textit{c.c.} \end{cases}, \tag{3.12}$$

em que $i \in \{1, 2, \ldots, e\}$ denota o estado a qual a entrada foi associada, e $\mathcal{P} \in \mathbb{R}$ é uma constante predefinida que define o percentual (no intervalo [0, 1]) de resíduo tolerável com base no conhecimento inicial. Na situação em que x_t não é considerado novidade, o mesmo é utilizado para atualizar o modelo c_t^*; caso seja, verifica-se se o percentual de novidades em um horizonte passado de $w \in \mathbb{N}$ amostas ultrapassa um limiar $v \in (0, 1]$, ambos definidos previamente. Caso ultrapasse, as amostras fora do limite são fornecidas ao método evolutivo, a qual deve criar os modelos para representar essa nova informação e retornar seus identificadores, para que assim o eRBM possa criar um novo estado associando os modelos retornados.

É possível ainda, na situação em que x_t não é novidade, atualizar os limiares do respectivo estado por meios das seguintes expressões:

$$[\mathcal{E}_i^m]^{t+1} = \frac{s_i}{s_i + 1}[\mathcal{E}_i^m]^t + \frac{1}{s_i + 1}\mathcal{E}_{\mathrm{x}_t,\hat{\mathrm{x}}_t}, \tag{3.13}$$

$$[\mathcal{E}_i^v]^{t+1} = \frac{s_i}{s_i+1}[\mathcal{E}_i^v]^t + \frac{1}{s_i+1}(\mathcal{E}_{\mathrm{x}_t,\hat{\mathrm{x}}_t} - [\mathcal{E}_i^m]^t)^2, \tag{3.14}$$

$$s_i^{t+1} = s_i^t + 1, \tag{3.15}$$

em que $s_i \in \mathbb{N}$ denota a quantidade de amostras associadas ao respectivo estado até o instante atual.

3.4 Resumo do Capítulo

Este capítulo foi dedicado a apresentar os principais conceitos relacionados a AutoEncoders, bem como os principais trabalhos científicos que buscaram, utilizando-os, contribuir para a modelagem de sistemas não estacionários.

Os principais fundamentos sobre granulação de informação também foram apresentados como introdução ao AutoEncoder Granular Evolutivo (eGAE) proposto, que utiliza grânulos de informação para criar uma base conhecimento do passado, que por sua vez é utilizada em seu processo de aprendizado contínuo e incremental. Um *framework* geral que realiza a modelagem com base na informação residual gerada por métodos evolutivos de reconstrução também foi proposto.

Capítulo 4

Modelagem de Eventos Baseada em Clusterização Evolutiva

Clusterizar dados, e de forma adequada, baseando-se apenas em uma medida de similaridade não é uma tarefa trivial. Sendo ainda um dos principais tópicos de pesquisa. Por exemplo, o método K-Means, um dos métodos de maior sucesso nessa tarefa, foi originalmente proposto na década de 1960 [MacQueen, 1967]; recentemente um grupo de pesquisadores propuseram uma nova versão[Rezaee et al., 2021].

Os desafios que envolvem a clusterização de dados residem principalmente em definir medidas de distância apropriadas para extrair informação do domínio; em definir modelos com estruturas que aproximem fidedignamente a relação dos dados no espaço de entrada; lidar com dados de alta dimensionalidade; e, mais recentemente, em desenvolver mecanismos de aprendizado evolutivo para agrupar os dados de forma online e incremental, e assim conseguir lidar com a grande quantidade de dados produzidas pelos sistemas modernos. Neste caso, métodos com aprendizado offline são inviáveis [Batool and Abbas, 2021].

Nesse sentido, nesta tese métodos de clusterização evolutiva são utilizados para realizar a modelagem evolutiva de sistemas não estacionários. A modelagem adotada consiste em associar um cluster ou conjuntos de clusters a cada comportamento do sistema, assim, cada comportamento pode ser facilmente detectado e identificado pela associação aos respectivos clusters criados.

O restante do capítulo, inicialmente apresenta uma revisão de literatura contendo os principais métodos e alguns dos mais recentes. Em seguida, é apresentada a proposta de uma nova abordagem para clusterização online, em que os clusters assumem formato arbitrário, seguindo a dispersão natural dos dados, sem utilizar mecanismos existentes na literatura, como composição de micro-clusters. Tal abordagem, é baseada em uma técnica largamente utilizada na indústria, a Modelagem Baseada em Similaridade de Modelos (SBM). Com isso, antes da apresentação da proposta, o método SBM é detalhadamente

explorado.

4.1 Revisão de Clusterização Evolutiva

Um dos métodos de clusterização mais bem-sucedidos na literatura é o CluStream [Aggarwal et al., 2003], que consiste em dois componentes. O primeiro componente é caracterizado por um procedimento de micro-aglomeração online que armazena periodicamente estatísticas resumidas detalhadas. O segundo é um procedimento offline que utiliza o algoritmo K-Means para organizar os micro-clusters armazenados em macro-clusters. Seu desempenho depende fortemente dos parâmetros definidos pelo usuário (por exemplo, raio do micro-cluster, horizonte temporal) e requer a especificação prévia do número de macro-clusters a serem usados pelo algoritmo K-Means.

O DenStream [Cao et al., 2006] também tem sido amplamente utilizado na literatura e é uma versão aprimorada do DBSCAN projetada para funcionar em ambientes online. O método funciona de forma semelhante ao CluStream, com um componente online para criar micro-clusters e um componente offline para definir os macro-clusters. O componente online utiliza uma janela de amortecimento para criar três tipos de micro-clusters: *core-micro-cluster*, *potential-micro-cluster* e *outlier-micro-cluster*, que resumem o comportamento do fluxo de dados. Os macro-clusters são definidos offline e apenas quando solicitado para predição, organizando os *core-micro-clusters* com DBSCAN. Os outros dois tipos de micro-clusters são usados para reter informações recentes que podem se tornar um novo macro-cluster. O uso da composição dos *core-micro-clusters* permite a formação de clusters com formas arbitrárias, abordando um problema em aberto importante na área.

O DBStream [Hahsler and Bolaos, 2016] foi proposto para resolver um problema que o DenStream e outras abordagens similares não haviam abordado na época: a densidade na região comum entre micro-clusters. Similar às abordagens anteriores, ele adota uma fase online e uma fase offline. Na fase online, o método identifica todos os micro-clusters nos quais a nova amostra se encontra dentro de um raio predefinido. Se nenhum micro-cluster for encontrado, um novo é criado com a nova amostra; em seguida, um gráfico de densidade compartilhada é atualizado com as informações de densidade. A fase offline envolve a exploração das informações de densidade no gráfico para definir os macro-clusters com base em limiares definidos pelo usuário e uma variante do DBSCAN.

Inspirado tanto no DBSCAN quanto no *Self Organizing Maps* (SOM) (também conhecido como rede de Kohonen) [Kohonen, 1982], o SOStream [Isaksson et al., 2012] foi proposto. O método utiliza o conceito de micro-clusters do DBSCAN combinado com aprendizado competitivo do SOM para capturar a densidade dos dados e definir automaticamente um limiar de densidade baseado na ideia de construir vizinhanças com um número mínimo de pontos. No entanto, o método não especifica um mecanismo para

definir os macro-clusters.

O CEDAS [Hyde et al., 2016] também emprega o conceito de micro-clusters, macro-clusters e aprendizado em duas etapas semelhante ao CluStream, DenStream e DBStream. No entanto, ao contrário dessas abordagens, ele funciona inteiramente online. Na primeira etapa, o método cria micro-clusters hiperesféricos, e na segunda etapa, esses micro-clusters são organizados em macro-clusters usando uma estrutura de grafo. O método é computacionalmente eficiente e, dependendo dos parâmetros definidos pelo usuário, pode ser preciso e robusto a ruídos. No entanto, o método não leva em conta informações de densidade.

O MicroTEDAClus [Maia et al., 2020] foi proposto com base no conceito de Typicality and Eccentricity Data Analysis (TEDA) [Angelov, 2014] e na composição de micro-clusters do DBSCAN. Essa abordagem é totalmente online, pode definir clusters com formas arbitrárias e demonstrou bom desempenho em experimentos, superando o estado da arte em alguns casos. O mecanismo que define os macro-clusters baseia-se em um limiar mínimo de distância entre micro-clusters determinado automaticamente.

MacroSOStream Oliveira et al. [2022] é uma extensão do SOStream e aborda o problema de que o método original não possui um mecanismo para organizar microclusters em macroclusters. Além disso, o método inclui um mecanismo de fusão de macroclusters, que tem como objetivo melhorar o desempenho e a robustez contra dados ruidosos, embora ainda seja fortemente dependente de parâmetros definidos pelo usuário.

Recentemente, o MCMSTStream [Erdinç et al., 2024] foi proposto, também baseado no uso de micro-clusters para criar clusters de formas arbitrárias. Utilizando uma janela deslizante de tamanho definido pelo usuário, o algoritmo processa os dados dessa janela utilizando uma KD-Tree e cria um micro-cluster sempre que uma vizinhança mínima surge em uma região não mapeada dos dados. O método considera um conjunto de amostras como vizinhas se estiverem dentro de um limiar de distância predefinido. Em seguida, qualquer subconjunto de micro-clusters sobrepostos é combinado em um macro-cluster utilizando um Minimum Spanning Tree (MST).

Também baseadas no conceito de micro- e macro-clusters, podemos mencionar as seguintes abordagens recentes: HCMstream [Laohakiat et al., 2016], VHEC [Wattanakitrungroj et al., 2017], BEstream [Wattanakitrungroj et al., 2018], ACSC [Fahy et al., 2019], DyClee [Roa et al., 2019], StreamSW Reddy and Bindu [2019] e DGStream [Ahmed et al., 2020]. Não as descreveremos aqui devido a limitações de espaço.

Além do conceito de micro- e macro-clusters, podemos mencionar o AutoCloud Bezerra et al. [2020], que utiliza o framework TEDA de forma semelhante ao MicroTEDAClus. No entanto, ele define seus clusters como hiperesferas que são continuamente atualizadas de acordo com o comportamento do fluxo de dados. A principal vantagem dessa

abordagem é sua baixa exigência computacional. Outra abordagem, chamada evolving Vector Quantization for Arbitrary ellipsoids with Merge-and-Split functionality (eVQ-AMS) [Lughofer and Sayed-Mouchaweh, 2015], utiliza clusters elipsoidais com rotação arbitrária para realizar a clusterização online com uma capacidade de "plug-and-play". Isso significa que o método pode ajustar automaticamente sua estrutura para seguir a dispersão natural dos dados, compensando valores inadequados de parâmetros definidos pelo usuário. O recurso "plug-and-play" permite que a abordagem funcione em diferentes fluxos de dados utilizando os mesmos valores de parâmetros.

O método baseado em probabilidades chamado Online Ellipsoidal Clustering (OEC) [Moshtaghi et al., 2016b] utiliza modelos gaussianos multivariados para representar clusters como hiperelipsoides. A abordagem atualiza continuamente um modelo gaussiano chamado *Tracker*, que possui um fator de esquecimento, lembrando-se apenas de um certo horizonte do passado. Quando o *Tracker* se separa completamente dos clusters existentes, assume-se que um novo cluster é necessário para representar o comportamento atual. O novo cluster é criado a partir de um *buffer* de amostras de anomalias armazenadas desde o início da divergência.

Baseado em modelos gaussianos e conjuntos fuzzy, Lemos et al. [2010] propõem um método evolutivo direcionado a detecção e diagnóstico de falhas em sistemas dinâmicos. A abordagem utiliza modelos gaussianos multivariáveis que são atualizado incrementalmente, permitindo a adaptação contínua do método a novas condições operacionais sem necessidade de reconfiguração manual. Validado experimentalmente em máquinas de indução, o método demonstrou eficácia na identificação de padrões anômalos e na evolução do conhecimento do sistema ao longo do tempo. Os resultados sugerem que a abordagem proposta é promissora para aplicações em aprendizado de máquina incremental, especialmente em ambientes de fluxo contínuo de dados, onde novas falhas podem emergir dinamicamente.

Esses métodos estão entre as abordagens mais bem-sucedidas e recentes, e alguns deles serão utilizados como *baseline* nos experimentos. Existem várias outras abordagens na literatura baseadas em diferentes conceitos. Para uma revisão abrangente do estado da arte em clusterização evolutiva, veja Zubaroğlu and Atalay [2020], Batool and Abbas [2021], Al-Amri et al. [2021], Ezugwu et al. [2022], Ghani et al. [2023b].

4.2 Modelagem Baseada em Similaridade de Modelos

A Modelagem Baseada em Similaridade de Modelos (SBM) (do inglês, *Similarity-Based Modeling*) é uma técnica não-paramétrica largamente utilizada em aplicações industriais para detecção de anomalia Singer et al. [1997], Wegerich et al. [2003], Wegerich [2004], Monkman and Wegerich [2005], Tobar et al. [2011], Perez et al. [2018, 2019]. Seu sucesso se deve ao seu processo de inferência simples, que envolve utilizar um conjunto de

dados históricos contendo $l \in \mathbb{N}$ amostras referente a condição normal do sistema, para computar uma estimativa $\hat{\mathrm{x}} \in \mathbb{R}^n$ para uma dada entrada $\mathrm{x} \in \mathbb{R}^n$. Utiliza-se então a erro de aproximação entre x e $\hat{\mathrm{x}}$ para determinar se a nova amostra é uma anomalia ou não.

Na terminologia da abordagem, o conjunto de dados históricos é chamado de *matriz memória*, denotado como $\mathrm{D} \in \mathbb{R}^{l \times n}$:

$$\mathrm{D} = \begin{bmatrix} \mathrm{x}^{11} & \mathrm{x}^{12} & \dots & \mathrm{x}^{1n} \\ \mathrm{x}^{21} & \mathrm{x}^{22} & \dots & \mathrm{x}^{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{x}^{l1} & \mathrm{x}^{l2} & \dots & \mathrm{x}^{ln} \end{bmatrix}, \tag{4.1}$$

e essas amostras são chamadas de *amostras representativas*. Estes termos são baseados na premissa fundamental da técnica: *se* x *é similar as amostras em* D*, então pode ser aproximada com pequeno erro* Singer et al. [1997]. Em outras palavras, isso significa que o comportamento normal do sistema é caracterizado somente por D, sendo então a "memória" da técnica relacionada a esse comportamento. E desde que D possui um tamanho limitado, as amostras armazenadas devem ser tão "representativa" (i.e., não-redundantes) quanto possível tendo em vista a maximização da performance do método e minimização de custo computacional. Por fim, se uma nova amostra pertence ao comportamento normal, essa será similar as representativas, e então apresentará um baixo erro de aproximação.

Formalmente, dada uma nova amostra $\mathrm{x}_t \in \mathbb{R}^n$ de um sistema dinâmico (e.g., processo industrial, dados climáticos, consumo energético, dentre outros), com t denotando um índice de tempo discreto, para computar uma aproximação baseada em D, uma possibilidade é através de uma combinação linear dessas amostras representativas Singer et al. [1997]:

$$\hat{\mathrm{x}}_t = \mathrm{D}^\top w_t, \tag{4.2}$$

em que $\top$ denota a matriz transposta e $w_t \in \mathbb{R}^l$ contém os pesos associados a cada amostra representativa na combinação linear. Desta forma, para encontrar os pesos que melhor aproximam x_t, pode-se definir a norma Euclideana entre x_t e $\hat{\mathrm{x}}_t$,

$$\begin{aligned} \|\mathrm{x}_t - \hat{\mathrm{x}}_t\|^2 &= \left\|\mathrm{x}_t - \mathrm{D}^\top w_t\right\|^2 \\ &= (\mathrm{x}_t - \mathrm{D}^\top w_t)^\top (\mathrm{x}_t - \mathrm{D}^\top w_t) \\ &= \mathrm{x}_t^\top \mathrm{x}_t - 2\mathrm{x}_t^\top \mathrm{D}^\top w_t + w_t^\top \mathrm{D}\mathrm{D}^\top w_t. \end{aligned} \tag{4.3}$$

Os pesos que minimizam (4.3) são dados a seguir:

$$w_t = (\mathrm{DD}^\top)^{-1}\mathrm{Dx}_t. \tag{4.4}$$

Contudo, (4.4) possui inúmeras limitações numéricas e práticas Singer et al. [1997]. A principal limitação prática é que $\hat{\mathrm{x}}_t$ é sempre igual a x_t, mesmo quando x_t é altamente dissimilar as amostras representativas.

Demonstração. Se x_t tiver sido aproximado com erro nulo, então:

$$(\mathrm{x}_t - \hat{\mathrm{x}}_t)^\top = [0, 0, \ldots, 0]^\top; \tag{4.5}$$

expandindo e rearranjando,

$$\begin{aligned} (\mathrm{x}_t - \hat{\mathrm{x}}_t)^\top &= [0, 0, \ldots, 0]^\top \\ \left(\mathrm{x}_t - \mathrm{D}^\top \left(\mathrm{DD}^\top\right)^{-1} \mathrm{Dx}_t\right) &= [0, 0, \ldots, 0]^\top \\ \mathrm{x}_t &= \mathrm{D}^\top \left(\mathrm{DD}^\top\right)^{-1} \mathrm{Dx}_t. \end{aligned} \tag{4.6}$$

Multiplicando (4.6) por D, obtêm-se:

$$\mathrm{Dx}_t = \mathrm{DD}^\top \left(\mathrm{DD}^\top\right)^{-1} \mathrm{Dx}_t, \tag{4.7}$$

e pela propriedade de matriz inversa tem-se que $\mathrm{AA}^{-1} = \mathrm{I}$, em que A é uma matriz quadrada arbitrária e I é uma matriz identidade. Definindo $\mathrm{A} = \mathrm{DD}^\top$ em (4.7), obtêm-se:

$$\begin{aligned} \mathrm{Dx}_t &= \mathrm{AA}^{-1}\mathrm{Dx}_t \\ \mathrm{Dx}_t &= \mathrm{IDx}_t \\ \mathrm{Dx}_t &= \mathrm{Dx}_t, \end{aligned} \tag{4.8}$$

o que conclui a prova.

□

Para contornar esse problema e alguns outros numéricos, uma operação de similaridade é proposta por Singer et al. [1997] para substituir o produto interno em (4.4). Com esta mudança, uma regularização implícita é adicionada aos pesos, prevenindo boas aproximações de amostras dissimilares as amostras representativas. Desta forma, os pesos "ótimos regularizados", denotados como w_t^r, são definidos a seguir Singer et al. [1997]:

$$w_t^r = (\mathrm{D} \circ \mathrm{D}^\top)^{-1}(\mathrm{D} \circ \mathrm{x}_t) = \mathrm{G}^{-1}\mathrm{a}_t, \tag{4.9}$$

em que $\circ$ representa uma operação de similaridade que usa uma função de similaridade $s : \mathbb{R}^n \times \mathbb{R}^n \to \mathcal{U}$, com $\mathcal{U} = [0, 1] \subset \mathbb{R}$; $\mathrm{G} = \mathrm{D} \circ \mathrm{D}^\top \in \mathcal{U}^{l \times l}$ é chamada de matriz de intra-similaridade e armazena a similaridade entre cada possível par de amostra representativa, e $\mathrm{a}_t = \mathrm{D} \circ \mathrm{x}_t \in \mathcal{U}^l$ armazena a similaridade entre x_t e cada amostra representativa. Vale ressaltar que com o uso da operação de similaridade, a aproximação agora pode ser tanto linear quanto não-linear, dependendo apenas da função de similaridade s Singer et al. [1997]. Um exemplo de função de similaridade considerada na formulação clássica da SBM é definida a seguir [Marins et al., 2018] (consulte Marins et al. [2018] para outras):

$$s(\mathrm{x}_k, \mathrm{x}_j) = \frac{1}{1 + \|\mathrm{x}_k - \mathrm{x}_j\|}, \tag{4.10}$$

com $\|\cdot\|$ denotando a norma Euclideana.

Usando os pesos regularizados em (4.9), a aproximação em (4.2) pode ser reescrita como:

$$\hat{\mathrm{x}}_t = \mathrm{D}^\top w_t^r. \tag{4.11}$$

No entanto, em situações em que a função de similaridade é dependente da escala dos dados, como em (4.10), é crucial restringir as estimativas para a vizinhança das amostras representativas por meio da normalização dos pesos [Marins et al., 2018]. Logo, a expressão final para estes tipos de funções de similaridade é definida a seguir:

$$\hat{\mathrm{x}}_t = \mathrm{D}^\top \frac{w_t^r}{|w_t^r|}, \tag{4.12}$$

em que $|\cdot|$ denota a norma L1 [Rolewicz, 1987], também conhecida como distância de Manhattan.

Em Singer et al. [1997], o *Sequential Probability Ratio Test* (SPRT) [Wald, 2004] foi utilizado para estabelecer um limiar e em seguida avaliar se x_t é uma anomalia baseada no erro de aproximação. Como alternativa, uma avaliação de similaridade entre x_t e $\hat{\mathrm{x}}_t$ também pode ser empregada. Nessa situação, um limiar pode ser definido simplesmente como um escalar no intervalo $(0, 1)$, tornado a questão em uma simples definição ótima de hiperparâmetro.

Uma extensão para a SBM foi proposta para problemas de classificação multi-classe por Marins et al. [2018]. A extensão, nomeada M-SBM, generaliza a SBM caracterizando cada classe com seu próprio conjunto de amostras representativas, ao contrário da SBM que armazena apenas amostras representativas da classe normal. O processo de inferência agora consiste em gerar uma aproximação considerando cada classe e atribuindo a entrada

a classe com o menor erro de aproximação. Esta versão é a base da nova abordagem proposta, ao qual é detalhada na próxima seção.

4.2.1 Modelagem baseada em similaridade de modelos multiclasse

A Modelagem Baseada em Similaridade de Modelos Multiclasse (M-SBM) [Marins et al., 2018] (do inglês, *Multiclass Similarity-Based Modeling*) é uma extensão da SBM clássica para lidar com problemas de multiclasse. Nesta abordagem, cada classe i é representada por sua própria matriz memória D_i. O procedimento de inferência permanece o mesmo utilizado na SBM original, mas é realizado separadamente para cada classe: dada uma nova amostra x_t, uma estimativa $\hat{\mathrm{x}}_{ti}$ é computada usando a matriz memória correspondente D_i,

$$\hat{\mathrm{x}}_{ti} = \mathrm{D}_i^{\top} w_{ti}^{r}, \tag{4.13}$$

com $i = 1, 2, \ldots, c$; em que c é o número total de classes sendo avaliadas; então, x_t é atribuída a classe com o menor erro de aproximação, denotada como c_t^*,

$$c_t^* = \underset{i \in \{1,2,\ldots,c\}}{\operatorname{argmax}} \; \boldsymbol{s}(\mathrm{x}_t, \hat{\mathrm{x}}_{ti}), \tag{4.14}$$

em que $\boldsymbol{s}(\cdot, \cdot)$ representa uma função de similaridade.

Um exemplo intuitivo da efetividade desta abordagem é dado na Figura 13. As figuras na parte superior apresentam o processo de *treinamento*, que envolve selecionar as amostras representativas do conjunto de dados históricos. As figuras na parte inferior, apresentam o processo de *inferência* para uma dada amostra de entrada e a região de decisão de cada classe obtida usando a técnica. Baseada na região de decisão obtida (Figura 13d), é possível observar a capacidade da técnica em definir fronteiras não-lineares.

Como pôde ser visto, a M-SBM é uma abordagem simples, intuitiva e robusta. O maior desafio reside em seu treinamento, que envolve duas fases: **a)** selecionar as amostras representativas de cada classe e, **b)** definir os parâmetros da função de similaridade (quando houver). Em geral, a primeira fase é mais difícil, que usualmente é realizada manualmente ou usando algum procedimento iterativo parametrizado que opera pela definição de uma distância mínima entre todas as amostras, para então descartar as classificadas como redundantes. Um procedimento proposto em Marins et al. [2018] com esse propósito é apresentado na próxima subseção.

4.2.1.1 Seleção de amostras representativas

Selecionar quais amostras irão compor a matriz memória é vital na SBM. Incorporar grandes quantidades de amostras distintas nessa matriz eleva substancialmente a

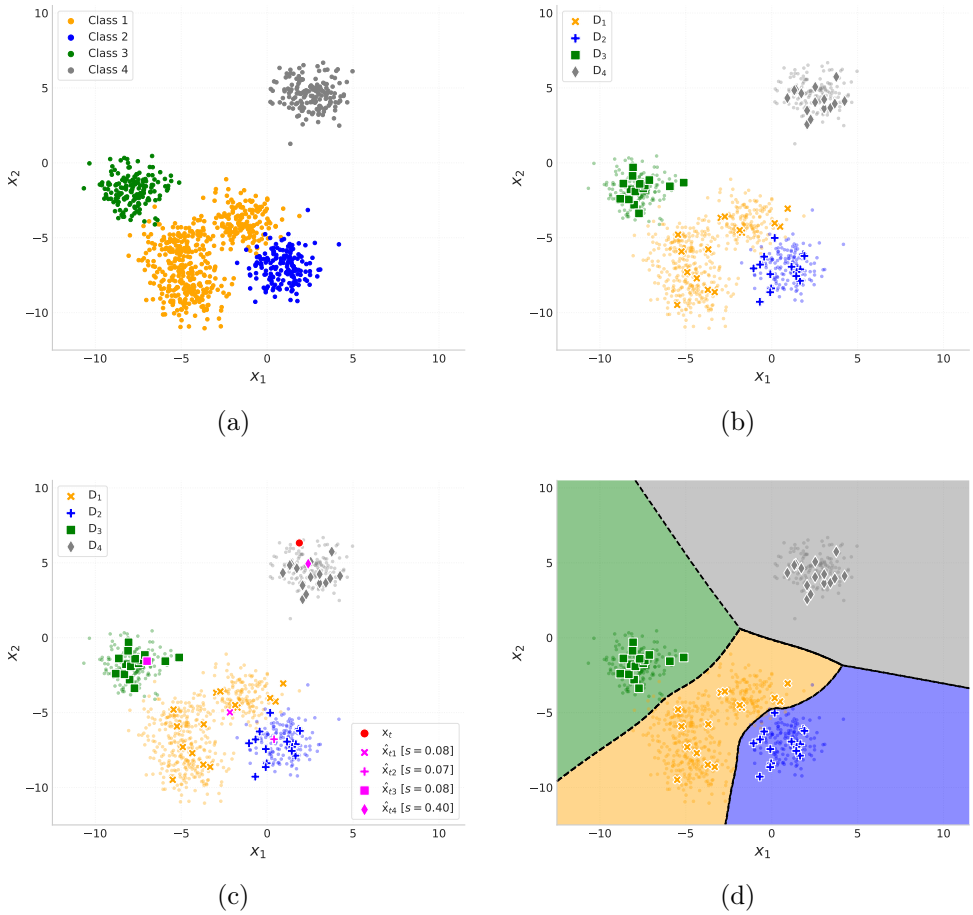


Figura 13 – Exemplo ilustrativo da M-SBM. (a) A partir de um conjunto de dados históricos de cada classe, (b) Um conjunto (15 neste exemplo) de amostras representativas é selecionado por classe. (c) Quando uma nova entrada é dada, uma estimativa é computada considerando casa classe, e então a classe mais similar é atribuída baseada na similaridade das estimativas ($c_t^* = 4$ neste exemplo). (d) Embora seja uma técnica simples, possui alta capacidade de modelagem não-linear.

qualidade da modelagem, porém, também eleva o custo computacional. Em contrapartida, definir um conjunto pequeno diminui o esforço computacional, mas pode comprometer a performance da abordagem. Logo, é importante definir a matriz de forma parcimoniosa.

