Universidade Federal de São Carlos– UFSCar
Centro de Ciências Exatas e de Tecnologia– CCET
Departamento de Computação– DC
Programa de Pós-Graduação em Ciência da Computação– PPGCC

Herbert Gonçalves Dias

Mineração de fluxo contínuo de dados para previsão da ação recomendada para o tráfego de rede em firewall

São Carlos
2025

Herbert Gonçalves Dias

Mineração de fluxo contínuo de dados para previsão da ação recomendada para o tráfego de rede em firewall

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências Exatas e de Tecnologia da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Metodologias e Técnicas de Computação

Orientador: Ricardo Cerri

São Carlos
2025

Agradecimentos

Primeiramente, agradeço a Deus pela saúde, pela força e pela sabedoria que me acompanhou em cada decisão ao longo desses anos.

Agradeço ao meu orientador, Prof. Ricardo Cerri, pela excelente orientação ao longo de todo o meu Mestrado. A confiança no meu trabalho, a paciência nos momentos difíceis, as correções detalhadas, o incentivo constante e a disponibilidade, inclusive fora dos horários habituais, foram essenciais para o êxito deste estudo. Também sou grato pelos valiosos ensinamentos e conselhos, que contribuíram diretamente para a realização deste trabalho.

Agradeço aos professores e funcionários do DC/UFSCar - São Carlos, que, de alguma forma, contribuíram para a realização deste trabalho, seja com apoio direto ou indireto, deixo meu sincero agradecimento.

Por fim, agradeço à minha família e aos meus amigos, que me ofereceram apoio incondicional e incentivo ao longo de toda esta jornada.

Resumo

O avanço da Internet resulta em um aumento na geração contínua de dados, elevando os riscos de danos causados por ameaças cibernéticas. Os *firewalls* desempenham um papel essencial na segurança das redes, fornecendo proteção contra ameaças. Eles operam organizando os registros de *log* com base em regras específicas, levando em consideração critérios como o propósito das conexões e as políticas estabelecidas pela organização. A atualização dessas regras é desafiadora, e escolhas inadequadas podem resultar em vulnerabilidades de segurança. Os trabalhos relacionados à previsão da ação recomendada para o tráfego de rede ainda utilizam métodos tradicionais de Aprendizado de Máquina (AM), onde o conjunto de dados é estático e pode ser percorrido repetidamente, e a detecção de novos padrões requer um novo ciclo de treinamento, teste e implementação de novos modelos. No entanto, o tráfego de rede é continuamente capturado por um *firewall*, e seu comportamento pode mudar tanto de forma abrupta quanto gradualmente. Isso é comum em ambientes de rede, onde é necessário adaptar-se a mudanças frequentes nas políticas de segurança, alterações na rede ou novas ameaças. Este trabalho propõe a aplicação dos algoritmos MINAS e Clustream Adaptado para a previsão da ação recomendada em tráfego de rede. Os *logs* são sessões de tráfego de rede capturados por um *firewall* de uma instituição de ensino. A adaptação é necessária para que esses algoritmos sejam capazes de classificar as ações recomendadas *(Allow, Deny, Drop e Reset-Both)* em um cenário de Fluxo Contínuo de Dados (FCDs). O CluStream é um algoritmo incremental, enquanto o MINAS, além de ser incremental, é especializado em Detecção de Novidade (DN). Ambos os algoritmos foram selecionados por suas especificidades e habilidades em aprender e evoluir o modelo à medida que novos dados chegam do FCDs. Experimentos realizados demonstram que os dois algoritmos apresentaram resultados distintos no cenário de FCDs, com desempenhos satisfatórios em diferentes métricas ao longo dos experimentos.

Palavras-chave: Fluxos contínuos de dados, segurança de rede, logs de firewall.

Abstract

The advancement of the Internet has resulted in an increase in the continuous generation of data, raising the risks of damage caused by cyber threats. Firewalls play a crucial role in network security by providing protection against these threats. They operate by organizing log records based on specific rules, considering criteria such as the purpose of connections and the policies established by the organization. Updating these rules is challenging, and inadequate choices can lead to security vulnerabilities. Related work on predicting recommended actions for network traffic still employs traditional machine learning methods, where the dataset is static and can be repeatedly traversed, while detecting new patterns requires a new cycle of training, testing, and implementing new models. However, network traffic is continuously captured by a firewall, and its behavior can change both abruptly and gradually. This is common in network environments, where it is necessary to adapt to frequent changes in security policies, network modifications, or new threats. This work proposes the application of the MINAS and Adapted CluStream algorithms to predict recommended actions in network traffic. The logs consist of sessions of network traffic captured by a firewall from an educational institution. Adaptation is necessary for these algorithms to classify the recommended actions (Allow, Deny, Drop, and ResetBoth) in a data stream scenario. CluStream is an incremental algorithm, while MINAS, besides being incremental, specializes in novelty detection. Both algorithms were selected for their specificities and capabilities in learning and evolving the model as new data arrives from the data stream. Experiments conducted demonstrate that both algorithms showed distinct results in the data stream, with satisfactory performance across different metrics throughout the experiments.

Keywords: Data streams, network security, firewall logs.

Lista de ilustrações

Lista de tabelas

Lista de siglas

AM Aprendizado de Máquina

AP Aprendizado Profundo

BIRCH Balanced Iterative Reducing and Clustering using Hierarchies

CFV Cluster Feature Vector

DN Detecção de Novidade

Dtree Decision Tree Clasifier

FCDs Fluxo Contínuo de Dados

IoT Internet das