Em [Marins et al., 2018] os autores propõem um método iterativo para efetuar esse procedimento de seleção considerando cada classe. Com base em um conjunto de dados históricos Υ_i referente a i-ésima classe, seleciona-se como primeira amostra a média geométrica desses dados, sendo obtida resolvendo o seguinte problema de otimização:

$$\mathrm{v}_i = \underset{\mathrm{z} \in \Upsilon_i}{\operatorname{argmin}} \sum_{\mathrm{x}_i \in \Upsilon_i} \|\mathrm{x}_i - \mathrm{z}\|, \tag{4.15}$$

ou usando uma aproximação baseada no gradiente estocástico proposta por Cardot et al. [2013], ao qual possui menor custo computacional. As próximas amostras são selecionadas

seguindo a seguinte estratégia: cada nova amostra $\mathrm{x}_j \in \Upsilon_i$ é comparada com as que já estão em D_i, se a similaridade entre x_j e todas as outras em D_i for menor que um limiar $\zeta \in (0, 1]$, a amostra é adicionada em D_i, caso contrário é descartada. Formalmente, $\mathrm{x}_j \in \Upsilon_i$ será incorporada em D_i se

$$s(\mathrm{x}_j, \mathrm{x}_k) < \zeta, \quad \forall \mathrm{x}_k \in \mathrm{D}_i. \tag{4.16}$$

Naturalmente, os resultados são diretamente dependentes da função de similaridade escolhida, bem como de ζ. A metodologia mais utilizada para definir a melhor configuração consiste em uma busca em grade, fazendo-a uma tarefa ainda mais custosa.

4.2.2 Abordagem proposta baseada na M-SBM

Como observado, o estado-da-arte da SBM é focado apenas em problemas de classificação com treinamento offline. Buscando estendê-la para clusterização online, uma tarefa que atualmente é tanto crucial quanto desafiante, Almeida [2020] propuseram, baseando-se no trabalho de Mariano [2019], uma nova versão evolutiva da SBM com esquema de aprendizado não-supervisionado, denominada Modelagem Evolutiva Baseada em Similaridade de Modelos (eSBM) (do inglês, *Evolving Similarity-Based Modeling*). A versão proposta faz uso de modelos gaussianos multivariados para reduzir e simplificar o processo de definição de hiperparâmetros, bem como para agregar mais robustez a essência da técnica, uma vez que por meio desse tipo de modelo o mecanismo de aprendizado passa a seguir fundamentos estatísticos.

Resumidamente, a eSBM é composta por duas etapas: 1ª) Inferência e 2ª) Aprendizado, como ilustrado na Figura 14. Na etapa de inferência, que compreende a M-SBM, são realizadas duas ações, 1a) *Estimação*: calcula-se uma estimativa para a nova amostra considerando cada grupo existente; e 1b) *Classificação*: atribui-se a entrada ao grupo grupo que gerou a estimativa mais similar. Na etapa de aprendizado, a estrutura do modelo é atualizada, implicando na atualização dos parâmetros que caracterizam cada grupo ou até mesmo na criação de um novo.

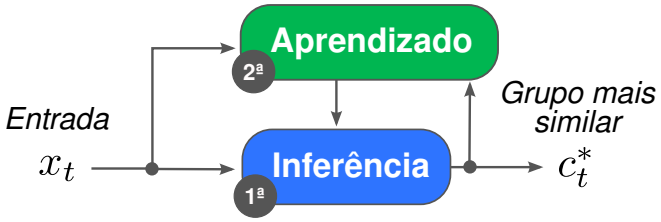


Figura 14 – Principais etapas da eSBM.

Embora a eSBM tenha agregado bastante à técnica original, a mesma ainda possui algumas limitações. A principal diz respeito ao formato da região de decisão de cada grupo.

Uma das principais qualidades da SBM está em permitir que as classes tenham uma região de decisão com formato arbitrário, sendo inteiramente baseada em como as amostras representativas estão dispersas no espaço de entrada. Contudo, devido ao uso de modelos gaussianos na eSBM, o formato dos grupos estão indiretamente limitados a hiper-elipsoides. Uma segunda limitação também é relacionada ao uso de modelos gaussianos. Dado seu uso, também se faz necessário armazenar os parâmetros da distribuição (centro e matriz de covariância) que caracteriza cada grupo, o que por consequência aumenta ainda mais os requisitos de memória.

Focando principalmente nessas duas limitações, propõe-se nessa seção algumas mudanças à eSBM. A essa nova versão com esse conjuntos de mudanças foi dado o nome de eSBM4Clus. O fluxo de execução é o mesmo da versão anterior, entretanto, foi removido o uso de modelos gaussianos, afetando assim todas as etapas. Deixando-a mais próxima da abordagem original, em que os modelos dependem unicamente da matriz memória. Uma visão geral da capacidade dessa nova versão é dada na Figura 15, onde é apresentada três bases de dados com características diferentes e o resultado obtido com o eSBM4Clus. É fácil perceber que o método foi capaz de modelar os grupos em formato e densidade próximo do original. Vale ressaltar que a métrica utilizada para definir a região de decisão no eSBM4Clus, chamada de Relação Mínima (MR) (do inglês, *Minimum Relationship*), é uma combinação de qualidade de estimativa e densidade de dados, sendo descrita ao longo desta seção.

O restante dessa seção visa detalhar o funcionamento de cada etapa no eSBM4Clus referenciando a M-SBM.

4.2.2.1 Inferência

Estimação

O procedimento de estimação é parecido àquele da M-SBM. Uma estimativa para cada classe pode ser determinada utilizando (4.11). No entanto, como função de similaridade, propõe-se uma versão modificada da definida em (4.10),conforme:

$$s(\mathrm{x}_k, \mathrm{x}_j; \sigma_i) = \frac{1}{1 + \left\| \frac{\mathrm{x}_k - \mathrm{x}_j}{\sigma_i} \right\|^2}, \quad i = 1, 2, \ldots, c_t, \tag{4.17}$$

em que c_t denota a quantidade atual de grupos e $\sigma_i \in \mathbb{R}$ é um fator de normalização calculado como

$$\tilde{\sigma}_i = \left[std(\mathrm{W}^1), std(\mathrm{W}^2), \ldots, std(\mathrm{W}^n) \right]^\top, \tag{4.18}$$

$$\sigma_i = mean\left(\tilde{\sigma}_i\right), \tag{4.19}$$

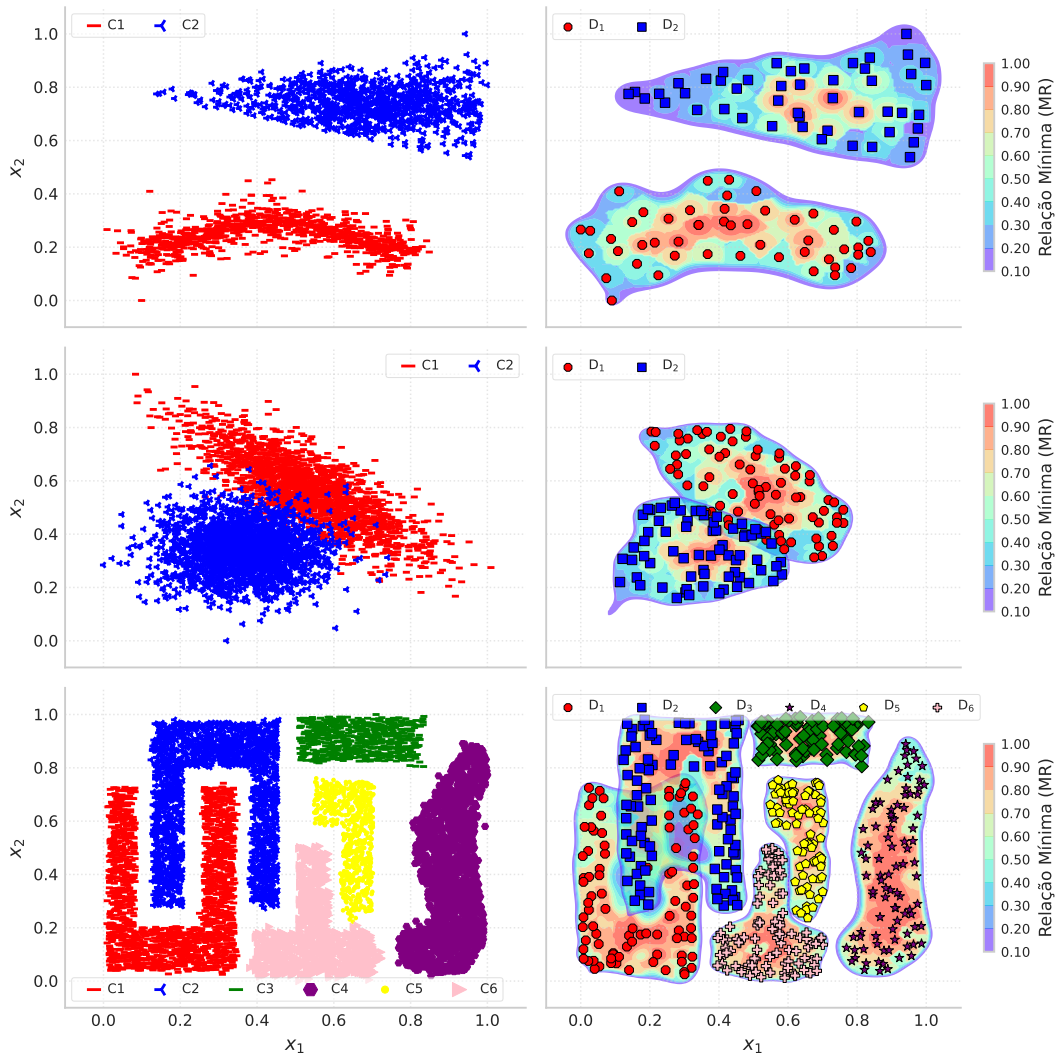


Figura 15 – Representação do formato dos clusters na eSBM4Clus usando conjunto de dados bidimensionais sintéticos. Primeiramente, ao lado esquerdo, clusters com formato incomum e com sobreposição são fornecidos. O eSBM4Clus é então aplicado, resultando nos clusters (lado direito) com formato e densidade de dados similar aos observados nos dados originais. A MR é apresentada em incrementos de 0.10 para melhor visualização. Observa-se que o formato dos clusters são diretamente influenciados pelas amostras representativas, bem como as correspondentes curvas de pertinência.

com $\tilde{\sigma}_i \in \mathbb{R}^n$ denotando um vetor que armazena o valor do desvio-padrão clássico computado para cada variável; $std(\cdot)$ denota uma função que efetivamente calcula o valor do desvio-padrão clássico, e $mean(\cdot)$ denota uma função que calcula a média aritmética dos desvios-padrão obtidos; W representa os dados iniciais do grupo (conforme descrito em mais detalhes na Seção 4.2.2.2). O objetivo com a adição de σ_i é padronizar a medida de similaridade, isto é, a similaridade terá a mesma interpretação/referência em todos os grupos.

Classificação

Dada a entrada x_t e as respectivas estimativas $\hat{\mathrm{x}}_{ti}$ da fase anterior. Nesta fase, é realizada um associação suave usando a chamada Relação Mínima (MR). A MR avalia quão bem os grupos atuais representam x_t em termos de:

- Erro de Aproximação: Fornecendo x_t e $\hat{\mathrm{x}}_{ti}$ como argumento para uma versão modificada da função de similaridade definida em (4.10), avalia-se quão bem cada grupo aproxima x_t. Tal versão modificada é denotada como Qualidade da Estimativa (EQ) (do inglês, *Estimate Quality*), sendo formalmente definida como:

$$\mathrm{EQ}_{ti} = \frac{1}{1 + \frac{\mathcal{E}_{ti}}{\tau_i}}, \quad i = 1, 2, \ldots, c_t, \tag{4.20}$$

$$\mathcal{E}_{ti} = \frac{\left\|\mathrm{x}_t - \hat{\mathrm{x}}_{ti}\right\|^2}{\left\|\mathrm{x}_t\right\|^2 + \epsilon}, \tag{4.21}$$

  em que $\tau_i \in \mathbb{R}$ é um limiar de erro de aproximação, sendo incrementalmente calculado conforme será oportunamente apresentado em (4.32) utilizando os erros das amostras associadas ao respectivo grupo; $\mathcal{E}_{ti} \in \mathbb{R}$ é o Erro Quadrático Relativo (RSE) (do inglês, *Relative Squared Error*) entre x_t and $\hat{\mathrm{x}}_{\mathrm{ti}}$; $\epsilon \in \mathbb{R}$ denota uma pequena constante utilizada para evitar indeterminação. Utilizou-se $\epsilon = 1 \times 10^{-10}$ nessa tese. É importante ressaltar que τ_i é apenas um fator de normalização para o RSE, tendo assim o mesmo propósito/interpretação do termo σ_i.

- Potencial de Densidade de Dados: A densidade dos dados é resumida pela associação de uma média de similaridade a cada amostra representativa de cada grupo. Tal média é incrementalmente computada com o vetor a_{ti} (advindo de (4.11)) de todas as amostras associadas ao respectivo grupo. Em outras palavras, considerando que o grupo i possui l_i amostras representativas no instante t, a similaridade média de cada amostra representativa é arranjada em um vetor $\mathrm{S}_i \in \mathcal{U}^{l_i}$, tal qual

$$\mathrm{S}_i = \left[\bar{\mathrm{s}}_1, \bar{\mathrm{s}}_2, \ldots, \bar{\mathrm{s}}_{l_i}\right]^\top, \tag{4.22}$$

  com $\bar{\mathrm{s}}_{\mathrm{l_i}} \in \mathcal{U}$ denotando a similaridade média da l_i-ésima amostra representativa, sendo computada incrementalmente utilizando (4.28) com cada amostra atribuída até o instante atual. Assim, baseando-se no vetor S_i, define-se uma métrica chamada Potencial de Densidade (PD) (do inglês, *Potential Density*), ao qual avalia quão denso é a área entorno de x_t, tendo como referencial os dados observados até o momento. A PD é formalmente definida como:

$$\mathrm{PD}_{ti} = \left[\mathrm{a}_{ti}\right]_{j_i^*} \frac{\left[\mathrm{S}_i\right]_{j_i^*}}{max\left(\mathrm{S}_i\right)}, \quad i = 1, 2, \ldots, c_t, \tag{4.23}$$

em que j_i^* é um índice que aponta para a amostra representativa mais similar a x_t de acordo com o vetor a_{ti}, isto é,

$$j_i^* = \underset{j \in \{1,2,\ldots,l_i\}}{\operatorname{argmax}} \left[\mathrm{a}_{ti}\right]_j. \tag{4.24}$$

Percebe-se que a PD é basicamente a similaridade média normalizada considerando a amostra representativa mais similar a x_t e a mais similar a todas as amostras observadas e associadas ao grupo i até o momento.

Por fim, a MR é formalmente definida como:

$$\mathrm{MR}_{ti} = min\left(\mathrm{EQ}_{ti}, \mathrm{PD}_{ti}\right), \quad i = 1, 2, \ldots, c_t, \tag{4.25}$$

em que o grau de pertencimento é definido como o valor mínimo (relação mínima) de x_t para o grupo i: EQ ou PD. Propõe-se a MR buscando lidar principalmente com as seguintes questões:

- *Classificação baseada no comportamento usual dos dados*: A classificação baseada somente na EQ não considera a região usual em que os dados "chegam" (área com maior densidade de dados), o que pode acarretar em altas taxas de erros de classificação, especialmente quando os grupos estão muitos próximos ou sobrepostos.
- *Expansão de grupo*: Usar apenas a posição espacial das amostras representativas para decidir quando uma nova entrada pode ser adicionada à matriz memória é uma tarefa inerentemente complexa, especialmente quando os grupos estão próximos. Logo, a PD facilita o processo de tomada de decisão fornecendo uma medida de densidade de dados. Consequentemente, por meio dessa métrica é possível saber qual é a sub-região usual de dados do respectivo grupo.

Por fim, x_t é atribuído ao grupo com a maior MR utilizando o principio *winner-takes-all*:

$$c_t^* = \underset{i \in \{1,2,\ldots,c_t\}}{\operatorname{argmax}} \mathrm{MR}_{ti}, \tag{4.26}$$

em que c_t^* denota o grupo vencedor.

4.2.2.2 Aprendizado

O aprendizado do eSBM4Clus é caracterizado por duas principais ações: i) Criar grupos; e ii) Atualizar os grupos e demais parâmetros existentes. Ambas as ações são

realizadas considerando a seguinte condição que utiliza o grupo mais similar a x_t, c_t^*, para verificar se a mesma é referente a uma situação `normal` ou `novidade`:

$$norm\_ou\_novi\left(\mathcal{E}_{tc_t^*}, \tau_{c_t^*}, \mathrm{PD}_{tc_t^*}\right) = \begin{cases} \texttt{normal}, & \mathcal{E}_{tc_t^*} \leq \tau_{c_t^*} \textbf{ e } \mathrm{PD}_{tc_t^*} \geq \beta, \\ \texttt{novidade}, & c.c., \end{cases} \tag{4.27}$$

em que $\beta \in (0, 1]$ é um hiperparâmetro definido pelo usuário que corresponde a um limiar de densidade. Na Subseção 4.2.2.3 será dada uma descrição detalhada. O restante dessa subseção descreve as duas ações que compõem o aprendizado do método.

i) Criação

Quando x_t é rotulado como `novidade` (de acordo com (4.27)), o mesmo é adicionado a um conjunto de novidades denotado por W. Esse conjunto irá armazenar sequencialmente "novidades" únicas que posteriormente são utilizadas como a matriz memória inicial do novo grupo. Contudo, um grupo é criado somente quando W contiver $w \in \mathbb{N}^*$ novidades, em que w é um hiperparâmetro do método. Vale ressaltar ainda que o primeiro grupo é criado com as w amostras iniciais do fluxo de dados.

O novo grupo é composto pelos seguintes atributos:

- Matriz Memória: D_{c_t+1}
  Valor Inicial: conjunto de novidades W.
- Fator de Normalização: σ_{c_t+1}
  Valor Inicial: Utiliza-se (4.19) e W como dados iniciais.
- Matriz de Intra-Similaridade: G_{c_t+1}
  Valor Inicial: $[\mathrm{G}^{kj}]_{c_t+1} = s(\mathrm{x}_k, \mathrm{x}_j), \quad \forall \mathrm{x}_k, \mathrm{x}_j \in \mathrm{W},$
  em que G^{kj} denota a célula que armazena a similaridade entre x_k e x_j.
- Inversa da Matriz de Intra-Similaridade: $\mathrm{G}^{-1}_{c_t+1}$
  Valor Inicial: $\mathrm{G}^{-1}_{c_t+1}$.
- Vetor de Similaridade Média: S_{c_t+1}
  Valor Inicial:: $\mathrm{S}_{c_t+1} = [1_1, 1_2, \ldots, 1_w]^\top$.
- Quantidade de Amostras Atribuídas: n_{c_t+1}
  Valor Inicial: w.
- Limiar do Erro de Aproximação: τ_{c_t+1}
  Valor Inicial: Utiliza-se o Algoritmo 2 passando D_{c_t+1} e G_{c_t+1} como argumentos. O algoritmo fornecerá o maior erro de aproximação do conjunto inicial de dados.

Algoritmo 2: *Procedimento para calcular o valor inicial do Limiar do Erro de Aproximação..*

Entrada : D, G
Saída : τ.

$\mathcal{E}_{all} \leftarrow \emptyset$.
foreach x ***in*** D **do**
  $\mathrm{D}_{temp} \leftarrow$ D sem x.
  $\mathrm{G}_{temp} \leftarrow$ G sem a linha e coluna associada a amostra x.
  $\hat{\mathrm{x}} \leftarrow$ calcula-se uma estimativa para x com (4.11) utilizando D_{temp} e G_{temp}.
  $\mathcal{E} \leftarrow$ calcula-se o erro de aproximação com (4.21).
  $\mathcal{E}_{all} \leftarrow \mathcal{E}_{all} \cup \mathcal{E}$.
end
$\tau \leftarrow mean(\mathcal{E}_{all})$.
return τ

ii) Atualização

Quando x_t é rotulado como "novidade", primeiramente o conjunto W é limpo, e em seguida a respectiva amostra é usada para atualizar o grupo c_t^*. A atualização envolve os seguintes passos:

1. Atualiza-se o Vetor de Similaridade Média

$$[\mathrm{S}_{c_t^*}]^{t+1} \leftarrow \frac{[n_{c_t^*}]^t}{[n_{c_t^*}]^t + 1}[\mathrm{S}_{c_t^*}]^t + \frac{1}{[n_{c_t^*}]^t + 1}\mathrm{a}_{tc_t^*}. \tag{4.28}$$

2. Adiciona-se x_t à matriz memória se o mesmo for uma amostra representativa: x_t é considerado representativo se

$$\mathcal{E}_{tc_t^*} \geq \gamma \cdot \tau_{c_t^*}, \tag{4.29}$$

   em que $\gamma \in (0, 1]$ é um hiperparâmetro do método que define uma distância mínima entre as atuais amostras representativas baseando-se no erro médio de aproximação do grupo (na Subseção 4.2.2.3 é fornecida uma descrição mais detalhada). No entanto, na tentativa de estabelecer um limite superior para o custo computacional, define-se um limite para o crescimento da matriz memória; tal limite é formalmente estabelecido pela introdução do hiperparâmetro $l_{max} \in \mathbb{N}^*$. Logo, se a matriz memória já possuir l_{max} amostras representativas, x_t é utilizado para substituir aquela a qual é mais similar. Desse modo, se (4.29) for satisfeita e,

   - $l_{c_t^*} < l_{max}$, x_t é adicionado à matriz memória e os atributos relacionados atualizados:

$$\begin{aligned}
[\mathrm{D}_{c_t^*}]^{t+1} &\leftarrow [\mathrm{D}_{c_t^*}]^t \cup \mathrm{x}_t, \\
[\mathrm{G}_{c_t^*}]^{t+1} &\leftarrow [\mathrm{G}_{c_t^*}]^t \cup \mathrm{a}_{tc_t^*}, \\
[\mathrm{G}^{-1}_{c_t^*}]^{t+1} &\leftarrow ([\mathrm{G}_{c_t^*}]^{t+1})^{-1} \\
[\mathrm{S}_{c_t^*}]^{t+1} &\leftarrow [\mathrm{S}_{c_t^*}]^{t+1} \cup \ min\left([\mathrm{S}_{c_t^*}]^{t+1}\right),
\end{aligned} \tag{4.30}$$

em que $[\mathrm{S}_{c_t^*}]^{t+1}$ no lado direito faz referência ao vetor de similaridade média atualizado no passo (1).

- $l_{c_t^*} = l_{max}$, x_t é utilizado para substituir a amostra representativa a qual é mais similar, $j^*_{c_t^*}$ (de acordo com (4.24))

$$\begin{aligned}
[\mathrm{D}_{c_t^*}]^{t+1} &\leftarrow replace^{\mathrm{D}}\left([\mathrm{D}_{c_t^*}]^t, \mathrm{x}_t, j^*_{c_t^*}\right) \\
[\mathrm{G}_{c_t^*}]^{t+1} &\leftarrow replace^{\mathrm{G}}\left([\mathrm{G}_{c_t^*}]^t, \mathrm{a}_{tc_t^*}, j^*_{c_t^*}\right) \\
[\mathrm{S}_{c_t^*}]^{t+1} &\leftarrow replace^{\mathrm{S}}\left([\mathrm{S}_{c_t^*}]^{t+1}, min\left([\mathrm{S}_{c_t^*}]^{t+1}\right), j^*_{c_t^*}\right) \\
[\mathrm{G}^{-1}_{c_t^*}]^{t+1} &\leftarrow ([\mathrm{G}_{c_t^*}]^{t+1})^{-1}
\end{aligned} \tag{4.31}$$

em que $replace^{\mathrm{D}}(\cdot,\cdot,\cdot)$ denota uma função que realiza a substituição da linha $j^*_{c_t^*}$ de $\mathrm{D}_{c_t^*}$ por x_t; $replace^{\mathrm{G}}(\cdot,\cdot,\cdot)$ denota uma outra função que realiza a substituição da linha e coluna $j^*_{c_t^*}$ de $\mathrm{G}_{c_t^*}$ por $\mathrm{a}_{tc_t^*}$; e, $replace^{\mathrm{S}}(\cdot,\cdot,\cdot)$ realiza a substituição do item $j^*_{c_t^*}$ pelo menor valor de similaridade média em $[\mathrm{S}_{c_t^*}]^{t+1}$, $min\left([\mathrm{S}_{c_t^*}]^{t+1}\right)$, em que $[\mathrm{S}_{c_t^*}]^{t+1}$ no lado direito é o vetor de similaridade média atualizado no passo (1).

3. Atualiza-se o Limiar do Erro de Aproximação

$$[\tau_{c_t^*}]^{t+1} \leftarrow \frac{[n_{c_t^*}]^t}{[n_{c_t^*}]^t + 1}[\tau_{c_t^*}]^t + \frac{1}{[n_{c_t^*}]^t + 1}\mathcal{E}_{tc_t^*}. \tag{4.32}$$

4. Atualiza-se a Quantidade de Amostras Atribuídas

$$[n_{c_t^*}]^{t+1} \leftarrow [n_{c_t^*}]^t + 1. \tag{4.33}$$

5. O Fator de Normalização ($\sigma_{c_t^*}$) é mantido fixo: recomenda-se mantê-lo fixo porque, para fins práticos (normalização), somente o valor inicial é suficiente. Atualizá-lo implicaria em recalcular toda a matriz de intra-similaridade sempre que um nova amostra representativa for adicionada. Mantendo-o fixo, se faz necessário apenas um expansão de linha e coluna.

Um resumo do eSBM4Clus dado no Algoritmo 3.

Algoritmo 3: *Evolving Similarity-Based Modeling for Online Clustering (eSBM4Clus).*

Entrada : $\mathrm{x}_t \,|\, t = 1, 2, \ldots, +\infty$.
Saída : c_t^*.

Inicialização
  Define-se w, β, γ e l_{max}.
  $\mathrm{W} \leftarrow \emptyset$.
  $c_t \leftarrow 0$.
begin
  $c_t^* \leftarrow 1$.
  if *não existir grupo* **then**
    Aguarda-se w amostras e cria-se o primeiro grupo com *create*($\mathrm{x}_{1:w}$).
    $c_t \leftarrow c_t + 1$.
  else
    foreach i ***in*** $\{1, 2, \ldots, c_t\}$ **do**
      $\hat{\mathrm{x}}_{ti} \leftarrow$ Calcula-se uma estimativa com (4.11) considerando o grupo i.
    end
    $c_t^* \leftarrow$ realiza-se a atribuição rígida usando (4.26).
    if (4.27) == ***novidade*** **then**
      $\mathrm{W} \leftarrow \mathrm{W} \cup \mathrm{x}_t$
      if W *possuir* w *amostras* **then**
        Cria-se um grupo com *create*(W).
        $c_t \leftarrow c_t + 1$.
        $c_t^* \leftarrow c_t$.
        $\mathrm{W} \leftarrow \emptyset$.
      end
    else
      $\mathrm{W} \leftarrow \emptyset$.
      Atualiza-se o grupo mais similar com *update*(x_t, c_t^*).
    end
  end
  return c_t^*.
end
function *create(*W*)*
  $\mathrm{D}_{c_t+1} \leftarrow \mathrm{W}$.
  $\sigma_{c_t+1} \leftarrow$ Calcula-se o fator de normalização usando (4.19).
  $\mathrm{S}_{c_t+1} \leftarrow [1_1, 1_2, \ldots, 1_w]^\top$.
  $\mathrm{G}_{c_t+1} \leftarrow$ Calcula-se a similaridade para cada par de amostras em W usando (4.17).
  $\mathrm{G}^{-1}_{c_t+1} \leftarrow$ Calcula-se a inversa de G_{c_t+1}.
  $\tau_{c_t+1} \leftarrow$ Calcula-se o valor inicial do limiar do erro de aproximação usando o Algoritmo 2.
  $\mathrm{n}_{c_t+1} \leftarrow w$.
end
function *update(*x_t, c_t^**)*
  $\mathrm{S}_{c_t^*} \leftarrow$ Atualiza-se usando (4.28).
  if (4.29) *for satisfeita* **then**
    if $l_{c_t^*} < l_{max}$ **then**
      Atualiza-se $\mathrm{D}_{c_t^*}, \mathrm{G}_{c_t^*}, \mathrm{G}^{-1}_{c_t^*}$ e $\mathrm{S}_{c_t^*}$ de acordo com (4.30).
    else
      Atualiza-se $\mathrm{D}_{c_t^*}, \mathrm{G}_{c_t^*}$ e $\mathrm{G}^{-1}_{c_t^*}$ de acordo com (4.31).
    end
  end
  $\tau_{c_t^*} \leftarrow$ Atualiza-se usando (4.32).
  $\mathrm{n}_{c_t^*} \leftarrow$ Atualiza-se usando (4.33).
end

4.2.2.3 Intuição na definição dos valores dos hiperparâmetros

Como pôde ser visto, a abordagem proposta possui um total de quatro hiperparâmetros : w, β, γ e l_{max}. Esta subseção dedica-se a explicação da intuição que envolve a definição de cada um. Para tanto, utiliza-se a mesma base de dados apresentada na Figura 15 para apresentar o comportamento da clusterização ao variá-los dentro do limites sugeridos. No contexto em que o hiperparâmetro não está sendo variado, utiliza-se por padrão os seguintes valores $w = 10, \beta = 0.99, \gamma = 0.1$ e $l_{max} = 200$.

Antes de tudo, vale comentar sobre um aspecto importante do eSBM4Clus: a métrica de distância. Sabe-se que a expressão obtida para o calculo da estimativa foi derivada a partir da distância Euclideana, logo, se faz importante mencionar o comportamento do método quando outras métricas de distância são utilizadas. Para isso, utiliza-se a norma LP, definida como [Rolewicz, 1987]:

$$d(x_k, x_l) = \left(\sum_{j=1}^{n} |x_k^j - x_l^j|^p \right)^{\frac{1}{p}}, \tag{4.34}$$

em que $p \in \mathbb{R}^+$ controla a métrica de distância alvo. Note que quando $p = 2$, a expressão fornece a distância Euclideana. A Figura 16 apresenta o comportamento da clusterização ao variar p. Por meio da figura observa-se que o formato dos grupos deteriora quando p se aproxima dos limites utilizados ($p \in [0.1, 4]$), esse comportamento ocorre porque, conforme dito anteriormente, a função objetivo utilizada para derivar a expressão de estimação (4.11) utiliza a distância Euclideana ($p = 2$) como medida de erro. Logo, outras métricas de distância podem não fornecer a medida de erro da forma apropriada. Com isso é possível concluir que, outras métricas distâncias podem ser utilizadas, o que em geral fornece novos formatos para os grupos, mas ao custo de impactar na qualidade da estimativa.

Tamanho do conjunto de novidades (w)

O parâmetro w controla a quantidade consecutiva necessária de amostras rotuladas como "novidade" para a criação de um grupo. Naturalmente, o mesmo influencia diretamente na clusterização, uma vez que tal conjunto de amostras é utilizado como a matriz memória inicial do novo grupo. Assim sendo, valores pequenos podem levar a criação de muitos grupos porque a matriz memória inicial pode não aproximar suficientemente bem a região real do grupo. Por outro lado, valores grandes podem tornar o método altamente conservador, ocasionado em sua incapacidade de detectar mudanças recorrentes rápidas [Gama et al., 2014] no fluxo de dados.

Além disso, vale ressaltar que devido ao critério de adição de amostras ao conjunto W não considerar a localização espacial da amostra, quando w é muito grande, novos

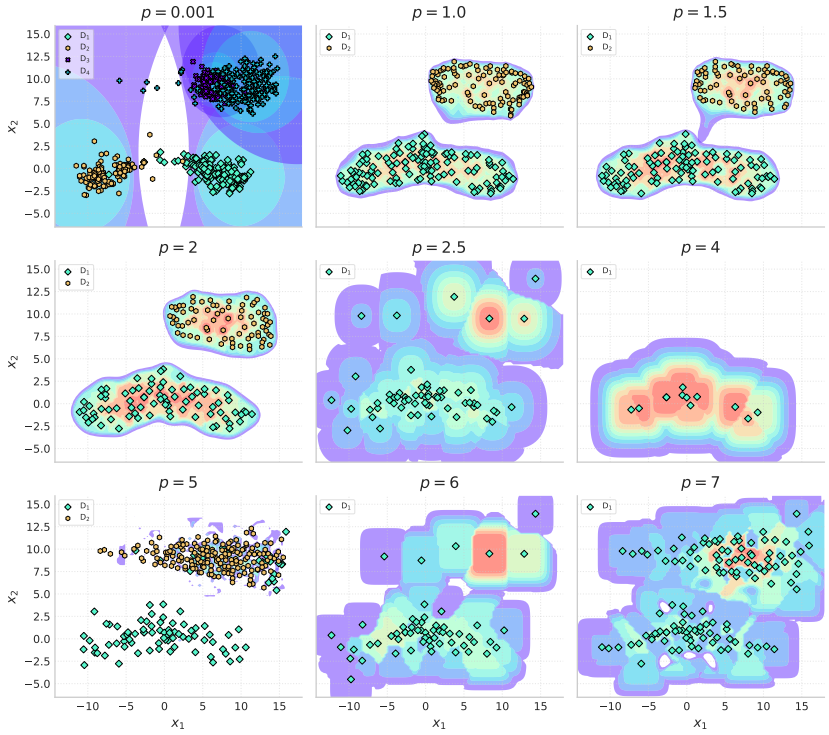


Figura 16 – Comportamento da clusterização ao variar p enquanto os demais hiperparâmetros são mantidos fixos.

grupos podem possuir alto grau de sobreposição com os existentes. Procedimentos para tratar essa deficiência têm sido estudados.

Uma ilustração do comportamento da clusterização ao variar w é dado na Figura 17. Percebe-se que valores pequenos levam o eSBM4Clus a criar mais grupos, pois o conjunto inicial de amostras representativas não é suficiente para representar a real região do grupo. Percebe-se também que a partir de um tamanho mínimo necessário ($w = 5$ nessa base de dados), o método fica estável.

Limiar do potencial de densidade(β)

A partir do critério de novidade ((4.27)), percebe-se que β é basicamente um limiar rígido. O mesmo define o valor mínimo de PD para que uma nova entrada seja considerada "usual" em termos de potencial de densidade. Em termos práticos, $\beta \approx 1$ torna a abordagem menos conservadora, permitindo apenas um/uma leve crescimento/expansão. Logo, amostras representativas um pouco mais distantes das iniciais não serão incorporadas à matriz memória, e assim o método tenderá a criar mais grupos, promovendo assim o/a crescimento/expansão dos grupos. Esse comportamento é ilustrado na Figura 18.

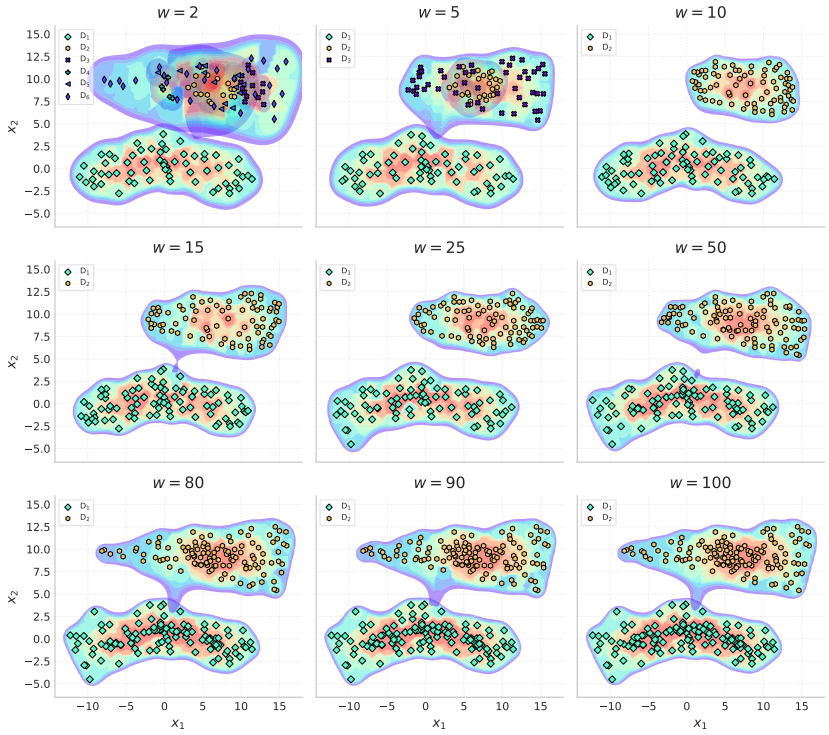


Figura 17 – Comportamento da clusterização ao variar w enquanto os demais hiperparâmetros são mantidos fixos.

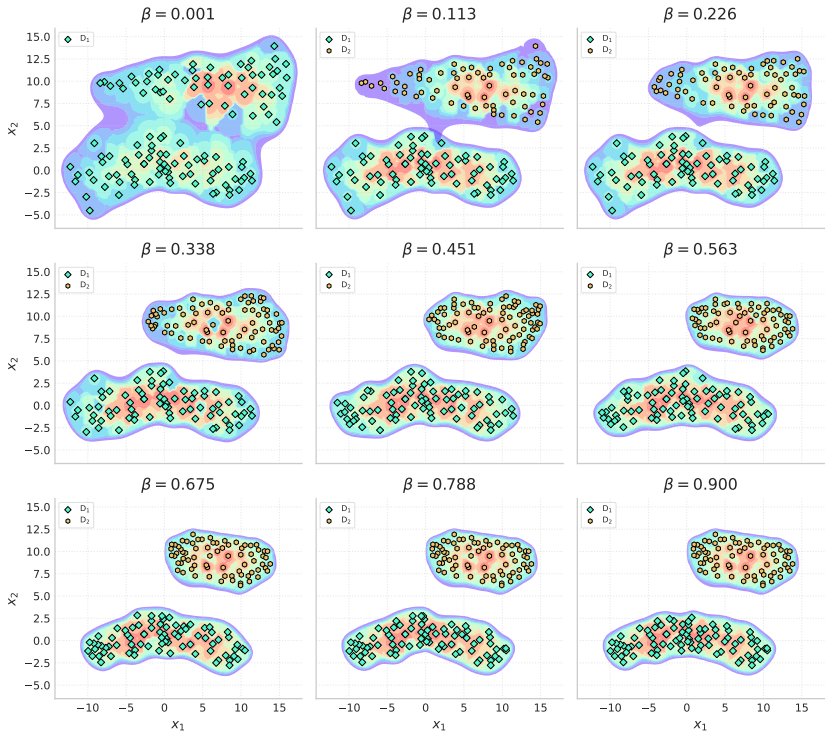


Figura 18 – Comportamento da clusterização ao variar β enquanto os demais hiperparâmetros são mantidos fixos.

Limiar de distância mínima entre amostras representativas (γ)

Imagine que, para uma amostra ser adiconada à matriz memória se faz necessário que a mesma passe por dois portões. O primeiro é controlado por (4.27), ao qual permite a

passagem na situação em que a amostra foi rotulada como "usual". Após cruzar o primeiro portão, para cruzar o segundo a amostra agora precisa ser rotulada como "não-redundante" por (4.29), que por sua vez é controlado diretamente por γ. Desse modo, ao verificar (4.29), leva-se em consideração que τ_i armazena a média do erro de aproximação e indiretamente mede o quanto em média a estimativa do grupo ficou distante da entrada. É possível concluir que γ determina a distância mínima que a nova entrada dever ter das amostras já contidas na matriz memória.

Consequentemente, $\gamma \approx 0$ promoverá o crescimento da matriz memória e quando atingir o tamanho máximo, a quantidade de substituições, pois o erro de aproximação necessário é reduzido. Por outro lado, $\gamma \approx 1$ tenderá a manter a matriz memória próxima de seu tamanho inicial e também tenderá a manter as amostras representativas mais espacialmente distantes. Vale ressaltar ainda que esse hiperparâmetro também influencia diretamente no comportamento da clusterização, uma vez que controla o conjunto de amostras representativas. Tal comportamento pode ser observado por meio do exemplo dado na Figura 19.

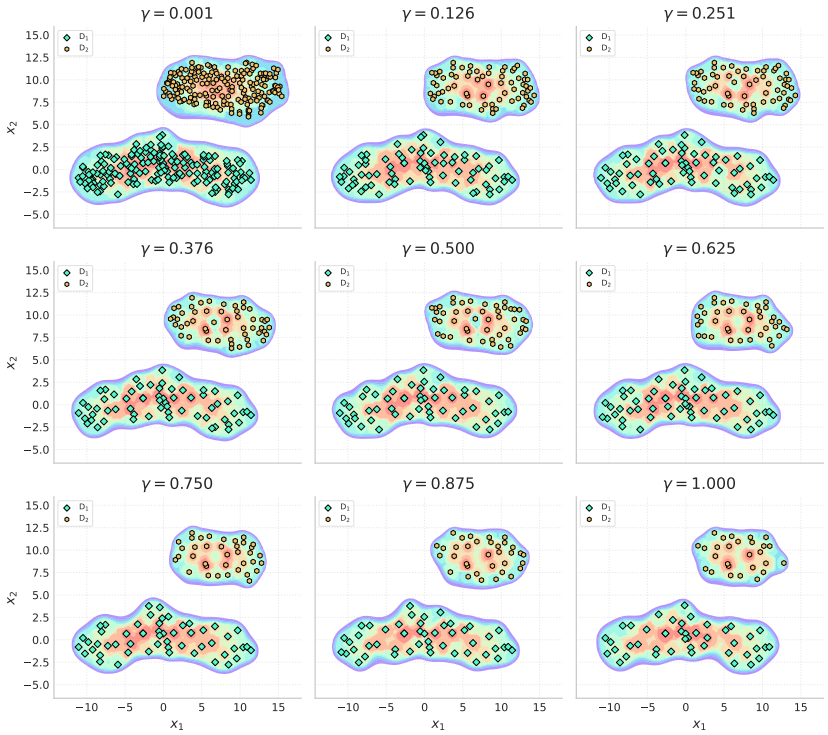


Figura 19 – Comportamento da clusterização ao variar γ enquanto os demais hiperparâmetros são mantidos fixos.

Tamanho máximo da matriz memória (l_{max})

O l_{max} define o tamanho máximo da matriz memória, logo, o mesmo define o limite superior (pior caso em termos complexidade algorítmica) do custo computacional. Naturalmente, o mesmo também influencia diretamente o comportamento da clusterização

porque determina, em uma perspectiva geral, a capacidade de representação dos grupos pelo método, visto que os grupos são definidos inteiramente pelas amostras representativas. Determina também o momento que amostras representativas antigas serão substituídas por novas. O comportamento da clusterização com diferentes valores de l_{max} é ilustrado na Figura 20.

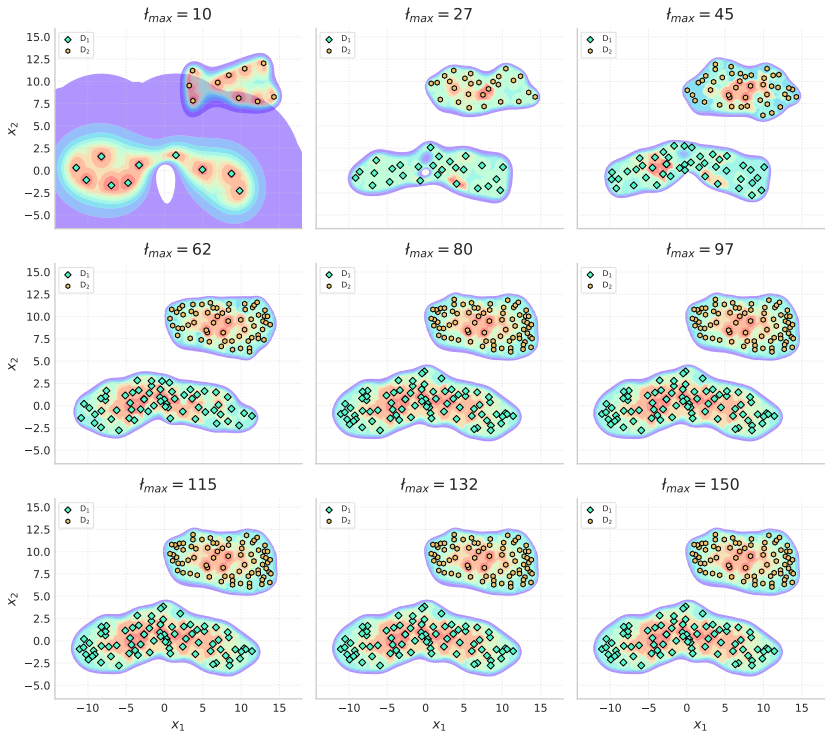


Figura 20 – Comportamento da clusterização ao variar l_{max} enquanto os demais hiperparâmetros são mantidos fixos.

4.3 Resumo do Capítulo

Neste capítulo foi apresentado uma revisão literária das principais e mais recentes abordagens voltadas a clusterização evolutiva. Também foi apresentada a proposta de um novo método evolutivo baseado na técnica SBM, o eSBM4Clus. E de forma geral, emprega uma esquema de aprendizado baseado inteiramente no uso de conjuntos mínimos de amostras representativas para representar os clusters. O funcionamento e forma de parametrização também foram detalhadamente apresentados através de figuras ilustrativas.

Capítulo 5

Experimentos

Neste capítulo são realizados alguns experimentos com o objetivo de avaliar a efetividade das abordagens propostas na tarefa de detecção e classificação de eventos. Os experimentos conduzidos buscam avaliar se os métodos são capazes de aprender a dinâmica do sistema observando-o uma única vez, e, em seguida, discriminá-la de forma eficaz. No entanto, como todas as abordagens são baseadas em um paradigma de aprendizado não supervisionado, um determinado comportamento do sistema pode ser representado por um conjunto de modelos locais. Assim, para utilizar métricas discriminativas, é necessário realizar um pré-processamento que rotule esses modelos locais, indicando a qual classe real cada um está associado. Em uma situação real, tal rotulagem é de responsabilidade do especialista, entretanto, aqui esse papel será automatizado utilizando os rótulos reais disponíveis nas bases de dados reais e públicas. Todos os arquivos-fonte estão disponíveis em `https://github.com/nayronmorais/phd-thesis-ppgee-ufmg`.

Para tanto, o restante do capítulo está organizado da seguinte forma: inicialmente são apresentadas as propriedades estatísticas e detalhes de três bases de dados utilizadas para simular fluxos. Em seguida, é descrito o procedimento de rotulação automática; por seguinte é apresentada a metodologia experimental utilizada. Por fim, os resultados obtidos são apresentados, comparados e discutidos.

5.1 *Benchmarks*

5.1.1 3W *dataset*

O 3W dataset foi criado por Vargas et al. [2019]. A base contém dados reais históricos de poços surgentes de petróleo. Os dados reais são oriundos de 21 poços produtores da empresa Petrobras, mensurados a uma frequência fixa de 1Hz.

Na Figura 21 é apresentado um esquema geral dos poços considerados no 3W dataset. Resumidamente, nesse tipo de poço o petróleo flui a partir do reservatório

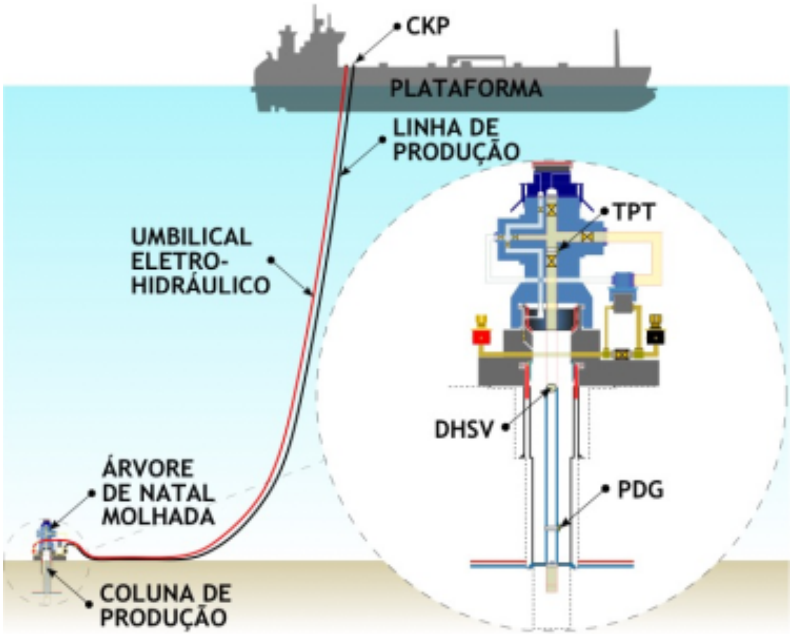


Figura 21 – Esquema geral de um poço marítimo surgente de petróleo do 3W dataset. Fonte: Vargas et al. [2019].

pela Coluna de Produção, Árvore de Natal Molhada (ANM), Linha de Produção e pela válvula *Choke* de Produção (CKP) até a plataforma. Vale destacar que a ANM é um equipamento composto basicamente por válvulas operadas remotamente e sensores, como *Temperature and Pressure Transducer* (TPT). CKP é uma válvula de controle instalada no início da unidade de produção. Ela é responsável pelo controle do poço na superfície. *Permanent Downhole Gauge* (PDG) e TPT são equipamentos que contém sensores de pressão e temperatura. *Down Hole Safety Valve* (DHSV) é um tipo de válvula hidráulica de segurança do tipo falha-fecha; e, o Umbilical Eletro-Hidráulico interliga as válvulas e sensores da ANM, assim como o PDG e a DSHV se conectam aos sistemas na superfície [Vargas et al., 2019].

O *benchmark* permite o monitoramento dos poços por meio de um total de cinco variáveis, listadas na Tabela 1. Também é fornecido, além da operação normal, dados de um total de oito anomalias. A Tabela 2 as relaciona. Nesta tese, as anomalias 3 e 4 foram desconsideradas, devido suas características discriminativas.

Tabela 1 – Variáveis do 3W dataset.

Identificador	**Descrição**
P-PDG	Pressão do fluido no PDG
P-TPT	Pressão do fluido no TPT
T-TPT	Temperatura do fluido no TPT
P-MON-CKP	Pressão do fluido montante à válvula *Choke* de Produção
T-JUS-CKP	Temperatura do fluido jusante à válvula *Choke* de Produção

Tabela 2 – Anomalias existentes no 3W dataset.

ID	Nome	Descrição
1	Aumento Abrupto de BSW	Aumento abrupto do *Basic Sediment and Water* (BSW). Definido como a razão entre a vazão de água mais sedimentos produzidos e a vazão de líquido produzido.
2	Fechamento Espúrio do DHSV	Fechamento espúrio da válvula DHSV.
3	Intermitência Severa	Instabilidade que pode resultar em estresse e danos aos equipamentos.
4	Instabilidade de Fluxo	Instabilidade com potencial para evoluir para uma Intermitência Severa.
5	Perda Rápida de Produtividade	Poço perde características favoráveis ao escoamento a níveis satisfatórios do petróleo à superfície.
6	Restrição Rápida em CKP	Ocorrência de restrição na válvula CKP por um curto período de tempo.
7	Incrustação em CKP	Ocorrência de depósitos inorgânicos na válvula CKP.
8	Hidrato em Linha de Produção	Formação de hidratos na Linha de Produção.

5.1.2 *Tennessee eastman process*

O *Tennessee Eastman Procces* (TEP) foi criado pela *Eastman Chemical Company* com o intuito de proporcionar um problema baseado em um processo real para avaliar o controle de processos e métodos de monitoramento [Downs and Vogel, 1993]. É um *benchmark* muito conhecido e aplicado na comparação de diferentes métodos de detecção e diagnóstico de faltas. O diagrama P&I (do inglês, *Process and Instrumentation*) é apresentado na Figura 22. São observadas 5 unidades principais: reator, condensador, compressor, separador e *stripper*. Cinco entradas gasosas (A, B, C, D e E) alimentam o reator, que forma dois produtos líquidos (G e H) e um subproduto (F). Os produtos inicialmente alimentam o condensador para resfriar o vapor, e, depois, vão para o separador. A parte condensada vai para o *stripper* onde os resíduos são removidos, e a parte não condensada é compressada e reenviada ao reator. Finalmente, o subproduto F é expurgado do sistema depois de passar pelo separador [Downs and Vogel, 1993, Liu, 2012].

O processo tem um total de 52 variáveis, conforme apresentado na Tabela 3; 41 são variáveis de medição (x_1 a x_{41}) e 11 são variáveis manipuladas (x_{42} a x_{52}). Na descrição original do processo são programados até 21 modos de operação, sendo um deles normal e 20 em falha. De acordo com a descrição original, para 5 dessas 20 falhas não se sabe a causa ou as variáveis responsáveis. Nesta tese, será utilizado apenas as falhas 6 e 18, devido suas propriedades discriminativas em relação as demais. A Falha 6 é caracterizada por uma perda na alimentação A (mudança em degrau no fluxo 1), enquanto que a Falha 18 tem sua origem desconhecida.

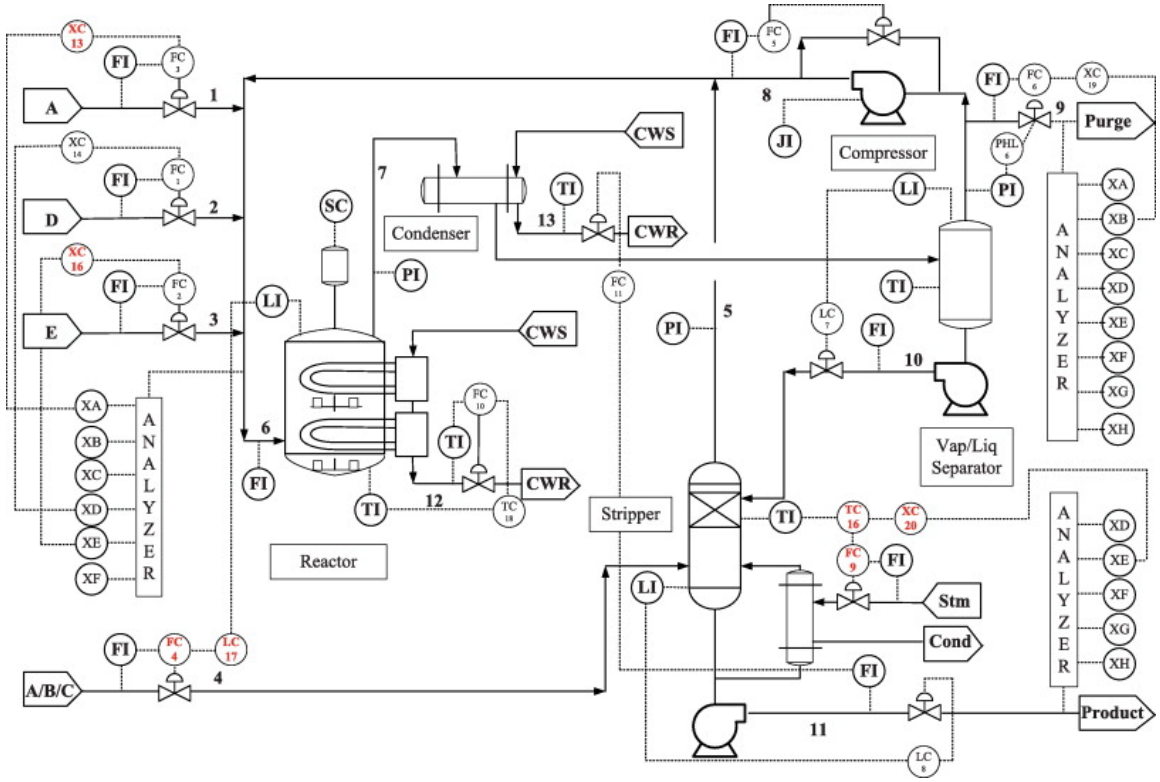


Figura 22 – Diagrama P&I do *Tennessee Eastman Process*.

Fonte: Liu [2012].

Tabela 3 – Variáveis do TEP.

ID	Descrição	ID	Descrição
x_1	Alimentação A (Fluxo 1)	x_{27}	Componente E (Fluxo 6)
x_2	Alimentação D (Fluxo 2)	x_{28}	Componente F (Fluxo 6)
x_3	Alimentação E (Fluxo 3)	x_{29}	Componente A (Fluxo 9)
x_4	Alimentação A e C (Fluxo 4)	x_{30}	Componente B (Fluxo 9)
x_5	Vazão de reciclagem (Fluxo 8)	x_{31}	Componente C (Fluxo 9)
x_6	Taxa de alimentação do reator (Fluxo 6)	x_{32}	Componente D (Fluxo 9)
x_7	Pressão reator	x_{33}	Componente E (Fluxo 9)
x_8	Nível reator	x_{34}	Componente F (Fluxo 9)
x_9	Temperatura reator	x_{35}	Componente G (Fluxo 9)
x_{10}	Taxa de expurgo (Fluxo 9)	x_{36}	Componente H (Fluxo 9)
x_{11}	Temperatura do produto no separador	x_{37}	Componente D (Fluxo 11)
x_{12}	Nível do produto no separador	x_{38}	Componente E (Fluxo 11)
x_{13}	Pressão do produto no separador	x_{39}	Componente F (Fluxo 11)
x_{14}	Sub-fluxo do produto no separador	x_{40}	Componente G (Fluxo 11)
x_{15}	Nível do *stripper*	x_{41}	Componente H (Fluxo 11)
x_{16}	Pressão do *stripper*	x_{42}	MV para vazão de alimentação D (Fluxo 2)
x_{17}	Sub-fluxo do *stripper* (Fluxo 11)	x_{43}	MV para vazão de alimentação E (Fluxo 3)
x_{18}	Temperatura do *stripper*	x_{44}	MV para vazão de alimentação A (Fluxo 1)
x_{19}	Vazão de vapor do *stripper*	x_{45}	MV para vazão de alimentação total (Fluxo 4)
x_{20}	Trabalho do compressor	x_{46}	Válvula de reciclagem do compressor
x_{21}	Temperatura de saída da água resfriada no reator	x_{47}	Vazão de expurgo (Fluxo 9)
x_{22}	Temperatura de saída da água resfriada no separador	x_{48}	Vazão do líquido no separador (Fluxo 10)
x_{23}	Componente A (Fluxo 6)	x_{49}	Vazão do produto líquido no *stripper* (Fluxo 11)
x_{24}	Componente B (Fluxo 6)	x_{50}	Válvula de vapor no *stripper*
x_{25}	Componente C (Fluxo 6)	x_{51}	Vazão de água resfriada no reator
x_{26}	Componente D (Fluxo 6)	x_{52}	Vazão de água resfriada no condensador

5.1.3 Processo de escoamento multifásico de cranfiled

A Instalação de Escoamento Multifásico da Universidade de Cranfield (CMFF) Cao [2025], Ruiz-Cárcel et al. [2015] (do inglês, *Cranfield Multiphase Flow Facility*), foi projetada para fornecer um fluxo controlado de água, óleo e ar para um sistema pressurizado. A Figura 23 apresenta um esquema simplificado da instalação e a Tabela 4 relaciona as 24 variáveis do processo. O sistema inclui tubulações com diferentes diâmetros e geometrias, um separador bifásico de gás e líquido (0.5 m de diâmetro e 1.2 m de altura) no topo de uma plataforma de 10.5 m de altura e um separador trifásico horizontal de 11 m^3 ao nível do solo. O ar retorna para a atmosfera, enquanto óleo e água são separados em coalescedores antes de serem armazenados em tanques com capacidade de 12.5 m^3 cada.

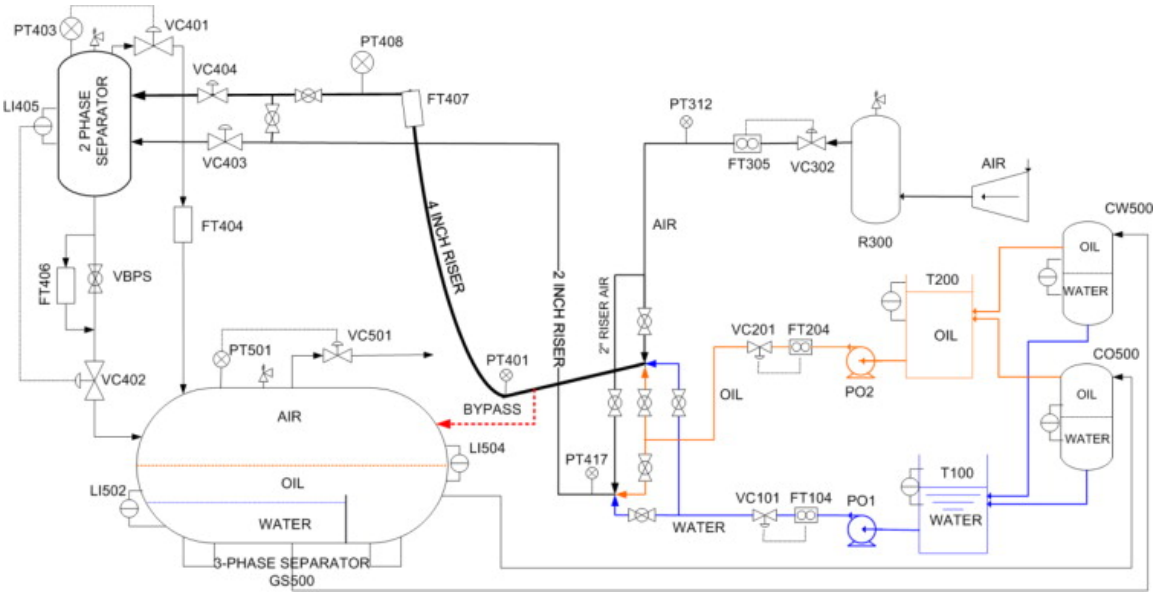


Figura 23 – Esquema geral da Instalação de Escoamento Multifásico da Universidade de Cranfield.

Fonte: Ruiz-Cárcel et al. [2015].

O ar é fornecido por dois compressores, capazes de gerar até 0.4 m^3/s a 0.7 MPa, sendo armazenado em um reservatório de 8 m^3 para estabilizar a pressão. Antes de passar pelo medidor de fluxo (FT305) e pela válvula pneumática (VC302), o ar é filtrado e resfriado. A água e o óleo são bombeados separadamente por bombas *Grundfos* CR90-5 (PO1 e PO2) e têm seu fluxo controlado pelas válvulas VC101 e VC201.

Os fluidos podem escoar por dois circuitos distintos: uma tubulação de 4", em que o símbolo " representa a unidade de polegadas, com 55 m de comprimento e inclinação descendente de 2° conectada a um *riser* de 10,5 m de altura ou uma tubulação de 2" com 40 m de comprimento ligada a um *riser* vertical de 10.5 m. Durante os testes do benchmark, a linha de 4" foi usada, exceto em um caso específico de falha (falha 6), que envolveu a linha de 2". Há também uma linha alternativa de 4" que permite desviar o fluxo diretamente para o separador trifásico, sem passar pelo *riser* e pelo separador superior.

Sensores monitoram a pressão em vários pontos do sistema, incluindo antes do ponto de mistura (PT312), no fundo (PT401) e topo do riser (PT408), dentro do separador superior (PT403) e dentro do separador trifásico (PT501). O fluxo na linha de 4" é medido por FT407, que também registra densidade e temperatura. O separador superior conta com medição de nível (LI405) e controle de saída por VC402. Já a pressão do separador trifásico é regulada por VC501, enquanto os níveis das fases líquidas são monitorados por LI502 e LI504. Além disso, há medição de pressão na linha de 2" no fundo do riser (PT417).

Tabela 4 – Variáveis do processo de instalação de escoamento multifásico.

ID	Localização	Descrição	Unidade
1	PT312	Pressão de entrega de ar	MPa
2	PT401	Pressão no fundo do riser	MPa
3	PT408	Pressão no topo do riser	MPa
4	PT403	Pressão no separador superior	MPa
5	PT501	Pressão no separador trifásico	MPa
6	PT408	Pressão diferencial (PT401-PT408)	MPa
7	PT403	Pressão diferencial sobre VC404	MPa
8	FT305	Vazão de entrada de ar	m^3/s
9	FT104	Vazão de entrada de água	kg/s
10	FT407	Vazão no topo do riser	kg/s
11	LI405	Nível no separador superior	m
12	FT406	Vazão na saída do separador superior	kg/s
13	FT407	Densidade no topo do riser	kg/m^3
14	FT406	Densidade na saída do separador superior	kg/m^3
15	FT104	Densidade da água na entrada	kg/m^3
16	FT407	Temperatura no topo do riser	°C
17	FT406	Temperatura na saída do separador superior	°C
18	FT104	Temperatura da água na entrada	°C
19	LI504	Nível gás-líquido no separador trifásico	%
20	VC501	Posição da válvula VC501	%
21	VC302	Posição da válvula VC302	%
22	VC101	Posição da válvula VC101	%
23	PO1	Corrente da bomba de água	A
24	PT417	Pressão na zona de mistura linha 2"	MPa

No processo, é possível simular um total de seis modos operacionais que representam falhas, que podem ocorrer de forma incipiente, intermitente ou abrupta. A Tabela 5 relaciona as possíveis falhas operacionais, bem como o respectivo modo de ocorrência (referenciando os conceitos apresentados na Subseção 2.1). Os dados de simulação utilizados nos experimentos foram obtidos em Cao [2025].

Tabela 5 – Falhas possíveis no CMFF.

ID	Descrição	Tipo
1	Bloqueio da linha de ar	Incipiente
2	Bloqueio da linha de água	Incipiente
3	Bloqueio da entrada do separador superior	Incipiente
4	Bypass direto aberto	Incipiente
5	Condições de golfadas	Intermitente
6	Pressurização da linha de 2"	Abrupto

5.2 Rotulação Automática

Para simplificar, o termo *classe* será utilizado para se referir a um único comportamento físico do sistema. Dito isso, considere uma base de dados rotulada denotada pela tupla (X, Y), em que $\mathrm{X} \in \mathbb{R}^{k \times n}$ são os dados de entrada e $\mathrm{Y} \in \mathcal{C}^k$ os rótulos ou classes/comportamentos/operações a que cada amostra se relaciona, com $\mathcal{C} = \{1, 2, \ldots, c\}$ e c denotando o total de classes.

O procedimento proposto se baseia na premissa de funcionamento de um processo real, em que os dados de cada comportamento do sistema são temporalmente relacionados. Logo, assume-se que os dados de uma mesma classe surgem de forma consecutiva. Considerando tal premissa é possível estabelecer que: se a primeira ocorrência de uma classe i é delimitada pelo instante inicial t_0^i e final t_1^i ($t_0^i < t_1^i$), idealmente, o método deve criar novos modelos neste intervalo para representá-la.

Portanto, a partir da saída do método contendo os identificadores dos modelos a qual cada entrada foi associada, denotada por $\tilde{Y} \in \mathrm{M}^k$, com $\mathrm{M} = \{1, 2, \ldots, m\}$ e m denota o total de modelos. Para efetuar a rotulação automática para cada classe i é suficiente avaliar quais os modelos presentes no intervalo $[t_0^i, t_1^i]$ que não ocorreram previamente. Estes modelos são aqueles criados em função do surgimento da respectiva classe. Logo, a ocorrência de qualquer um desses modelos implica a ocorrência da classe i, uma vez que o mapeamento é exclusivo, isto é, modelos já associados a uma classe são desconsiderados em novas associações.

Esse procedimento pode ser formalmente representado como:

$$\hat{y}_t = r(\tilde{y}_t; \mathrm{Y}, \tilde{Y}) = i \mid \tilde{y}_t \in \mathcal{M}_i, \quad i = \{1, 2, \ldots, c\} \tag{5.1}$$

$$\mathcal{M}_i = \mathrm{U}\left(\tilde{Y}_{[t_0^i, t_1^i]}\right) - \mathrm{U}\left(\underline{\tilde{Y}}^{i,j}\right), \tag{5.2}$$

em que $t \in \{1, 2, \ldots, k\}$; $\hat{y}_t \in \mathcal{M}$ é a saída após o pré-processamento; $\tilde{y}_t \in \tilde{Y}$; $\mathcal{M}_i \subseteq \mathrm{M}$ representa o conjunto com os identificadores dos modelos associados a classe i, tal que $\mathrm{M} = \bigcup_{i=1}^{c} \mathcal{M}_i$ e $\bigcap_{i=1}^{c} \mathcal{M}_i = \emptyset$; $\mathrm{U}(\cdot)$ representa uma função que retorna um conjunto com

os elementos únicos do vetor dado como argumento; $\tilde{Y}_{[t_0^i,t_1^i]}$ denota um vetor com os rótulos de cada amostra referente ao período do primeiro surgimento da classe i; $-$ representa um operador de diferença entre conjuntos; e $\underline{\tilde{Y}}^{i,j}$ denota um vetor com os rótulos de todo o período anterior a t_0^i.

5.3 Metodologia Experimental

A aplicação dos métodos nos *bechmarks* descritos ocorreu individualmente, organizando as classes de cada um sequencialmente a partir da classe normal. A Tabela 6 relaciona a ordem, classes e quantidade de amostras utilizadas. No 3W, devido a limitações de recursos computacionais, os dados foram subamostrados a uma taxa de 2 minutos para a classe normal e 30 segundos para as anomalias. Foram utilizados apenas 11 mil amostras da classe normal após a subamostragem. Além disso, devido a qualidade (considerando ausência, congelamento, etc.) dos dados, os mesmos foram selecionados manualmente mesclando poços que apresentavam maior qualidade. Após avaliação, a classe Normal e Anomalia 1 foi obtida do Poço 01, a Anomalia 2 e 6 do Poço 02, a Anomalia 5 do Poço 15, a Anomalia 7 do Poço 06 e a Anomalia 8 do Poço 21. Com relação aos dados da base de dados TEP, os mesmos foram obtidos em Chen [2019], e utilizados sem qualquer modificação. No CMFF, uma subamostragem também foi aplicada, utilizando 10 minutos e 4 minutos para a classe normal e falhas, respectivamente. A variável 24, *Pressão na zona de mistura linha 2"*, foi descartada por não apresentar variabilidade em todo o período de dados considerado.

Apenas na base 3W foi aplicado o seguinte preprocessamento considerando todos os dados organizados sequencialmente (conforme Tabela 6): i) Dados faltantes foram preenchidos com o último válido; ii) Dados ainda ausentes após (i) foram preenchidos com 0; iii) Por último, um ruído gaussiano com média 0 e variância 0.05 foi adicionado a cada amostra e coordenada como forma de tratamento a dados congelados presentes na base de dados, que por sua vez podem ser devido a falha de comunicação.

Para avaliar quantitativamente foram utilizada as métricas de Precisão, Revocação e F1-score. Tais métricas foram calculadas considerando cada classe (perspectiva binária em que as demais classes compõem uma só), sendo as mesmas definidas conforme a seguir [Bishop and Nasrabadi, 2006]:

$$\text{Precisão} = \frac{\text{VP}}{\text{VP} + \text{FP}}, \tag{5.3}$$

$$\text{Revocação} = \frac{\text{VP}}{\text{VP} + \text{FN}}, \tag{5.4}$$

Benchmark	**ID**	**Classe**	**Amostras × Variáveis**
3W	0	*Normal*	1100 × 5
	1	*Anomalia 1*	359 × 5
	2	*Anomalia 2*	305 × 5
	5	*Anomalia 5*	632 × 5
	6	*Anomalia 6*	1596 × 5
	7	*Anomalia 7*	6514 × 5
	8	*Anomalia 8*	807 × 5
TEP	0	*Normal*	960 × 52
	6	*Falha 6*	960 × 52
	18	*Falha 18*	960 × 52
CMFF	0	*Normal*	1038 × 23
	1	*Falha 1*	904 × 23
	2	*Falha 2*	1094 × 23
	3	*Falha 3*	1805 × 23
	4	*Falha 4*	1336 × 23
	5	*Falha 5*	121 × 23
	6	*Falha 6*	270 × 23

Tabela 6 – Organização de classes e quantidade de amostras, por *benchmark*, considerada nos experimentos. Tal organização representa a ordem em que os dados foram apresentados aos métodos.

$$\text{F1\_score} = 2 \cdot \frac{\text{Precisão} \cdot \text{Revocação}}{\text{Precisão} + \text{Revocação}}, \tag{5.5}$$

em que VP, FP, FN denotam a quantidade de Verdadeiros Positivos, Falsos Positivos e Falsos Negativos, respectivamente. Logo, neste contexto a Precisão indicará qual a chance do método indicar a classe alvo e de fato ser; a Revocação indicará, considerando todas as instâncias da classe alvo, o quão bem o método a está discriminando; e o F1-score é a média harmônica da precisão e revocação, utilizada para resumir a performance do método. Vale lembrar que a saída do método não é utilizada diretamente para computar as métricas, utiliza-se a saída do procedimento de rotulagem automática.

Com relação aos métodos considerados como *baseline*, foram avaliadas as seguintes abordagens que fazem parte do estado-da-arte em aprendizado de evolutivo (na perspectiva adotada nesta tese) em fluxos de dados:

- AutoCloud [Bezerra et al., 2020]
- OEC [Moshtaghi et al., 2016a]
- MicroTEDAClus [Maia et al., 2020]
- SOStream [Isaksson et al., 2012]
- MacroSOStream [Oliveira et al., 2022]

- CEDAS [Hyde et al., 2016]
- MCMSTStream [Erdinç et al., 2024]

Com relação as abordagens propostas, foram consideradas as seguintes configurações:

- eSBM4Clus: Método de clusterização.
- eRBM-eGAE: eRBM utilizando o eGAE como modelo evolutivo e os seus grânulos como modelos discriminativos.

Por fim, o processo de parametrização de todas a abordagens foi realizado por meio de uma busca em grade. Para o ranqueamento nessa busca foi utilizado a métrica F1-score calculada após cada método observar todo o conjunto de dados uma única vez. Para cada parâmetro considerado na busca, foram gerados cinco valores uniformemente espaçados em um intervalo predefinido. A listagem dos valores considerados é dada na Tabela 7. Os hiperparâmetros que não tiverem um intervalo como valor na coluna Valor foram mantidos fixos. A arquitetura utilizada no eGAE foi a mesma utilizada na Subseção 3.2.1, sendo: **Encoder**: `Linear (3, 512) -> ReLU() -> Linear(512, 3)`; e, **Decoder**: `Linear (2, 512) -> ReLU() -> Linear(512, 3)`.

Uma última informação diz respeito a normalização. Os dados foram normalizados para o intervalo $[0, 1]$ para facilitar a parametrização dos métodos sensíveis à escala. Uma exceção foi o OEC, em que não foi aplicada normalização em nenhuma base de dados; e o eSBM4Clus, em que não foi aplicada normalização na base 3W. Uma visão geral da configuração experimental é dada por meio do fluxograma da Figura 24, e foi executado considerando cada método individualmente.

5.4 Resultados e Discussões

5.4.1 Qualidade de detecção e classificação

Os resultados obtidos estão dispostos na Tabela 8 para o 3W dataset, na Tabela 9 para o TEP e na Tabela 10 para o CMFF. Perceba que com relação ao 3W, os métodos tiveram grande dificuldade em discriminar a classe normal das anomalias. A razão por trás disso é que grande parte das anomalias são caracterizadas por uma mudança incremental, logo amostras referente ao início da anomalia ainda estão bem próximas espacialmente da classe normal. Ademais, algumas anomalias são bem semelhantes, como é possível observar por meio da representação latente gerada com eGAE, e apresentada na Figura 25; na figura, algumas anomalias (Anomalia 1, 2, 5 e 6) ocupam a mesma região, indicando que

Tabela 7 – Parametrização definida na busca em grade.

Método	**Parâmetro**	**Valor**
Autocloud	`m`	$[1.5, 5]$
MicroTEDAClus	`r0`	$[0.001, 0.1]$
OEC	`gamma_normal`	$[0.5, 0.99]$
	`gamma_gzone`	`gamma_normal` $+0.009$
	`stab_period`	$[0, 50]$
	`forgetting_factor`	$[0.1, 0.99]$
eSBM4Clus	`w`	$[20, 100]$
	`beta`	$[0.1, 0.999]$
	`gamma`	0.1
	`l_max`	200
CEDAS	`min_samples`	$[2, 20]$
	`radius`	$[0.001, 0.5]$
	`decay`	$[100, 2000]$
MacroSOStream	`min_pts`	$[1, 20]$
	`alpha`	$[0.01, 0.9]$
	`merge_threshold`	$[0.001, 0.5]$
	`p`	$[1, 10]$
SOSTream	`min_pts`	$[2, 20]$
	`alpha`	$[0.01, 0.9]$
	`merge_threshold`	$[0.001, 0.5]$
MCMSTStream	`min_samples`	$[2, 20]$
	`sliding_window_size`	$[100, 1000]$
	`min_micros`	$[2, 20]$
	`radius`	$[0.001, 0.5]$
eRBM-eGAE	`min_samples`	$[2, 20]$
	`p`	$[0.01, 3]$
	`upsilon`	$[0.01, 1]$
	`w`	$[20, 200]$
	`batch_size`	100
	`tau`	10
	`delta`	2.0
	`lambda1`	0.1
	`lambda2`	0.9
	`n_iter`	10
	`learning_rate`	0.01

possivelmente estão bem próximas no espaço original. Ainda na figura, percebe-se que a maioria das classes possuem diferentes condições de operação, ficando localizadas em diferentes regiões do espaço latente. Isso também pode ser utilizado como uma justificativa ao péssimo desempenho da grande maioria dos métodos, que possuem limitações de representação de regiões arbitrárias e/ou disjuntas, sendo muito sensíveis a parametrização. De forma geral, as abordagens propostas apresentaram bom desempenho, em especial o eSBM4Clus, que foi o único a apresentar 100% de acerto na detecção (revocação) e classificação (precisão) de todas as classes. Alguns outros métodos, como o OEC,

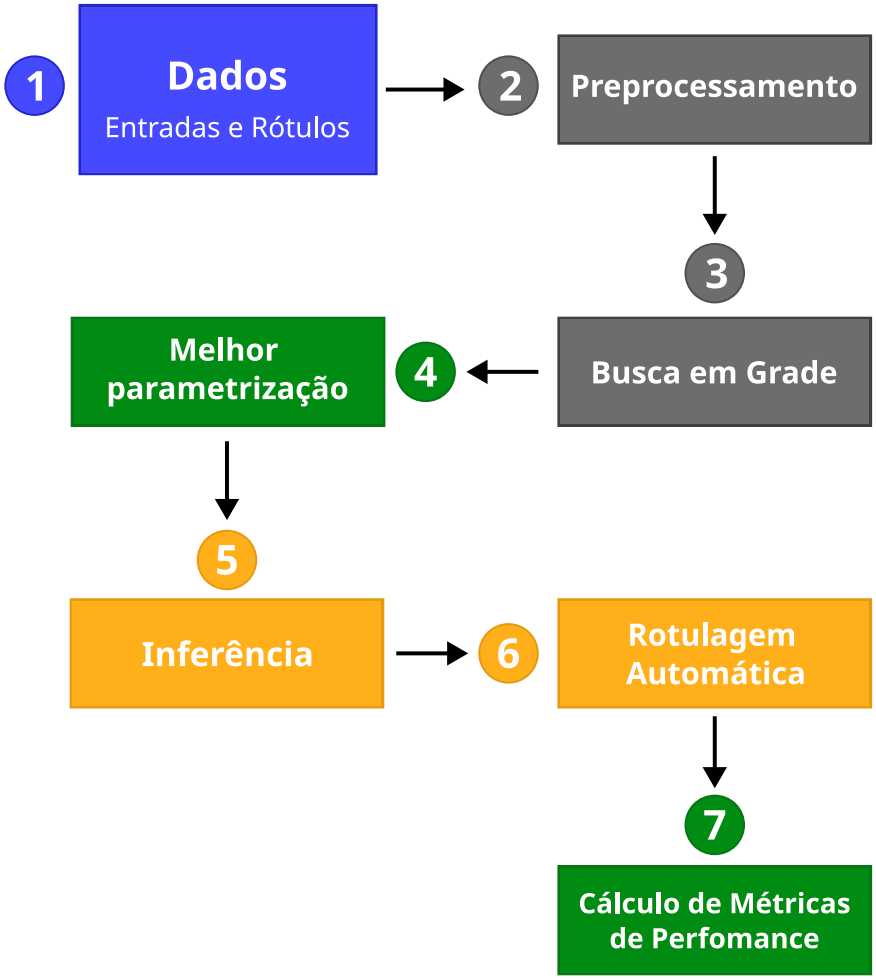


Figura 24 – Fluxograma da configuração experimental utilizada. Esse fluxo é aplicado considerando cada método evolutivo individualmente.

MicroTEDAClus apresentaram bom desempenho considerando as anomalias 1, 7 e 8, desempenho razoável para as anomalias 5 e 6, e péssimo desempenho para as demais classes. Todos os demais métodos apresentaram, de uma forma geral, péssimo desempenho.

Tabela 8 – Resultados obtidos para base 3W. Os melhores (por operação e métrica) estão destacados em negrito.

Classe ID	**Métrica**	**Métodos**								
		AutoCloud	**CEDAS**	**eRBM-eGAE**	**eSBM4Clus**	**MacroSOStream**	**MCMSTStream**	**MicroTEDAClus**	**OEC**	**SOSTream**
0	Precisão	0.282	0.438	0.746	**1.000**	0.229	0.301	0.418	0.524	0.304
	Revocação	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	F1-score	0.440	0.610	0.854	**1.000**	0.373	0.463	0.590	0.688	0.466
1	Precisão	0.000	0.210	0.603	**1.000**	0.000	0.000	0.821	**1.000**	0.000
	Revocação	0.000	**1.000**	**1.000**	**1.000**	0.000	0.000	0.777	0.891	0.000
	F1-score	0.000	0.347	0.753	**1.000**	0.000	0.000	0.798	0.943	0.000
2	Precisão	**1.000**	0.000	0.053	**1.000**	0.000	0.000	0.089	**1.000**	0.000
	Revocação	0.193	0.000	0.213	**1.000**	0.000	0.000	0.161	**1.000**	0.000
	F1-score	0.324	0.000	0.085	**1.000**	0.000	0.000	0.115	**1.000**	0.000
5	Precisão	**1.000**	0.985	0.829	**1.000**	0.000	0.325	**1.000**	**1.000**	**1.000**
	Revocação	0.503	0.318	0.698	**1.000**	0.000	0.590	0.650	0.587	0.590
	F1-score	0.669	0.481	0.758	**1.000**	0.000	0.419	0.788	0.740	0.742
6	Precisão	0.000	0.000	0.517	**1.000**	0.000	0.000	**1.000**	**1.000**	0.000
	Revocação	0.000	0.000	0.114	**1.000**	0.000	0.000	0.081	0.625	0.000
	F1-score	0.000	0.000	0.187	**1.000**	0.000	0.000	0.151	0.769	0.000
7	Precisão	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	Revocação	**1.000**	**1.000**	0.998	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	F1-score	**1.000**	**1.000**	0.999	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
8	Precisão	**1.000**	**1.000**	**1.000**	**1.000**	0.000	0.000	**1.000**	**1.000**	**1.000**
	Revocação	0.644	0.462	0.789	**1.000**	0.000	0.000	0.917	0.876	**1.000**
	F1-score	0.784	0.632	0.882	**1.000**	0.000	0.000	0.957	0.934	**1.000**
Média	**Precisão**	0.612	0.519	0.678	**1.000**	0.176	0.232	0.761	0.932	0.472
	Revocação	0.477	0.540	0.687	**1.000**	0.286	0.370	0.655	0.854	0.513
	F1-score	0.460	0.439	0.645	**1.000**	0.196	0.269	0.628	0.868	0.458

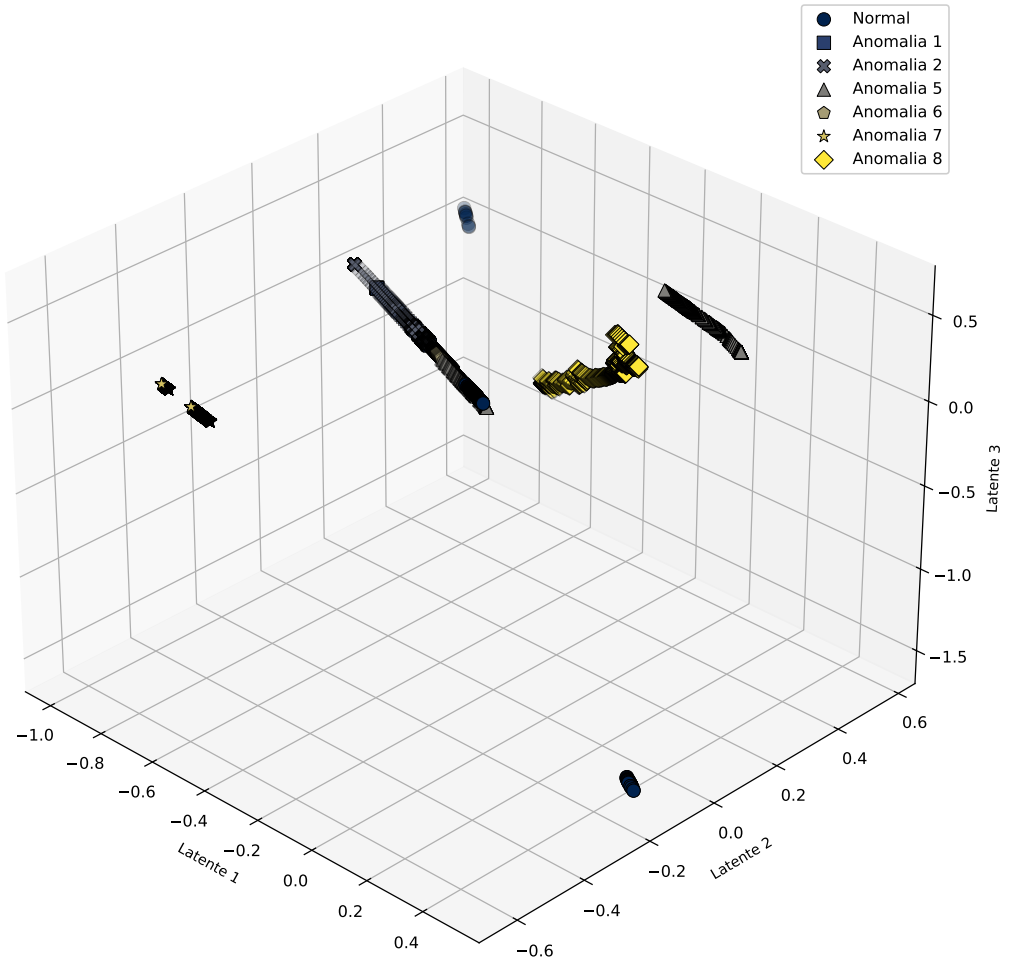


Figura 25 – Representação latente tridimensional obtida com o eGAE para a base 3W.

Com relação aos resultado do TEP, percebe-se uma situação similar ao anterior no que diz respeito a classe normal. A razão aqui é que a mudança que caracteriza a falha é muito suave, além disso, a fase inicial da falha está sobreposta a região de normalidade, como observada na representação latente apresentada na Figura 26; justificando assim a baixa taxa de identificação em todos os métodos para a classe normal. No que diz respeito ao desempenho na detecção e identificação das falhas, a maioria dos métodos apresentaram bom desempenho, apenas o SOSTream e MacroSOStream apresentaram péssimo desempenho para as falhas 6 e 18, respectivamente. O péssimo desempenho se deu por conta da i) incapacidade de representar formatos não lineares, e possivelmente pela ii) parametrização avaliada ter sido insuficiente. Vale ressaltar, que as abordagens propostas, eSBM4Clus e eRBM-eGAE, foram aquelas que apresentaram, em conjunto, melhor precisão e revocação considerando todas as classes.

Tabela 9 – Resultados obtidos para a base TEP. Os melhores (por operação e métrica) estão destacados em negrito.

Classe ID	**Métrica**	**Métodos**								
		AutoCloud	**CEDAS**	**eRBM-eGAE**	**eSBM4Clus**	**MacroSOStream**	**MCMSTStream**	**MicroTEDAClus**	**OEC**	**SOSTream**
0	Precisão	0.616	0.630	0.684	**0.716**	0.626	0.603	0.472	0.673	0.423
	Revocação	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	F1-score	0.762	0.773	0.812	**0.835**	0.770	0.753	0.641	0.804	0.595
6	Precisão	**1.000**	0.971	0.997	**1.000**	0.567	**1.000**	**1.000**	**1.000**	0.000
	Revocação	0.759	0.768	**0.799**	0.730	0.796	0.739	0.583	0.786	0.000
	F1-score	0.863	0.857	**0.887**	0.844	0.662	0.850	0.737	0.880	0.000
18	Precisão	**1.000**	**1.000**	**1.000**	**1.000**	0.000	**1.000**	**1.000**	**1.000**	**1.000**
	Revocação	0.617	0.623	0.736	**0.874**	0.000	0.604	0.297	0.727	0.638
	F1-score	0.763	0.768	0.848	**0.933**	0.000	0.753	0.458	0.842	0.779
Média	**Precisão**	0.872	0.867	0.894	**0.905**	0.398	0.868	0.824	0.891	0.474
	Revocação	0.792	0.797	0.845	**0.868**	0.599	0.781	0.627	0.838	0.546
	F1-score	0.796	0.799	0.849	**0.871**	0.477	0.785	0.612	0.842	0.458

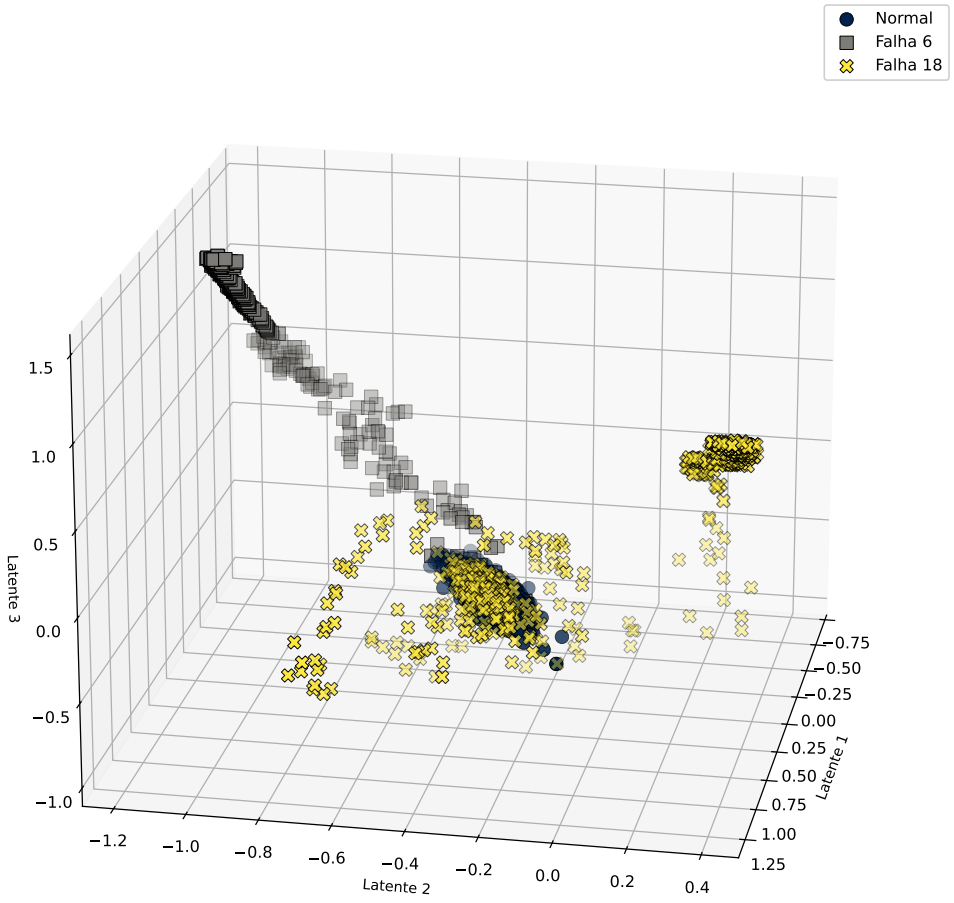


Figura 26 – Representação latente tridimensional obtida com o eGAE para a base TEP.

Por fim, os resultados dos métodos no CMFF foram bem destoantes do observado nos benchmarks anteriores. Grande maioria dos métodos, com exceção aos propostos, apresentaram péssimo desempenho. Uma possível justificativa seria a grande não linearidade do processo, e existência de diferentes regimes de funcionamento para a mesma classe, como observado pelo espaço latente apresentado na Figura 27. Os únicos métodos que apresentaram bom desempenho para todas as classes, foram as abordagens propostas; o MicroTEDAClus apresentou desempenho razoável para a classe Normal, Falha 1, Falha 2, Falha 3 e Falha 6; e, o SOSTream para a classe Normal e Falha 6. Os demais, de forma geral, apresentaram péssimo desempenho.

Pelos resultados numéricos, observa-se que as abordagens propostas apresentaram, na maioria dos casos apresentados, performance superior ou equivalente aos métodos considerados como referência. Na próxima subseção uma discussão em função do consumo de recursos computacionais é apresentado para cada método e em cada base de dados.

Tabela 10 – Resultados obtidos para a base CMFF. Os melhores (por operação e métrica) estão destacados em negrito.

Classe ID	Métrica	Métodos								
		AutoCloud	**CEDAS**	**eRBM-eGAE**	**eSBM4Clus**	**MacroSOStream**	**MCMSTStream**	**MicroTEDAClus**	**OEC**	**SOSTream**
0	Precisão	0.160	0.443	0.939	**1.000**	0.165	0.158	0.739	0.167	0.815
	Revocação	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	F1-score	0.276	0.614	0.969	**1.000**	0.283	0.273	0.850	0.287	0.898
1	Precisão	0.000	0.256	0.946	0.823	**1.000**	0.000	0.600	0.000	0.185
	Revocação	0.000	0.980	0.926	**1.000**	0.147	0.000	**1.000**	0.000	0.929
	F1-score	0.000	0.406	**0.936**	0.903	0.257	0.000	0.750	0.000	0.309
2	Precisão	**1.000**	0.000	0.931	0.874	0.000	0.000	0.788	**1.000**	0.000
	Revocação	0.083	0.000	0.956	**0.998**	0.000	0.000	0.833	0.148	0.000
	F1-score	0.154	0.000	**0.944**	0.932	0.000	0.000	0.810	0.258	0.000
3	Precisão	0.000	**1.000**	0.967	**1.000**	0.000	0.000	0.981	0.000	0.000
	Revocação	0.000	0.421	0.958	0.895	0.000	0.000	**0.984**	0.000	0.000
	F1-score	0.000	0.593	0.963	0.945	0.000	0.000	**0.983**	0.000	0.000
4	Precisão	**1.000**	0.000	0.931	**1.000**	0.000	0.000	**1.000**	**1.000**	**1.000**
	Revocação	0.001	0.000	**0.956**	0.879	0.000	0.000	0.322	0.056	0.294
	F1-score	0.003	0.000	**0.943**	0.936	0.000	0.000	0.487	0.106	0.455
5	Precisão	0.000	**1.000**	0.275	**1.000**	**1.000**	0.000	0.000	0.445	**1.000**
	Revocação	0.000	0.066	0.207	**1.000**	**1.000**	0.000	0.000	0.471	0.884
	F1-score	0.000	0.124	0.236	**1.000**	**1.000**	0.000	0.000	0.458	0.939
6	Precisão	0.000	0.000	**1.000**	**1.000**	**1.000**	0.000	**1.000**	0.000	**1.000**
	Revocação	0.000	0.000	0.756	**1.000**	0.026	0.000	0.956	0.000	0.941
	F1-score	0.000	0.000	0.861	**1.000**	0.051	0.000	0.977	0.000	0.969
Média	Precisão	0.309	0.386	0.856	**0.957**	0.452	0.023	0.730	0.373	0.571
	Revocação	0.155	0.352	0.823	**0.967**	0.310	0.143	0.728	0.239	0.578
	F1-score	0.062	0.248	0.836	**0.959**	0.227	0.039	0.694	0.158	0.510

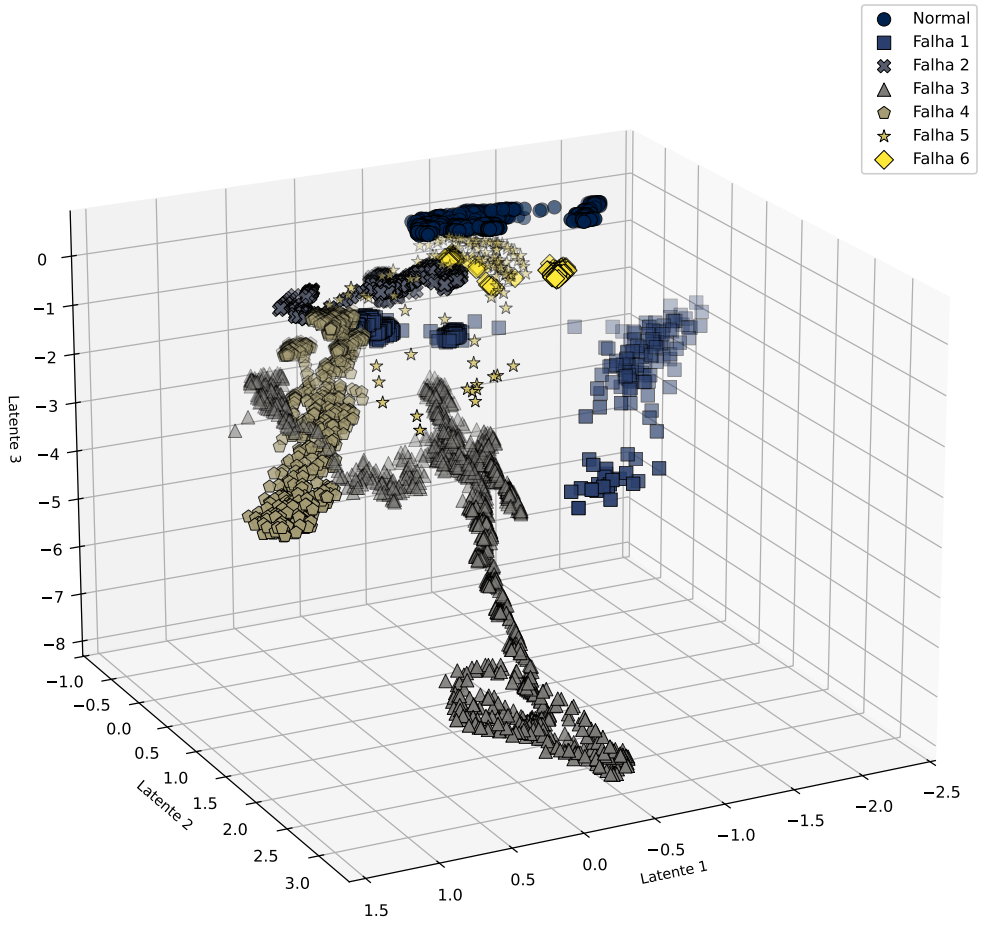


Figura 27 – Representação latente tridimensional obtida com o eGAE para a base CMFF.

5.4.2 Custo computacional

A avaliação dos recursos computacionais apresentada considera duas métricas principais: o tempo de processamento por amostra (em milissegundos) e a quantidade de memória utilizada ao longo da execução do método (em megabytes). Embora essas métricas sejam diretamente influenciadas pela linguagem de programação adotada e pelo *hardware* da máquina utilizada, aqui elas são empregadas como uma medida relativa de uso entre os métodos avaliados. Dessa forma, espera-se que, mesmo com mudanças no ambiente de execução, a proporção de consumo de recursos entre os métodos seja preservada.

Ambas as métricas foram coletadas para todos os métodos ao longo da execução em cada um dos três conjuntos de dados analisados. Para fins de referência, os experimentos foram conduzidos em um laptop equipado com processador AMD Ryzen 7 4800H e 16GB de RAM DDR4 (3200MHz). Todos os métodos foram implementados na linguagem de programação Python, e os resultados reportados correspondem à média de cinco execuções.

A Figura 28 e Figura 29 apresentam os resultados para a base de dados 3W, e a

Tabela 11 resume as informações das figuras em função do valor médio e desvio-padrão. Observa-se que com exceção aos métodos MacroSOSTream e SOSTream, que apresentaram tendência de crescimento expressiva de tempo de processamento e consumo de memória, os demais se mostram estáveis. Um ponto a enfatizar com relação as abordagens propostas, se refere aos comportamentos de picos de consumo do processador. Essa característica se dá no eSBM4Clus devido ao processo de inversão da matriz de intra-similaridade, e no eRBM-eGAE, devido ao treinamento inicial, e após, em batelada do eGAE. Percebe-se, pelos resultados médios, que as duas abordagens estão dentre 5 com menor tempo de processamento.

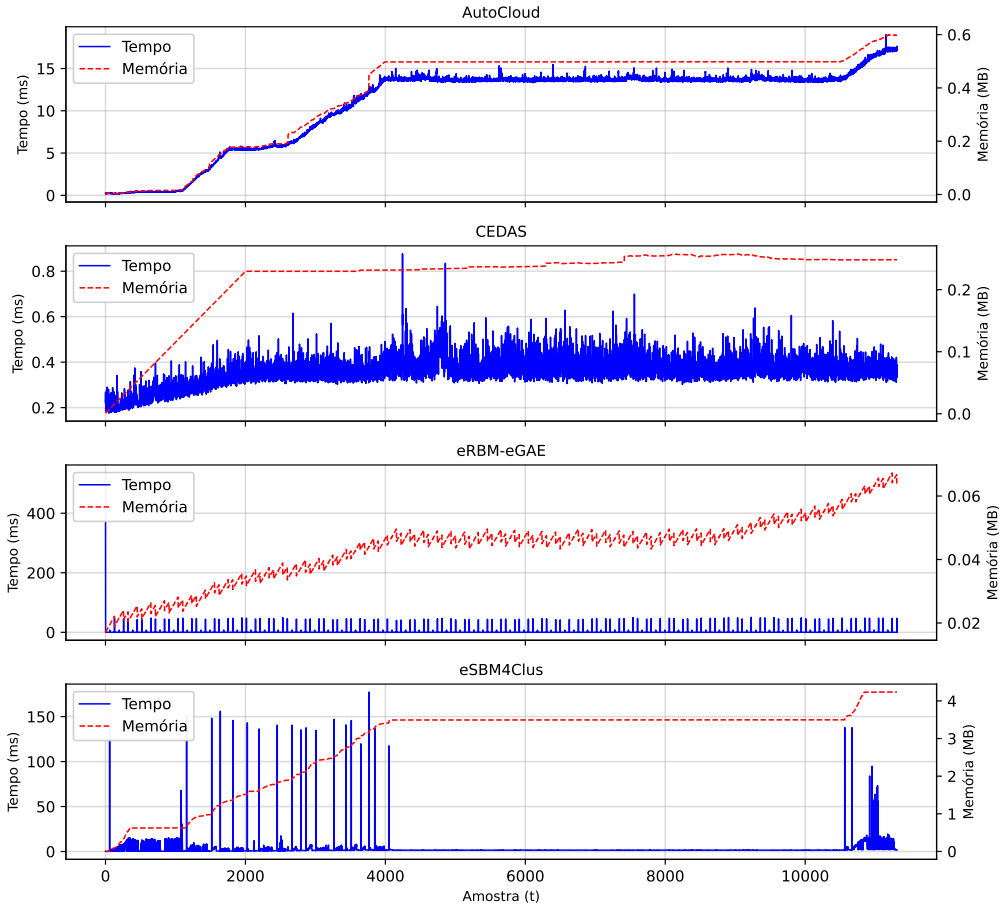


Figura 28 – Tempo de processamento (em milissegundos) e memória (em megabytes) pelos métodos AutoCloud, CEDAS, eRBM-eGAE e eSBM4Clus na base 3W.

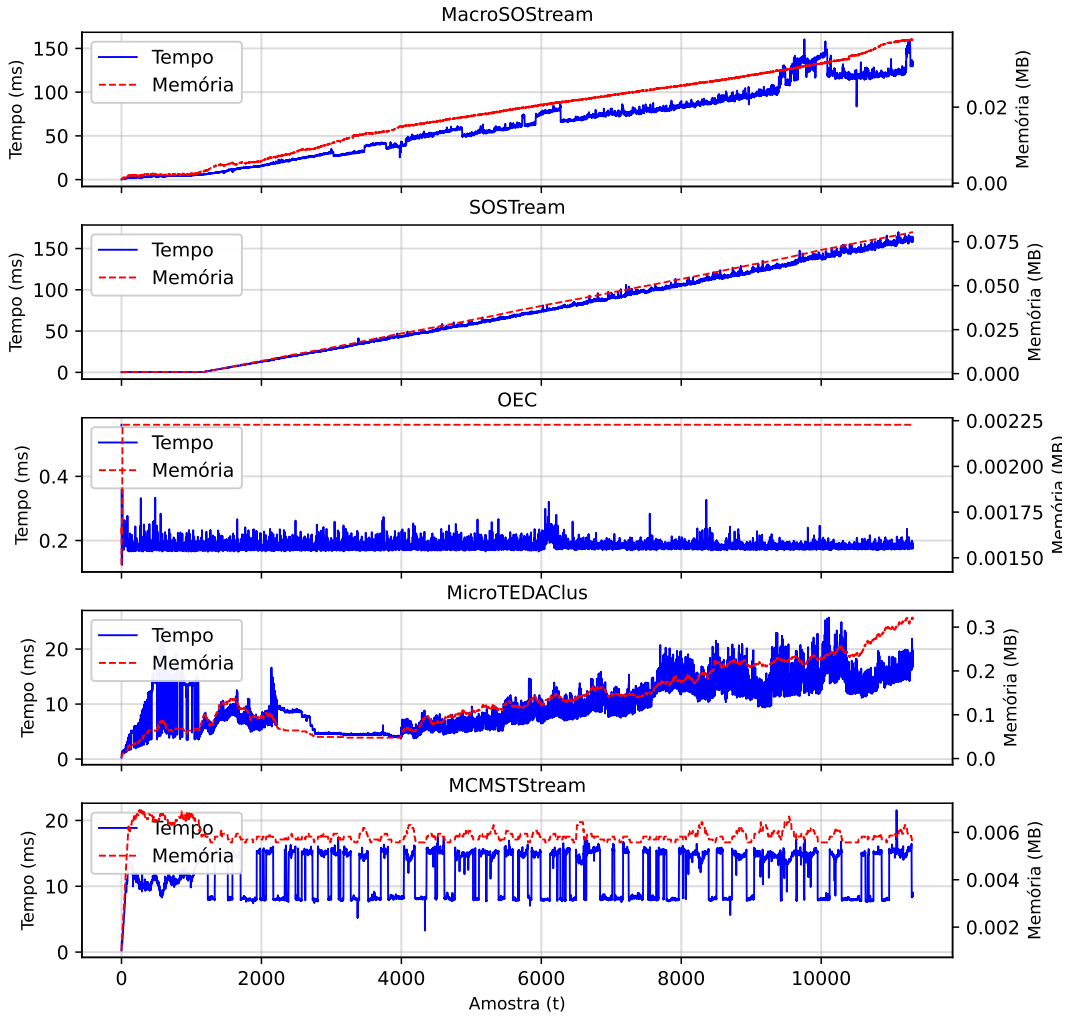


Figura 29 – Tempo de processamento (em milissegundos) e memória (em megabytes) pelos métodos MacroSOStream, SOSTream, OEC, MicroTEDAClus e MCMSTStream na base 3W.

Tabela 11 – Tempo médio de processamento e consumo de memória dos métodos na base 3W. Os valores estão ordenados de forma ascendente em função da coluna de média de tempo.

	Média de Tempo (ms)	**Média de Memória (MB)**
OEC	0.1829 ± 0.0132	0.0022 ± 0.0000
CEDAS	0.3581 ± 0.0601	0.2196 ± 0.0564
eRBM-eGAE	0.6661 ± 6.7798	0.0436 ± 0.0102
eSBM4Clus	2.5143 ± 6.9643	2.8502 ± 1.1098
MicroTEDAClus	10.2901 ± 4.4819	0.1374 ± 0.0736
AutoCloud	10.8241 ± 4.8810	0.3880 ± 0.1789
MCMSTStream	11.9177 ± 3.3575	0.0059 ± 0.0004

Contínua na próxima página

	Média de Tempo (ms)	**Média de Memória (MB)**
MacroSOStream	62.2065 ± 40.0347	0.0187 ± 0.0104
SOSTream	71.0723 ± 50.1084	0.0362 ± 0.0249

Na base de dados TEP, os resultados são apresentados na Figura 30 e Figura 31, e a versão resumida na Tabela 12. Observa-se o mesmo comportamento anterior, porém, por ser um problema com maior dimensão, o tempo de processamento foi mais elevado para a maioria dos métodos. Contudo, ainda estão dentro de um limite aceitável, e pode ser ajustado modificando a parametrização dos métodos; em especial ao eSBM4Clus, que foi bastante impactado pelas atualizações constantes na matriz de intra-similaridade. Como é possível observar na tabela, o eRBM-eGAE apresentou o menor tempo médio de processamento, e o eSBM4Clus o terceiro método com menor consumo médio de memória.

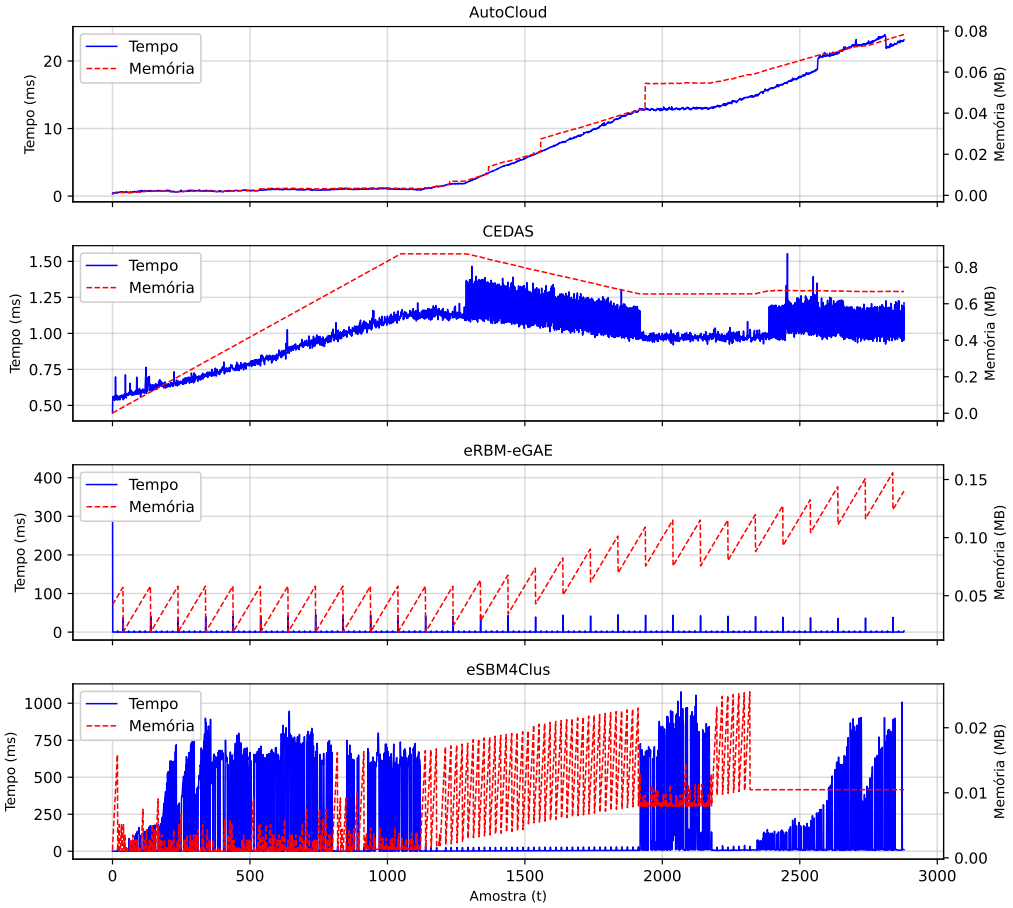


Figura 30 – Tempo de processamento (em milissegundos) e memória (em megabytes) pelos métodos AutoCloud, CEDAS, eRBM-eGAE e eSBM4Clus no TEP.

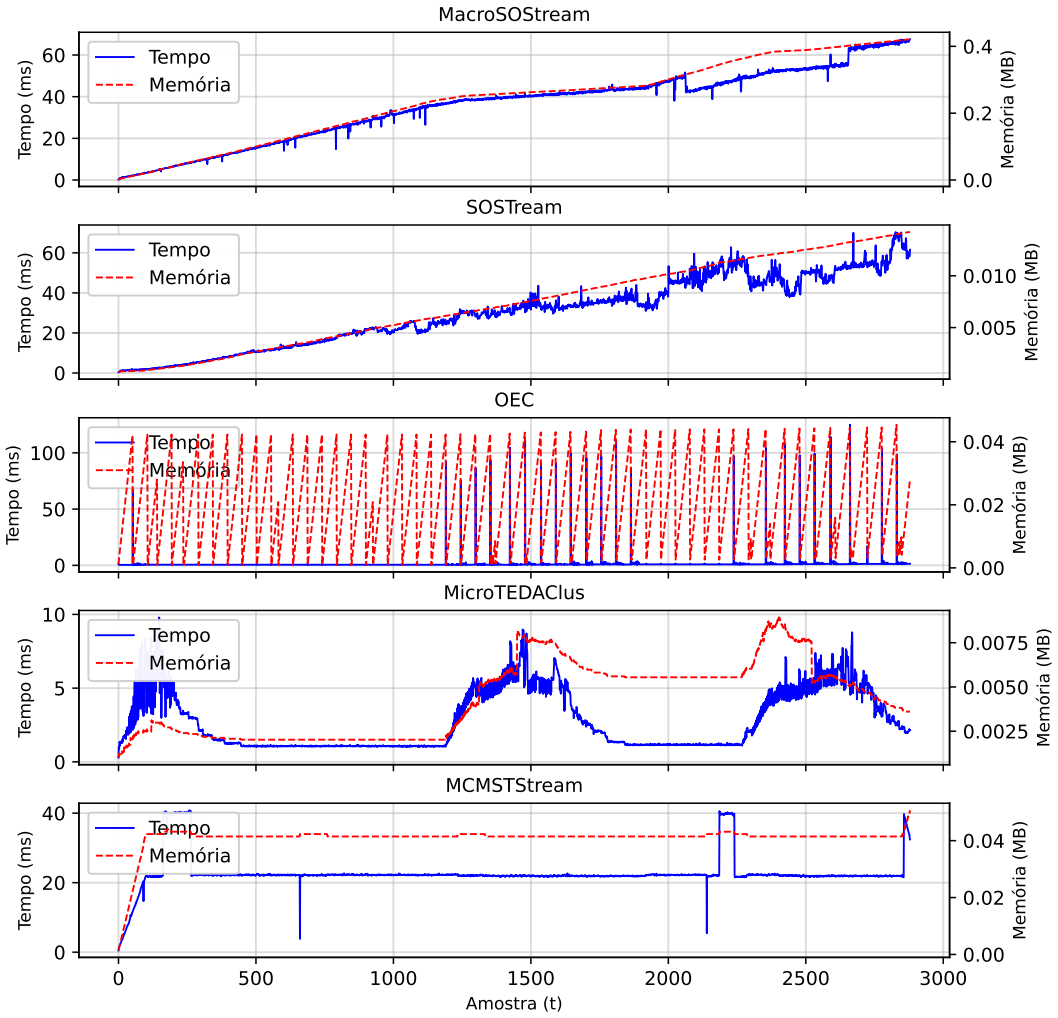


Figura 31 – Tempo de processamento (em milissegundos) e memória (em megabytes) pelos métodos MacroSOStream, SOSTream, OEC, MicroTEDAClus e MCMSTStream no TEP.

Tabela 12 – Tempo médio de processamento e consumo de memória dos métodos na base TEP. Os valores estão ordenados de forma ascendente em função da coluna de média de tempo.

	Média de Tempo (ms)	**Média de Memória (MB)**
eRBM-eGAE	0.8086 ± 8.7242	0.0689 ± 0.0356
CEDAS	0.9852 ± 0.1844	0.6197 ± 0.2157
OEC	1.6560 ± 8.7916	0.0213 ± 0.0124
MicroTEDAClus	2.8407 ± 2.0095	0.0043 ± 0.0021
AutoCloud	7.7295 ± 7.5244	0.0280 ± 0.0271
MCMSTStream	22.7701 ± 4.8925	0.0410 ± 0.0044
SOSTream	30.0765 ± 17.2093	0.0073 ± 0.0041

Contínua na próxima página

	Média de Tempo (ms)	**Média de Memória (MB)**
MacroSOStream	35.8286 ± 17.1950	0.2403 ± 0.1189
eSBM4Clus	150.4218 ± 273.7040	0.0080 ± 0.0058

Por último, os resultados para o CMFF são apresentados nas Figuras Figura 32 e Figura 33, bem como na Tabela Tabela 13. Observa-se um comportamento semelhante ao analisado anteriormente, especialmente em relação ao eSBM4Clus, que apresentou picos de tempo de processamento em torno de 8 segundos, devido, novamente, à inversão da matriz de intra-similaridade, que se configura como o principal gargalo do método.

Vale destacar que, em média, o tempo de processamento do eSBM4Clus foi de aproximadamente 30 milissegundos, um valor viável para a maioria das aplicações práticas. Os demais métodos apresentaram um tempo médio de processamento inferior a 15 milissegundos, com consumo médio de memória abaixo de 0.3 MB.

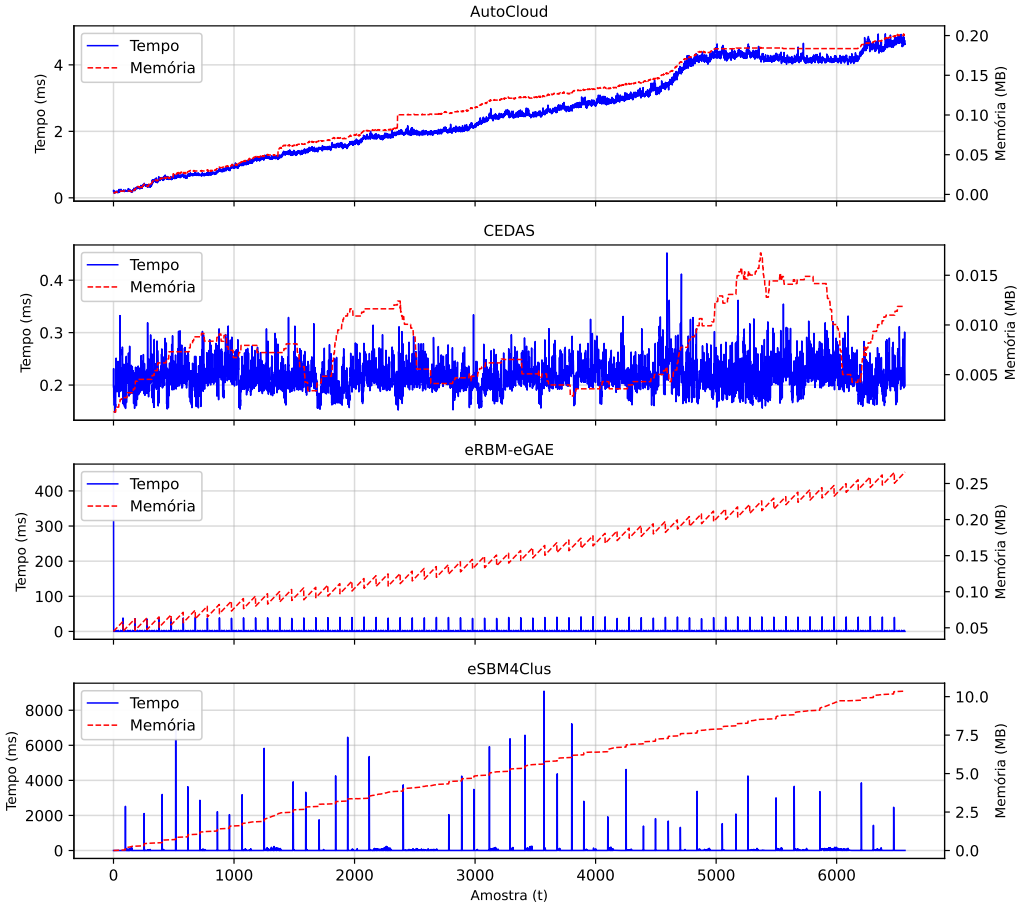


Figura 32 – Tempo de processamento (em milissegundos) e memória (em megabytes) pelos métodos AutoCloud, CEDAS, eRBM-eGAE e eSBM4Clus no CMFF.

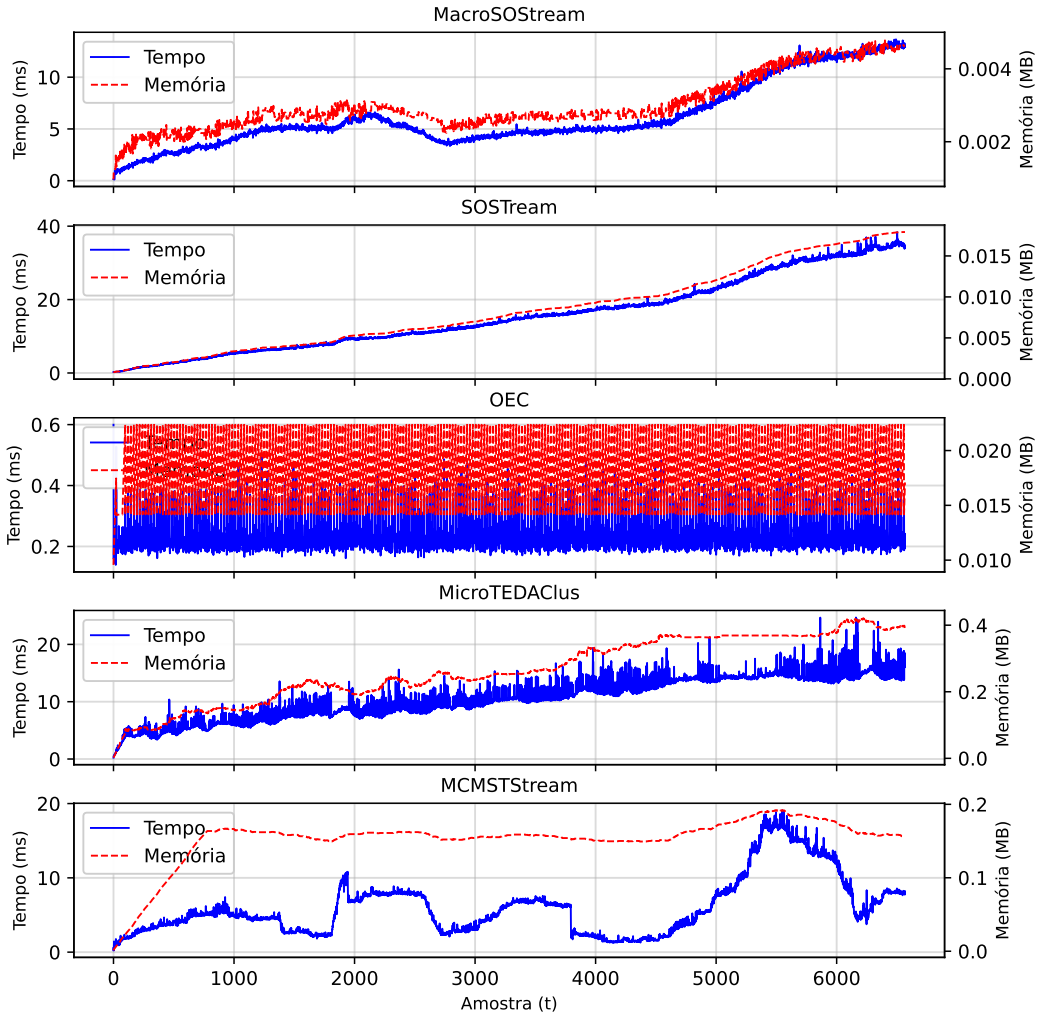


Figura 33 – Tempo de processamento (em milissegundos) e memória (em megabytes) pelos métodos MacroSOStream, SOSTream, OEC, MicroTEDAClus e MCMSTStream no CMFF.

Tabela 13 – Tempo médio de processamento e consumo de memória dos métodos na base CMFF. Os valores estão ordenados de forma ascendente em função da coluna de média de tempo.

	Média de Tempo (ms)	**Média de Memória (MB)**
CEDAS	0.2156 ± 0.0274	0.0078 ± 0.0036
OEC	0.2275 ± 0.0444	0.0183 ± 0.0025
eRBM-eGAE	0.7192 ± 6.8308	0.1508 ± 0.0612
AutoCloud	2.5262 ± 1.3476	0.1131 ± 0.0587
MCMSTStream	6.0145 ± 3.8999	0.1528 ± 0.0318
MacroSOStream	6.1353 ± 3.1193	0.0030 ± 0.0008
MicroTEDAClus	10.7127 ± 3.7390	0.2671 ± 0.0970

Contínua na próxima página

	Média de Tempo (ms)	**Média de Memória (MB)**
SOSTream	15.6356 ± 9.8746	0.0084 ± 0.0049
eSBM4Clus	33.6825 ± 330.8247	5.2510 ± 2.9911

5.4.3 Sensibilidade de hiperparametrização

Uma última avaliação foi realizada para analisar a sensibilidade de cada método em relação aos seus hiperparâmetros. Com base nos resultados da busca em grade, foram construídos diagramas de categorias paralelas, apresentados nos Apêndices A, B e C para os métodos 3W, TEP e CMFF, respectivamente.

Nos diagramas, os hiperparâmetros são representados por colunas dispostas horizontalmente e espaçadas uniformemente. Cada coluna é dividida em seções, onde cada seção corresponde a um valor atribuído ao respectivo hiperparâmetro. As combinações possíveis de valores entre os hiperparâmetros são representadas por linhas coloridas conectando as diferentes seções das colunas, sendo que a intensidade da cor reflete o valor do F1-score para aquela configuração específica.

Por exemplo, considere um método qualquer com os hiperparâmetros `A` e `B`, que assumem valores $[1, 25, 50, 75, 100]$ e $[0.1, 0.32, 0.55, 0.78, 1.0]$, respectivamente. Ao avaliar a acurácia como métrica de desempenho para cada combinação de valores, obtém-se o diagrama de categorias paralelas ilustrado na Figura 34. Observe na figura que, para esse cenário, o melhor valor para `A` pode ser identificado claramente (`A` $= 1$), enquanto a variação de `B` não apresenta impacto na performance final do método, sendo dependente apenas do valor de `A`.

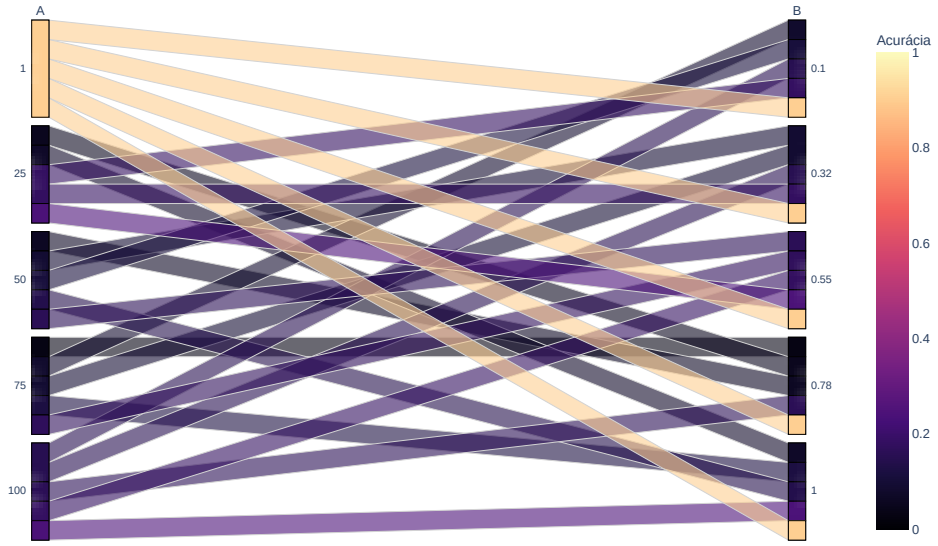


Figura 34 – Exemplo de um diagrama de categorias paralelas considerando um método com hiperparâmetros `A` e `B`, e a acurácia como métrica de performance.

Pelos resultados dos diagramas apresentados nos apêndices, observa-se que os

únicos métodos que apresentaram estabilidade em termos de parametrização foram os propostos nesta tese. Ou seja, o melhor desempenho foi consistentemente alcançado dentro dos mesmos intervalos ou mesma combinação de valores para os hiperparâmetros. Nos demais métodos, nota-se que, para cada base de dados, há apenas uma ou poucas combinações específicas que resultam em um bom desempenho, evidenciando uma maior sensibilidade à escolha dos hiperparâmetros.

5.5 Resumo do Capítulo

Neste capítulo, foram conduzidos experimentos com *benchmarks* da literatura para avaliar a performance das abordagens propostas em comparação com métodos de sucesso. De modo geral, os resultados sugerem que as abordagens propostas são promissoras, alcançando um desempenho semelhante ao dos métodos de referência. Além disso, apresentaram um consumo estável de recursos computacionais e consistência na parametrização ao longo de diferentes domínios de aplicação.

Capítulo 6

Conclusões

Esta tese apresentou três novas abordagens direcionadas a modelagem evolutiva de eventos em fluxos de dados. A primeira foi um autoencoder evolutivo granular, a segunda um *framework* geral para modelagem baseado em resíduos, e a terceira um método de clusterização evolutiva capaz de modelar cluster com qualquer formato. Embora os experimentos aqui tenham sido realizados com *benchmarks* da área de processos industriais, as abordagens propostas podem ser utilizadas em quaisquer ambientes que sejam caracterizados por um fluxo contínuo de dados, especialmente aqueles com alta dinâmica de funcionamento.

Pelos resultado obtidos, foi possível observar que as abordagens apresentaram grande potencial, e quando comparadas com aquelas utilizadas como referência, sempre ficaram entre as melhores, sendo que para a primeira base de dados, a eSBM4Clus, foi a única a apresentar 100% de acerto em todas as classes. E embora esse método tenha um ponto crítico, a necessidade da inversão da matriz de intra-similaridade, ainda apresentou bom desempenho com consumo razoável de recursos computacionais para problemas que variaram entre 3 a 7 na quantidade de classes, e 5 a 52 na quantidade de entradas. Podendo ser controlado por meio de seus hiperparâmetros.

O uso de grânulos de informação no aprendizado do AutoEncoder Granular Evolutivo, o proporcionou grande capacidade de aprendizado online e generalização. Embora melhorias sejam necessárias, em especial a manutenção do conjuntos de grânulos, o método demonstrou bom desempenho, alta capacidade de modelagem não linear e consumo estável de recursos computacionais.

Por fim, o *framework* proposto demonstrou também grande potencial para uso prático em ambientes não estacionários e com comportamentos não lineares. Embora, nesta tese tenha sido explorada a modelagem baseada em resíduos, a modelagem baseada no espaço latente pode facilitar essa tarefa, ao possibilitar o uso de métodos mais simples. Contudo, cabe estudos mais aprofundados para tratar questões relacionadas a mudança constante da representação latente.

6.1 Propostas de Continuidade

Como propostas de continuidade, são indicadas as seguintes principais questões envolvendo os métodos propostos eSBM4Clus e eGAE:

- Desenvolver novas formas que sejam computacionalmente eficientes para lidar com a tarefa de inversão da matriz G no eSBM4Clus;
- Desenvolver um procedimento de junção de clusters no eSBM4Clus, como procedimentos baseados em hiper-volume, ou distância de Hausdorff entre grânulos;
- Desenvolver novos mecanismos de aprendizado evolutivo para a eGAE, especialmente no que se diz respeito a mudar a estrutura da rede.
- Utilizar novas representações de grânulos de informação no eGAE para novos domínios, como imagem e texto;
- Usar estimadores da dimensão intrínseca de bases de dados para ser usada como alvo para a projeção no espaço latente;
- Propor procedimentos para lidar com a mudança constante do espaço latente, para então ser possível utilizá-lo na modelagem evolutiva.

Referências

C. C. Aggarwal, P. S. Yu, J. Han, and J. Wang. A framework for clustering evolving data streams. *Proceedings 2003 VLDB Conference: 29th International Conference on Very Large Databases (VLDB)*, pages 81–92, 1 2003. doi: https://doi.org/10.1016/B978-012722442-8/50016-1.

R. Ahmed, G. Dalkılıç, and Y. Erten. Dgstream: High quality and efficiency stream clustering algorithm. *Expert Systems with Applications*, 141:112947, 3 2020. ISSN 0957-4174. doi: https://doi.org/10.1016/J.ESWA.2019.112947.

R. Al-Amri, R. K. Murugesan, M. Man, A. F. Abdulateef, M. A. Al-Sharafi, and A. A. Alkahtani. A Review of Machine Learning and Deep Learning Techniques for Anomaly Detection in IoT Data. *Applied Sciences 2021, Vol. 11, Page 5320*, 11(12):5320, jun 2021. ISSN 2076-3417. doi: 10.3390/APP11125320.

N. M. Almeida. Proposta de framework para detecção de eventos em processos industriais e diagnóstico de causa raiz: Abordagens baseadas em aprendizado on-line. Master's thesis, Universidade Federal de Minas Gerais, Escola de Engenharia, Belo Horizonte, MG, 2020.

P. Angelov. Anomaly detection based on eccentricity analysis. In *2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*, pages 1–8, 2014. doi: 10.1109/EALS.2014.7009497.

P. Angelov, D. P. Filev, and N. Kasabov. *Evolving intelligent systems: methodology and applications*. John Wiley & Sons, 2010.

P. P. Angelov and D. P. Filev. An approach to online identification of takagi-sugeno fuzzy models. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 34:484–498, 2 2004. ISSN 10834419. doi: 10.1109/TSMCB.2003.817053.

R. Babuška and H. Verbruggen. Neuro-fuzzy methods for nonlinear system identification. *Annual reviews in control*, 27(1):73–85, 2003.

M. Baena-Garcıa, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno. Early drift detection method. In *Fourth international workshop on knowledge discovery from data streams*, volume 6, pages 77–86, 2006.

D. Bank, N. Koenigstein, and R. Giryes. Autoencoders. 2020. doi: 10.48550/ARXIV.2003.05991. URL `https://arxiv.org/abs/2003.05991`.

K. Batool and G. Abbas. A Comprehensive Review on Evolving Data Stream Clustering. *3rd International Conference on Communication Technologies, ComTech 2021*, pages 138–143, 2021. doi: 10.1109/COMTECH52583.2021.9616754.

C. G. Bezerra, B. S. J. Costa, L. A. Guedes, and P. P. Angelov. An evolving approach to data streams clustering based on typicality and eccentricity data analytics. *Information Sciences*, 518:13–28, 2020. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2019.12.022.

C. M. Bishop and N. M. Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

D. Bueso, M. Piles, and G. Camps-Valls. Nonlinear complex pca for spatio-temporal analysis of global soil moisture. In *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 5780–5783, 2018. doi: 10.1109/IGARSS.2018.8518155.

F. Cao, M. Ester, W. Qian, and A. Zhou. Density-based clustering over an evolving data stream with noise. *Proceedings of the Sixth SIAM International Conference on Data Mining*, 2006:328–339, 2006. doi: 10.1137/1.9781611972764.29.

Y. Cao. A benchmark case for statistical process monitoring - cranfield multiphase flow facility, 2025. Disponível em: <https://www.mathworks.com/matlabcentral/fileexchange/50938-a-benchmark-case-for-statistical-process-monitoring-cranfield-multiphase-flow-facility>. Acessado em: 10 de março de 2025.

H. Cardot, P. Cénac, and P.-A. Zitt. Efficient and fast estimation of the geometric median in hilbert spaces with an averaged stochastic gradient algorithm. *Bernoulli*, 19(1):18–43, 2013. doi: 10.3150/11-BEJ390.

X. Chen. Tennessee eastman simulation dataset, 2019. URL `https://dx.doi.org/10.21227/4519-z502`.

S. L. Chiu. Fuzzy model identification based on cluster estimation. *Journal of Intelligent and Fuzzy Systems*, 2(3):267–278, 1994. ISSN 18758967. doi: 10.3233/IFS-1994-2306.

L. A. Q. Cordovil, P. H. S. Coutinho, I. V. D. Bessa, M. F. S. V. D'Angelo, and R. M. Palhares. Uncertain data modeling based on evolving ellipsoidal fuzzy information

granules. *IEEE Transactions on Fuzzy Systems*, 28:2427–2436, 10 2020. ISSN 19410034. doi: 10.1109/TFUZZ.2019.2937052.

A. M. da Silva. *Sistemas Neuro-fuzzy Evolutivos: Novos Algoritmos de Aprendizado e Aplicações*. PhD thesis, (Doutorado em Engenharia Elétrica) - Escola de Engenharia, Universidade Federal de Minas Gerais, abril 2014.

A. R. Diallo, L. Homri, J.-Y. Dantan, F. Bonnet, and T. Boeuf. Faults diagnosis based on autoencoder and classifier techniques. In *2024 International Conference on Control, Automation and Diagnosis (ICCAD)*, pages 1–6, 2024. doi: 10.1109/ICCAD60883.2024.10553914.

G. Ditzler, M. Roveri, C. Alippi, and R. Polikar. Learning in nonstationary environments: A survey. *IEEE Computational Intelligence Magazine*, 10:12–25, 11 2015. ISSN 1556603X. doi: 10.1109/MCI.2015.2471196.

J. J. Downs and E. F. Vogel. A plant-wide industrial process control problem. *Computers & Chemical Engineering*, 17:245–255, 3 1993. ISSN 0098-1354. doi: 10.1016/0098-1354(93)80018-I.

L. M. Elshenawy and T. A. Mahmoud. Fault diagnosis of time-varying processes using modified reconstruction-based contributions. *Journal of Process Control*, 70:12–23, 10 2018. ISSN 09591524. doi: 10.1016/J.JPROCONT.2018.07.017.

B. Erdinç, M. Kaya, and A. Şenol. Mcmststream: applying minimum spanning tree to kd-tree-based micro-clusters to define arbitrary-shaped clusters in streaming data. *Neural Computing and Applications*, 36:7025–7042, 5 2024. ISSN 14333058. doi: https://doi.org/10.1007/S00521-024-09443-1.

D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660, 2010. URL `https://www.jmlr.org/papers/volume11/erhan10a/erhan10a.pdf`.

A. E. Ezugwu, A. M. Ikotun, O. O. Oyelade, L. Abualigah, J. O. Agushaka, C. I. Eke, and A. A. Akinyelu. A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Engineering Applications of Artificial Intelligence*, 110:104743, 2022. ISSN 0952-1976. doi: https://doi.org/10.1016/j.engappai.2022.104743.

C. Fahy, S. Yang, and M. Gongora. Ant colony stream clustering: A fast density clustering algorithm for dynamic data streams. *IEEE Transactions on Cybernetics*, 49:2215–2228, 6 2019. ISSN 21682267. doi: https://doi.org/10.1109/TCYB.2018.2822552.

J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. In *Advances in Artificial Intelligence–SBIA 2004: 17th Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, September 29-Ocotber 1, 2004. Proceedings 17*, pages 286–295. Springer, 2004.

J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys (CSUR)*, 46, 3 2014. ISSN 15577341. doi: 10.1145/2523813.

T. Gao, J. Yang, and S. Jiang. A novel fault detection model based on vector quantization sparse autoencoder for nonlinear complex systems. *IEEE Transactions on Industrial Informatics*, 19(3):2693–2704, 2023. doi: 10.1109/TII.2022.3174715.

N. L. A. Ghani, I. A. Aziz, and S. J. AbdulKadir. Subspace clustering in high-dimensional data streams: A systematic literature review. *Computers, Materials & Continua*, 75: 4649–4668, 3 2023a. ISSN 1546-2218. doi: 10.32604/CMC.2023.035987.

N. L. A. Ghani, I. A. Aziz, and S. J. AbdulKadir. Subspace clustering in high-dimensional data streams: A systematic literature review. *Computers, Materials & Continua*, 75: 4649–4668, 3 2023b. ISSN 1546-2218. doi: https://doi.org/10.32604/CMC.2023.035987.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

S. Gupta, A. Venugopal, and M. Jidhu Mohan. Fault detection and diagnosis using autoencoders and interpretable ai - case study on an industrial chiller. In *2022 IEEE International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, pages 198–203, 2022. doi: 10.1109/AdCONIP55568.2022.9894262.

M. Hahsler and M. Bolaos. Clustering data streams based on shared density between micro-clusters. *IEEE Transactions on Knowledge and Data Engineering*, 28:1449–1461, 6 2016. ISSN 10414347. doi: https://doi.org/10.1109/TKDE.2016.2522412.

Y. He, C. An, J. Lu, Y.-J. Wu, Z. Lu, and J. Xia. Bayesian deep learning approach for real-time lane-based arrival curve reconstruction at intersection using license plate recognition data. *IEEE Transactions on Intelligent Transportation Systems*, 26(1): 661–672, 2025. doi: 10.1109/TITS.2024.3491433.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. doi: 10.1126/science.1127647.

Z. Hu, H. Zhao, and J. Peng. Low-rank reconstruction-based autoencoder for robust fault detection. *Control Engineering Practice*, 123:105156, 2022. ISSN 0967-0661. doi: https://doi.org/10.1016/j.conengprac.2022.105156.

J. Huang, X. Sun, S. X. Ding, X. Yang, and O. K. Ersoy. Variational discriminative stacked auto-encoder: Feature representation using a prelearned discriminator, and its application to industrial process monitoring. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–12, 2024. doi: 10.1109/TNNLS.2024.3435519.

R. Hyde, P. Angelov, and A. R. Mackenzie. Fully online clustering of evolving data streams into arbitrarily shaped clusters. *Information Sciences*, 382-383:96–114, 2016. doi: https://doi.org/10.1016/j.ins.2016.12.004. URL `http://dx.doi.org/10.1016/j.ins.2016.12.0040020-0255/`.

M. J. Inacio. *Diagnóstico de Falhas Baseado em Sistema Inteligente Evolutivo*. PhD thesis, (Doutorado em Engenharia Elétrica) - Escola de Engenharia, Universidade Federal de Minas Gerais, dezembro 2014.

C. Isaksson, M. H. Dunham, and M. Hahsler. Sostream: Self organizing density-based clustering over data stream. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7376 LNAI: 264–278, 2012. ISSN 1611-3349. doi: https://doi.org/10.1007/978-3-642-31537-4_21.

K. Jang, K. E. S. Pilario, N. Lee, I. Moon, and J. Na. Explainable artificial intelligence for fault diagnosis of industrial processes. *IEEE Transactions on Industrial Informatics*, 21(1):4–11, 2025. doi: 10.1109/TII.2023.3240601.

L. Kaupp, B. Humm, K. Nazemi, and S. Simons. Autoencoder-ensemble-based unsupervised selection of production-relevant variables for context-aware fault diagnosis. *Sensors*, 22 (21), 2022. ISSN 1424-8220. doi: 10.3390/s22218259.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, 12 2013. URL `https://arxiv.org/abs/1312.6114v10`.

T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1 1982. ISSN 03401200. doi: https://doi.org/10.1007/BF00337288/METRICS.

V. Kreinovich. *From Interval (Set) and Probabilistic Granules to Set-and-Probabilistic Granules of Higher Order*, pages 1–16. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-19820-5. doi: 10.1007/978-3-642-19820-5_1.

S. Laohakiat, S. Phimoltares, and C. Lursinsap. Hyper-cylindrical micro-clustering for streaming data with unscheduled data removals. *Knowledge-Based Systems*, 99:183–200, 5 2016. ISSN 0950-7051. doi: https://doi.org/10.1016/J.KNOSYS.2016.02.004.

D. Leite, R. Ballini, P. Costa, and F. Gomide. Evolving fuzzy granular modeling from nonstationary fuzzy data streams. *Evolving Systems*, 3:65–79, 6 2012a. ISSN 18686478. doi: 10.1007/s12530-012-9050-9.

D. Leite, R. Ballini, P. Costa, and F. Gomide. Evolving fuzzy granular modeling from nonstationary fuzzy data streams. *Evolving Systems*, 3:65–79, 6 2012b. ISSN 18686478. doi: 10.1007/s12530-012-9050-9.

D. Leite, I. Škrjanc, S. Blažič, A. Zdešar, and F. Gomide. Interval incremental learning of interval data streams and application to vehicle tracking. *Information Sciences*, 630: 1–22, 6 2023. ISSN 0020-0255. doi: 10.1016/J.INS.2023.02.027.

D. Leite, G. Casalino, K. Kaczmarek-Majer, and G. Castellano. Incremental learning and granular computing from evolving data streams: An application to speech-based bipolar disorder diagnosis. *Fuzzy Sets and Systems*, 500:109205, 2025. ISSN 0165-0114. doi: https://doi.org/10.1016/j.fss.2024.109205.

A. Lemos, W. Caminhas, and F. Gomide. Fuzzy multivariable gaussian evolving approach for fault detection and diagnosis. In E. Hüllermeier, R. Kruse, and F. Hoffmann, editors, *Computational Intelligence for Knowledge-Based Systems Design*, pages 360–369, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

A. P. Lemos. *Modelagem Nebulosa Evolutiva: Novas Topologias e Algoritmos de Aprendizagem*. PhD thesis, (Doutorado em Engenharia Elétrica) - Escola de Engenharia, Universidade Federal de Minas Gerais, fevereiro 2011.

C. Li, C. Wen, and Z. Zhou. An industrial process fault diagnosis method based on independent slow feature analysis and stacked sparse autoencoder network. *Journal of the Franklin Institute*, 361(1):234–247, 2024a. ISSN 0016-0032. doi: https://doi.org/10.1016/j.jfranklin.2023.10.004.

G. Li, J. Chen, Y. Tao, A. Li, and X. Zhang. Residual life prediction of pneumatic control valves based on trans-tcn-gru modeling. *IEEE Access*, 12:191670–191681, 2024b. doi: 10.1109/ACCESS.2024.3513484.

G. Liu, H. Bao, and B. Han. A stacked autoencoder-based deep neural network for achieving gearbox fault diagnosis. *Mathematical Problems in Engineering*, 2018, 2018.

J. Liu. Fault diagnosis using contribution plots without smearing effect on non-faulty variables. *Journal of Process Control*, 22:1609–1623, 10 2012. ISSN 0959-1524. doi: 10.1016/J.JPROCONT.2012.06.016.

Y. Liu, K. Huang, B. J. Ma, K. Wei, Y. Li, C. Yang, and W. Gui. Quality-related fault detection for dynamic process based on quality-driven long short-term memory

network and autoencoder. *Neural Networks*, 181:106819, 2025. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2024.106819.

J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31:2346–2363, 12 2019. ISSN 15582191. doi: 10.1109/TKDE.2018.2876857.

E. Lughofer and M. Sayed-Mouchaweh. Autonomous data stream clustering implementing split-and-merge concepts – towards a plug-and-play approach. *Information Sciences*, 304:54–79, 5 2015. ISSN 0020-0255. doi: 10.1016/J.INS.2015.01.010.

S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30: 4765–4774, 2017. URL `https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf`.

J. MacQueen. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pages 281–297, 1967.

J. Maia, C. A. Severiano Junior, F. Gadelha Guimarães, C. Leite De Castro, P. Lemos, J. Camilo, F. Galindo, and W. Cohen. Evolving clustering algorithm based on mixture of typicalities for stream data mining. *Future Generation Computer Systems*, 106:672–684, 2020. doi: 10.1016/j.future.2020.01.017.

D. Mariano. Desenvolvimento e aplicação de uma técnica de detecção de falhas baseada em predição por similaridade de modelos em processos e equipamentos, 2019. Monografia (Graduação em Engenharia de Controle e Automação), Universidade Federal de Minas Gerais (UFMG), Belo Horizonte-MG, Brazil.

M. A. Marins, F. M. Ribeiro, S. L. Netto, and E. A. da Silva. Improved similarity-based modeling for the classification of rotating-machine failures. *Journal of the Franklin Institute*, 355(4):1913–1930, 2018.

M. Markou and S. Singh. Novelty detection: a review—part 1: statistical approaches. *Signal Processing*, 83:2481–2497, 12 2003. ISSN 0165-1684. doi: 10.1016/J.SIGPRO.2003.07.018.

A. K. Mishra and T. Motaung. Application of linear and nonlinear pca to sar atr. In *2015 25th International Conference Radioelektronika (RADIOELEKTRONIKA)*, pages 349–354, 2015. doi: 10.1109/RADIOELEK.2015.7129065.

G. Monkman and S. Wegerich. Similarity-based modeling of vibration features for fault detection and identification. *Sensor Review*, 2005.

M. Moshtaghi, C. Leckie, and J. C. Bezdek. Online clustering of multivariate time-series. *16th SIAM International Conference on Data Mining 2016, SDM 2016*, pages 360–368, 2016a. doi: 10.1137/1.9781611974348.41.

M. Moshtaghi, C. Leckie, and J. C. Bezdek. Online clustering of multivariate time-series. In *Proceedings of the 2016 SIAM international conference on data mining*, pages 360–368. SIAM, 2016b.

K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN 0262018020.

A. Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19, 2011. URL `https://web.stanford.edu/class/cs294a/sparseAutoencoder.pdf`.

V. D. M. Nhat and S. Lee. Two-dimensional weighted pca algorithm for face recognition. In *2005 International Symposium on Computational Intelligence in Robotics and Automation*, pages 219–223, 2005. doi: 10.1109/CIRA.2005.1554280.

A. S. Oliveira, R. S. de Abreu, and L. A. Guedes. Macro sostream: An evolving algorithm to self organizing density-based clustering with micro and macroclusters. *Applied Sciences 2022, Vol. 12, Page 7161*, 12:7161, 7 2022. ISSN 2076-3417. doi: https://doi.org/10.3390/APP12147161.

W. Pedrycz. Granular computing - the emerging paradigm. 2007.

W. Pedrycz. Information granules and their use in schemes of knowledge management. *Scientia Iranica*, 18:602–610, 6 2011. ISSN 1026-3098. doi: 10.1016/J.SCIENT.2011.04.013.

W. Pedrycz and W. Homenda. Building the fundamentals of granular computing: A principle of justifiable granularity. *Applied Soft Computing*, 13:4209–4218, 10 2013. ISSN 1568-4946. doi: 10.1016/J.ASOC.2013.06.017.

A. Perez, F. Jaramillo, V. Quintero, and M. Orchard. Characterizing the degradation process of lithium-ion batteries using a similarity-based-modeling approach. In *PHM Society European Conference*, volume 4, 2018.

A. Perez, H. Rozas, F. Jaramillo, V. Quintero, and M. Orchard. A simulation engine for the characterization of capacity degradation processes in lithium-ion batteries undergoing heterogeneous operating conditions. In *Proceedings of the Annual Conference of the PHM Society*, volume 11, 2019.

M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko. A review of novelty detection. *Signal processing*, 99:215–249, 2014.

K. S. S. Reddy and C. S. Bindu. Streamsw: A density-based approach for clustering data streams over sliding windows. *Measurement*, 144:14–19, 10 2019. ISSN 0263-2241. doi: https://doi.org/10.1016/J.MEASUREMENT.2018.11.041.

A. Rehman, A. Khan, M. A. Ali, M. U. Khan, S. U. Khan, and L. Ali. Performance analysis of pca, sparse pca, kernel pca and incremental pca algorithms for heart failure prediction. In *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, pages 1–5, 2020. doi: 10.1109/ICECCE49384.2020.9179199.

M. J. Rezaee, M. Eshkevari, M. Saberi, and O. Hussain. Gbk-means clustering algorithm: An improvement to the k-means algorithm based on the bargaining game. *Knowledge-Based Systems*, 213:106672, 2021.

B. Ribeiro. Fault detection in a thermoplastic injection molding process using neural networks. In *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No.99CH36339)*, volume 5, pages 3352–3355 vol.5, 1999. doi: 10.1109/IJCNN.1999.836199.

M. T. Ribeiro, S. Singh, and C. Guestrin. "why should i trust you?": Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016. doi: 10.1145/2939672.2939778.

N. B. Roa, L. Travé-Massuyès, and V. H. Grisales-Palacio. Dyclee: Dynamic clustering for tracking evolving environments. *Pattern Recognition*, 94:162–186, 10 2019. ISSN 0031-3203. doi: https://doi.org/10.1016/J.PATCOG.2019.05.024.

S. Rolewicz. *Functional Analysis and Control Theory: Linear Systems*, volume 29 of *Mathematics and its Applications, East European Series*. Springer Netherlands, Dordrecht, 1987. ISBN 9027721866. doi: 10.1007/978-94-015-7758-8.

C. Ruiz-Cárcel, Y. Cao, D. Mba, L. Lao, and R. Samuel. Statistical process monitoring of a multiphase flow facility. *Control Engineering Practice*, 42:74–88, 2015. ISSN 0967-0661. doi: https://doi.org/10.1016/j.conengprac.2015.04.012.

J. Senthilnath, B. Zhou, Z. W. Ng, D. Aggarwal, R. Dutta, J. W. Yoon, A. P. P. Aung, K. Wu, M. Wu, and X. Li. Self-evolving autoencoder embedded q-network, 2024. URL `https://arxiv.org/abs/2402.11604`.

R. Singer, K. Gross, J. Herzog, R. King, and S. Wegerich. Model-based nuclear power plant monitoring and fault detection: Theoretical foundations. In *Proceedings of the International Conference on Intelligent Systems Applications to Power Systems*, 6 1997.

F. A. Tobar, L. Yacher, R. Paredes, and M. E. Orchard. Anomaly detection in power generation plants using similarity-based modeling and multivariate analysis. In *Proceedings of the 2011 American Control Conference*, pages 1940–1945. IEEE, 2011.

R. E. V. Vargas, C. J. Munaro, P. M. Ciarelli, A. G. Medeiros, B. G. do Amaral, D. C. Barrionuevo, J. C. D. de Araújo, J. L. Ribeiro, and L. P. Magalhães. A realistic and public dataset with rare undesirable real events in oil wells. *Journal of Petroleum Science and Engineering*, 181:106223, 10 2019. ISSN 0920-4105. doi: 10.1016/J.PETROL.2019.106223.

P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 1096–1103, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390294.

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

A. Wald. *Sequential analysis*. Courier Corporation, Mineola, New York, 2004.

K. Wang, L. Zhou, and Y. Wang. Enhanced fault detection using deviation degree penalty with stacked autoencoder in industry process. In *2020 IEEE 9th Data Driven Control and Learning Systems Conference (DDCLS)*, pages 1084–1089, 2020. doi: 10.1109/DDCLS49620.2020.9275084.

Z. Wang, C. Wang, and Y. Li. Variational autoencoder based on knowledge sharing and correlation weighting for process-quality concurrent fault detection. *Engineering Applications of Artificial Intelligence*, 133:108051, 2024. ISSN 0952-1976. doi: https://doi.org/10.1016/j.engappai.2024.108051.

C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, 1992. doi: 10.1007/BF00992698.

N. Wattanakitrungroj, S. Maneeroj, and C. Lursinsap. Versatile hyper-elliptic clustering approach for streaming data based on one-pass-thrown-away learning. *Journal of Classification*, 34:108–147, 4 2017. ISSN 14321343. doi: https://doi.org/10.1007/S00357-017-9222-1/METRICS.

N. Wattanakitrungroj, S. Maneeroj, and C. Lursinsap. Bestream: Batch capturing with elliptic function for one-pass data stream clustering. *Data & Knowledge Engineering*, 117:53–70, 9 2018. ISSN 0169-023X. doi: https://doi.org/10.1016/J.DATAK.2018.07.002.

S. Wegerich, A. Wilks, and R. Pipke. Nonparametric modeling of vibration signal features for equipment health monitoring. In *Proceedings of the IEEE Aerospace Conference*, volume 7, pages 3113–3121. IEEE, 2003.

S. W. Wegerich. Similarity based modeling of time synchronous averaged vibration signals for machinery health monitoring. In *2004 IEEE Aerospace Conference Proceedings (IEEE Cat. No. 04TH8720)*, volume 6, pages 3654–3662. IEEE, 2004.

L. Weng. From autoencoder to beta-vae, 2018. Disponível em: <`https://lilianweng.github.io/posts/2018-08-12-vae/`>. Acessado em: 08 de março de 2025.

Y. Wiseman. Autonomous vehicles. In *Research Anthology on Cross-Disciplinary Designs and Applications of Automation*, pages 878–889. IGI Global, 2022.

R. R. Yager. Measures of specificity over continuous spaces under similarity relations. *Fuzzy Sets and Systems*, 159(17):2193–2210, 2008. ISSN 0165-0114. doi: https://doi.org/10.1016/j.fss.2007.12.026. Theme: Fuzzy Relations.

Y. Yao. A triarchic theory of granular computing. *Granular Computing*, 1(2):145–157, 06 2016. ISSN 2364-4974. doi: 10.1007/s41066-015-0011-0.

Y. Yao, J. Ma, S. Feng, and Y. Ye. Svd-ae: An asymmetric autoencoder with svd regularization for multivariate time series anomaly detection. *Neural Networks*, 170: 535–547, 2024. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2023.11.023.

B. Yue, K. Wang, H. Zhu, X. Yuan, and C. Yang. Spiking autoencoder for nonlinear industrial process fault detection. *Information Sciences*, 665:120389, 2024. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2024.120389.

L. A. Zadeh. Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. *Fuzzy Sets and Systems*, 90(2):111–127, 1997. ISSN 0165-0114. doi: https://doi.org/10.1016/S0165-0114(97)00077-8. Fuzzy Sets: Where Do We Stand? Where Do We Go?

L. Zeng, X. Zhang, K. Yu, Q. Jin, Y. Wu, L. Chen, and X. Wu. Incipient fault detection and process monitoring of thermal power plant pulverizing system based on deep representation learning. *Transactions of the Institute of Measurement and Control*, 0(0): 01423312231182464, 2023. doi: 10.1177/01423312231182464.

Y. Zheng and P. Zheng. Image matching based on fast pca-sift descriptors with automatic determination of dimensionality for pca. In *2022 IEEE 2nd International Conference on Information Communication and Software Engineering (ICICSE)*, pages 115–120, 2022. doi: 10.1109/ICICSE55337.2022.9828915.

A. Zubaroğlu and V. Atalay. Data stream clustering: a review. *Artificial Intelligence Review*, 54:1201–1236, 2020. doi: 10.1007/s10462-020-09874-x.

I. Škrjanc, J. A. Iglesias, A. Sanchis, D. Leite, E. Lughofer, and F. Gomide. Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification:

A survey. *Information Sciences*, 490:344–368, 2019. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2019.03.060.

Apêndice A

Resultado da busca em grade para o 3W

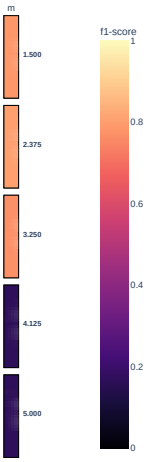


Figura 35 – Resultado da busca em grade no 3W: AutoCloud.

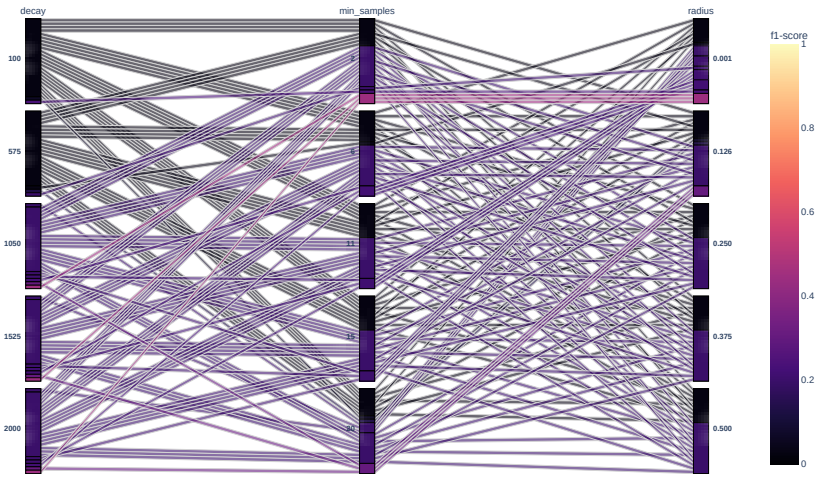


Figura 36 – Resultado da busca em grade no 3W: CEDAS.

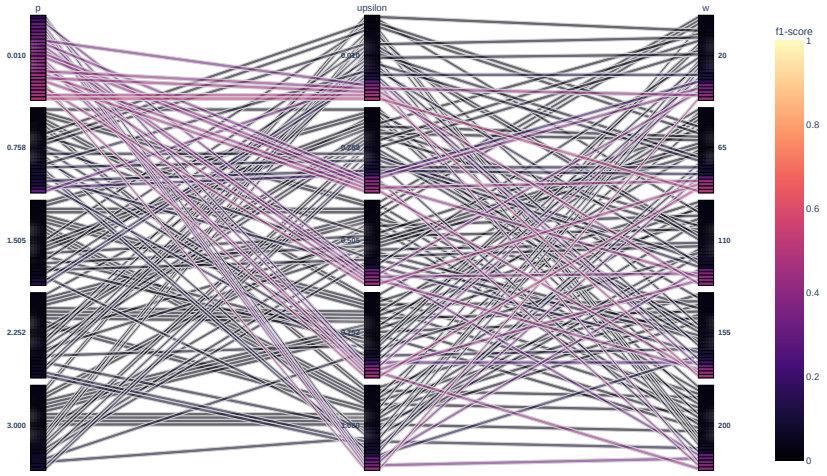


Figura 37 – Resultado da busca em grade no 3W: eRBM-eGAE.

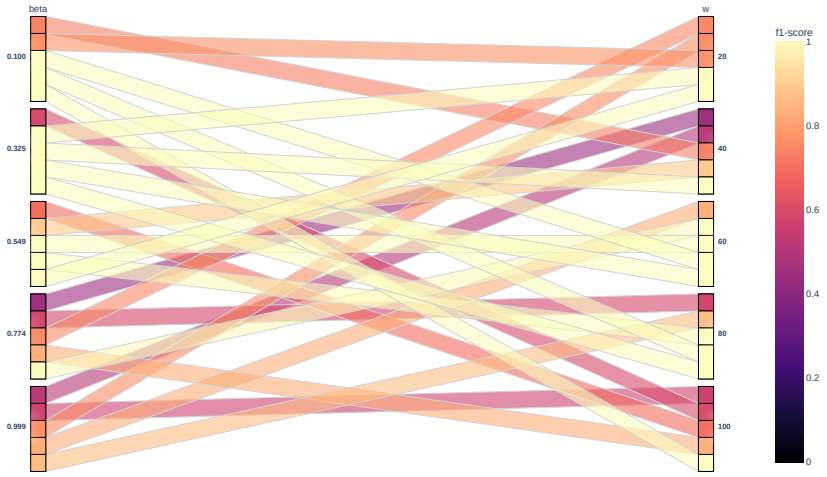


Figura 38 – Resultado da busca em grade no 3W: eSBM4Clus.

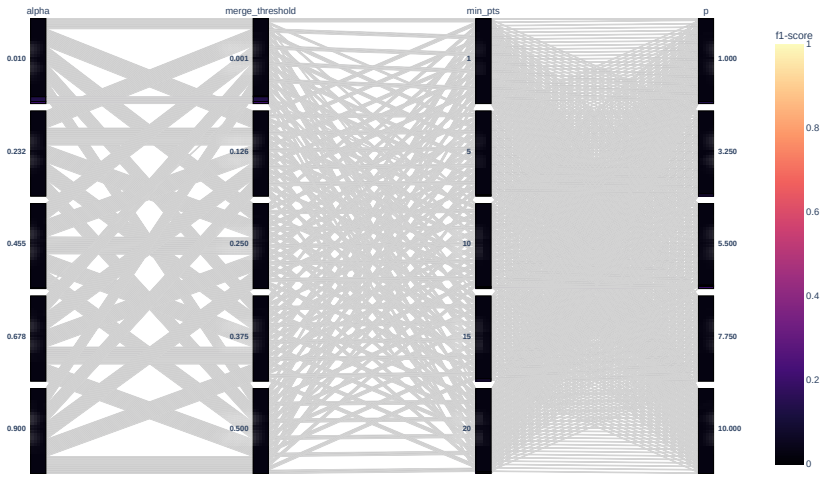


Figura 39 – Resultado da busca em grade no 3W: MacroSOStream.

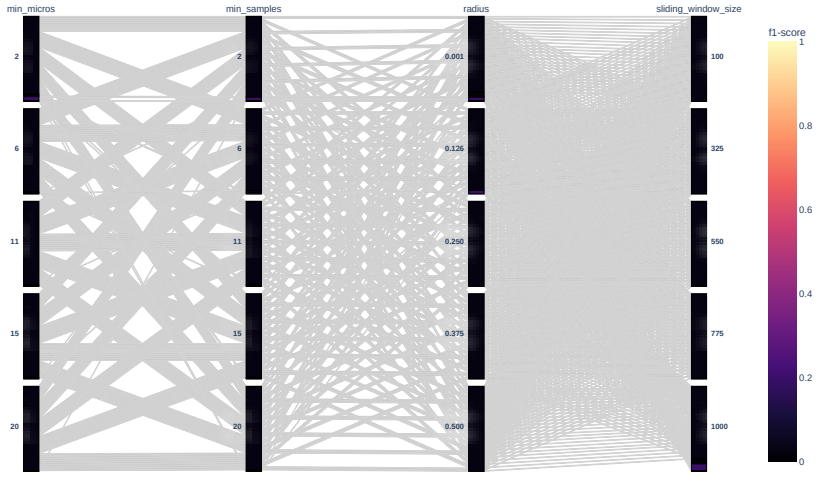


Figura 40 – Resultado da busca em grade no 3W: MCMSTStream.

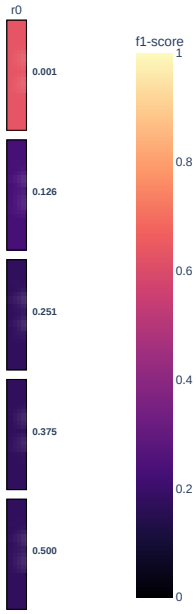


Figura 41 – Resultado da busca em grade no 3W: MicroTEDAClus.

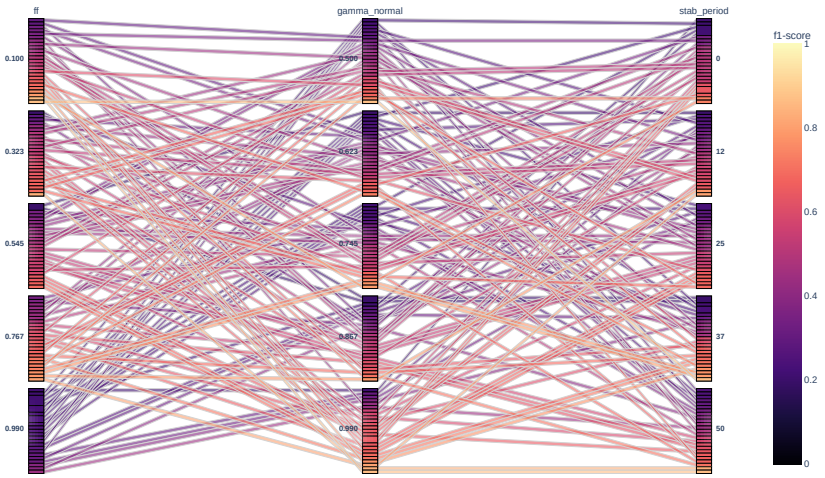


Figura 42 – Resultado da busca em grade no 3W: OEC.

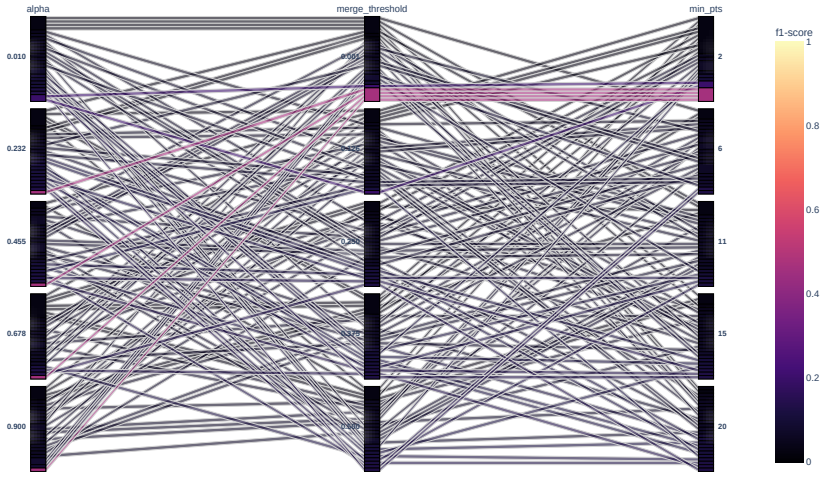


Figura 43 – Resultado da busca em grade no 3W: SOSTream.

Apêndice B

Resultado da busca em grade para o TEP

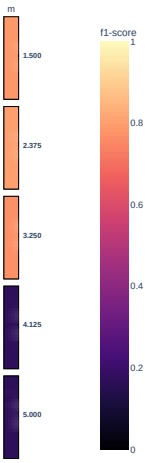


Figura 44 – Resultado da busca em grade no TEP: AutoCloud.

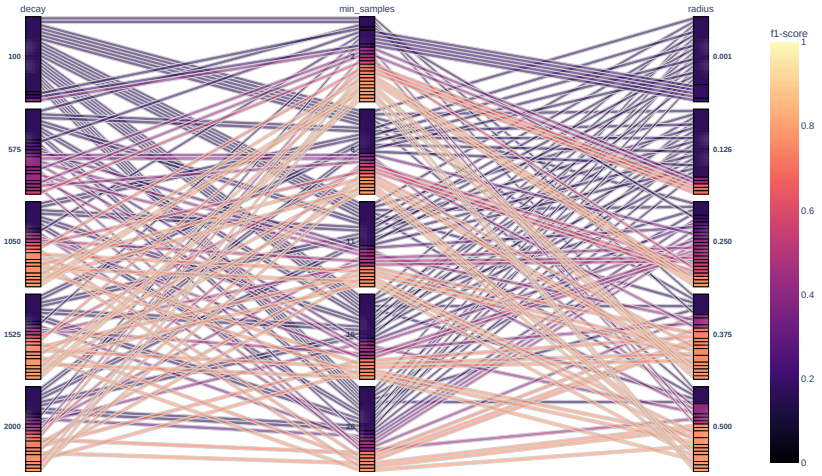


Figura 45 – Resultado da busca em grade no TEP: CEDAS.

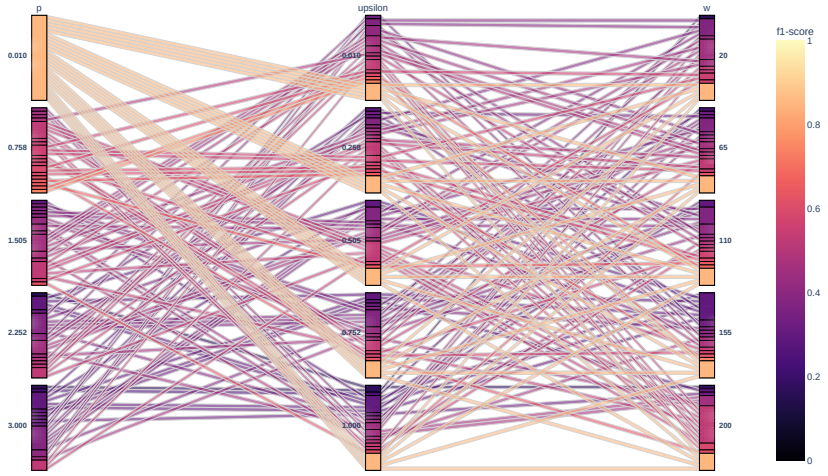


Figura 46 – Resultado da busca em grade no TEP: eRBM-eGAE.

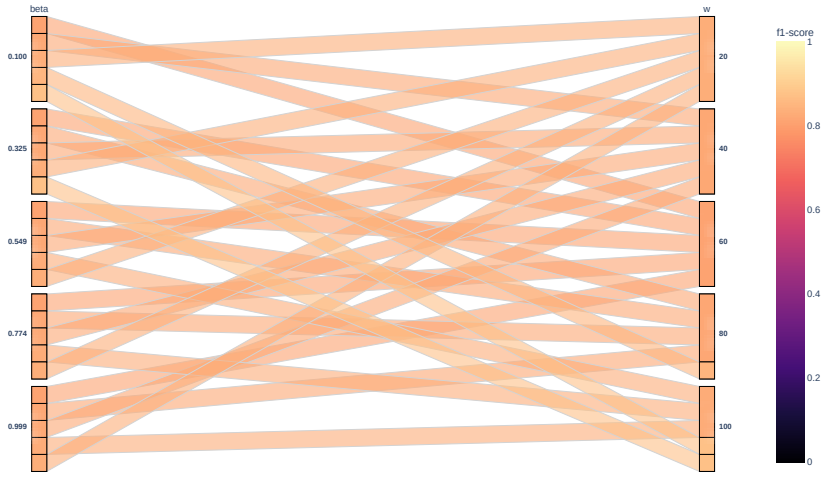


Figura 47 – Resultado da busca em grade no TEP: eSBM4Clus.

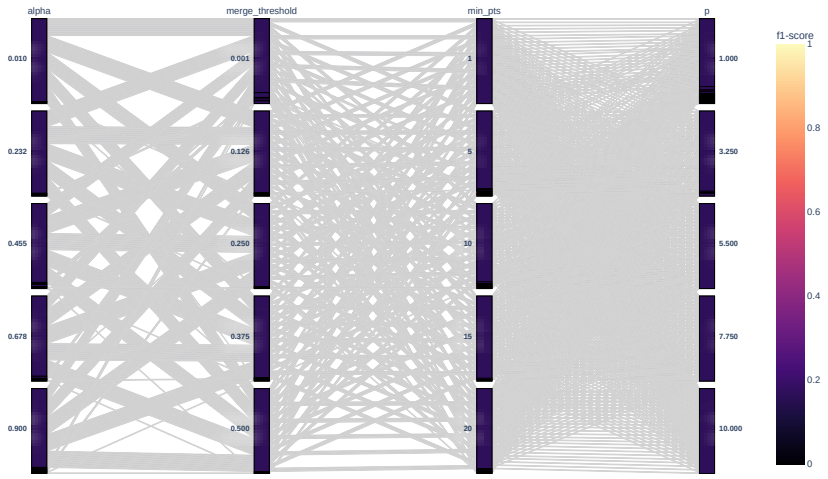


Figura 48 – Resultado da busca em grade no TEP: MacroSOStream.

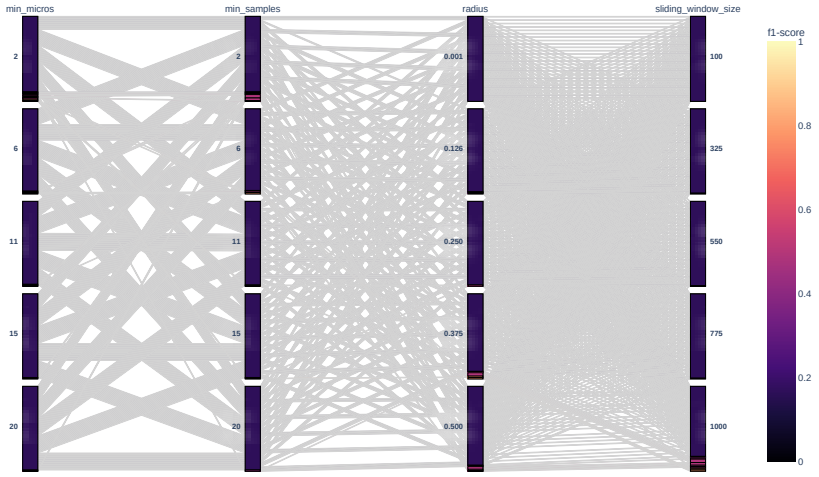


Figura 49 – Resultado da busca em grade no TEP: MCMSTStream.

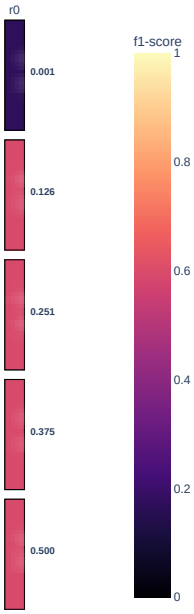


Figura 50 – Resultado da busca em grade no TEP: MicroTEDAClus.

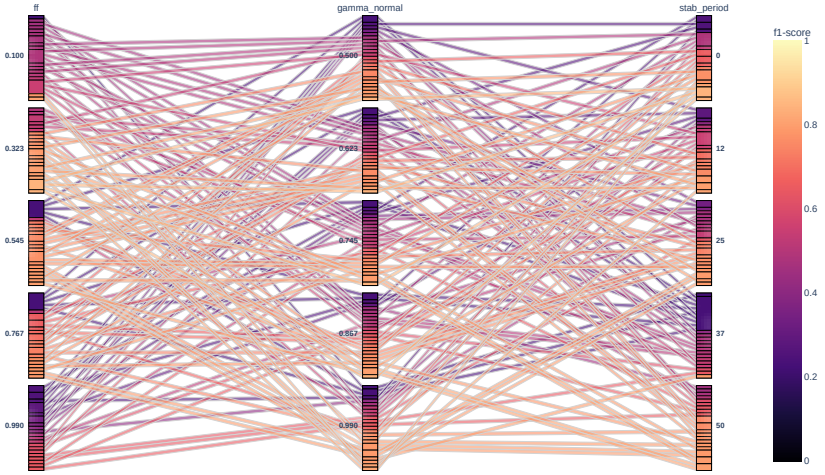


Figura 51 – Resultado da busca em grade no TEP: OEC.

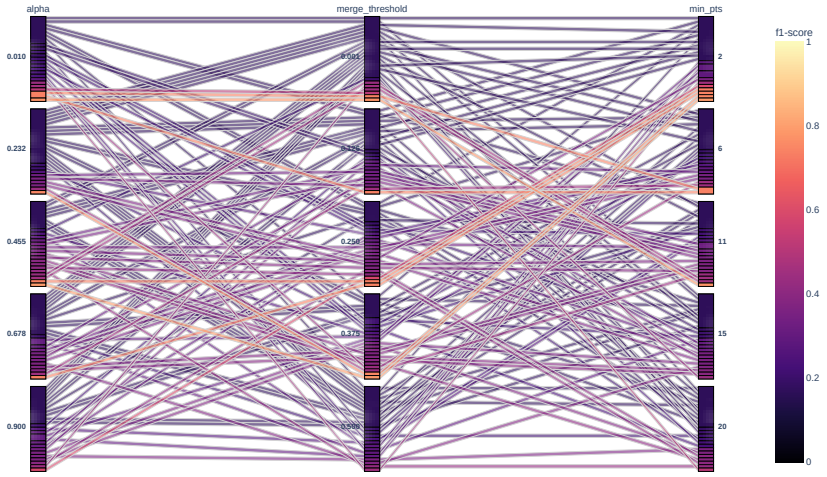


Figura 52 – Resultado da busca em grade no TEP: SOSTream.

Apêndice C

Resultado da busca em grade para o CMFF

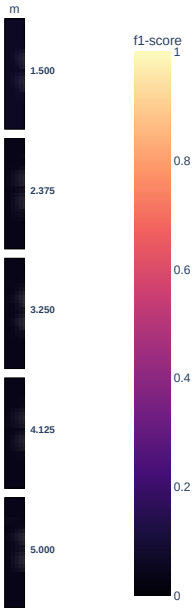


Figura 53 – Resultado da busca em grade no CMFF: AutoCloud.

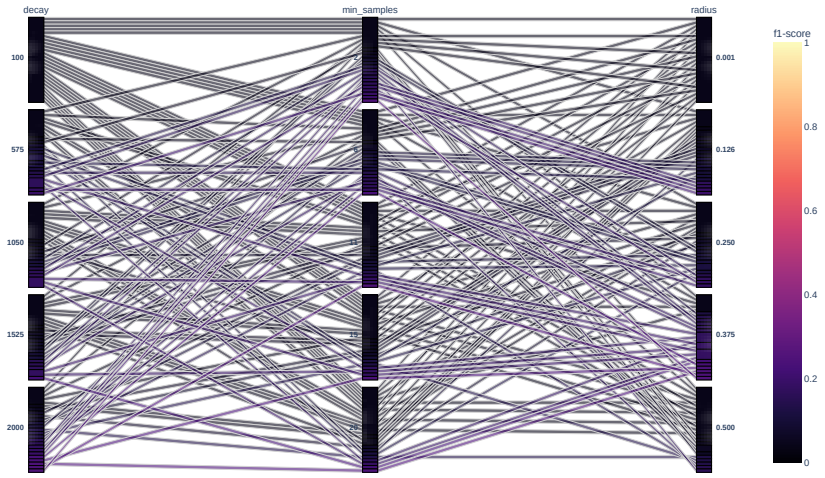


Figura 54 – Resultado da busca em grade no CMFF: CEDAS.

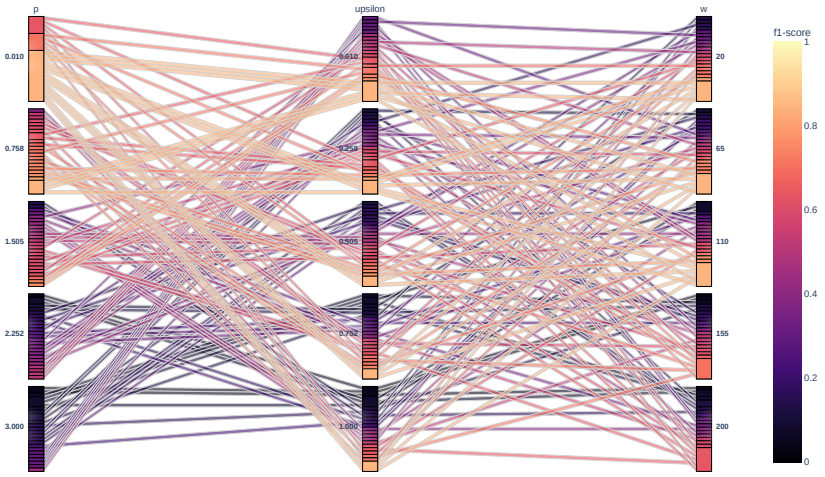


Figura 55 – Resultado da busca em grade no CMFF: eRBM-eGAE.

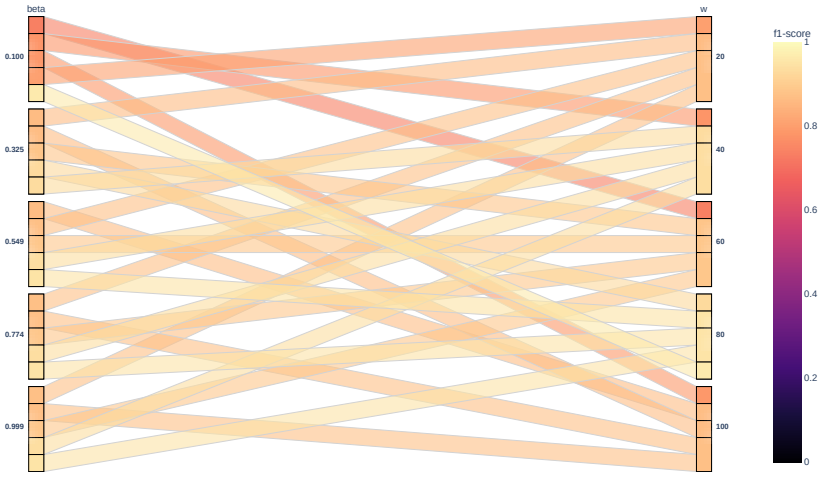


Figura 56 – Resultado da busca em grade no CMFF: eSBM4Clus.

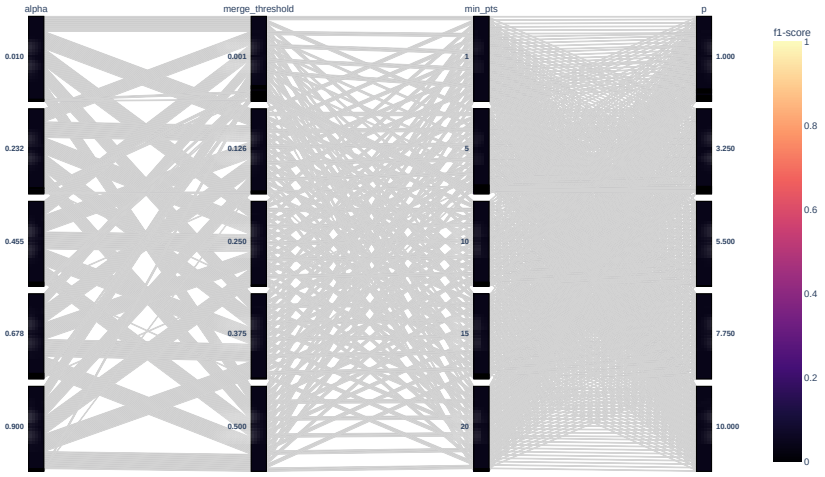


Figura 57 – Resultado da busca em grade no CMFF: MacroSOStream.

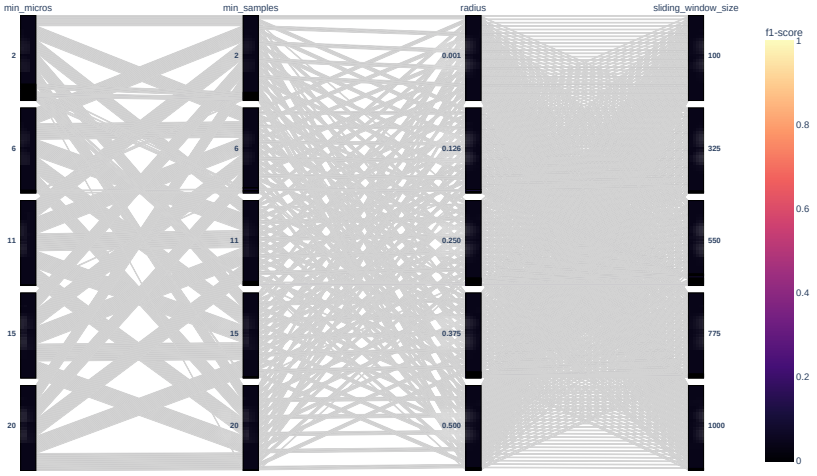


Figura 58 – Resultado da busca em grade no CMFF: MCMSTStream.

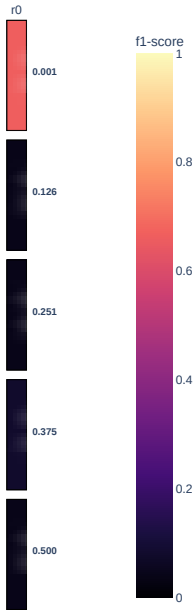


Figura 59 – Resultado da busca em grade no CMFF: MicroTEDAClus.

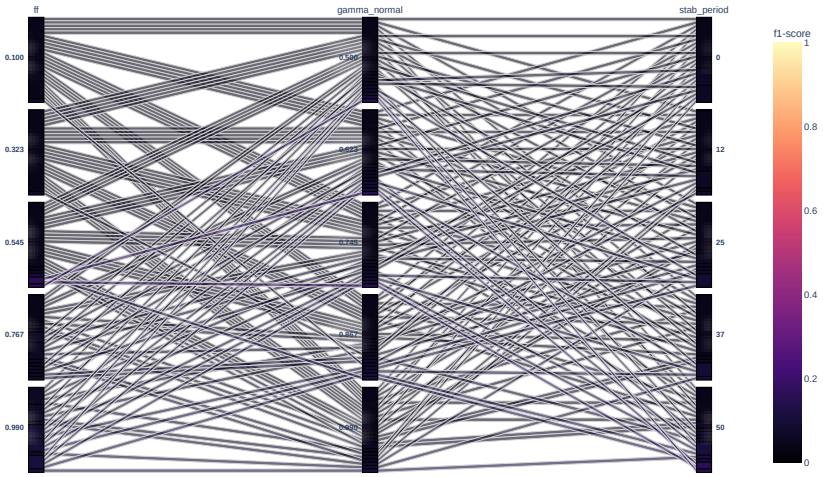


Figura 60 – Resultado da busca em grade no CMFF: OEC.

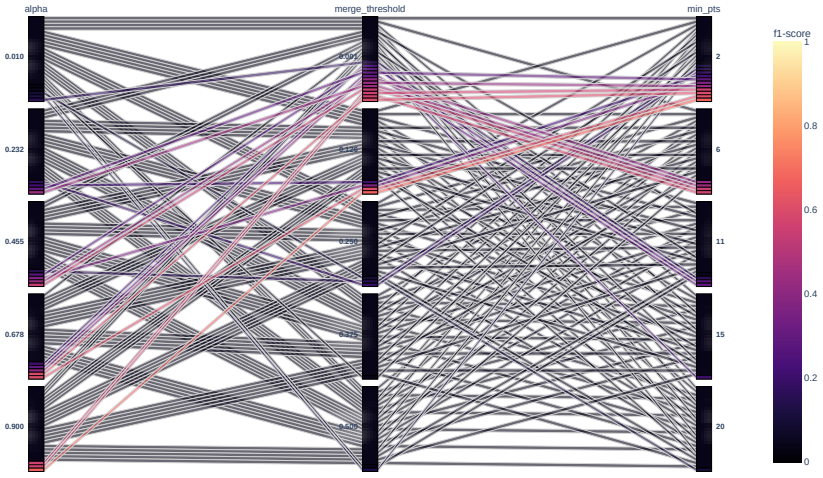


Figura 61 – Resultado da busca em grade no CMFF: SOSTream.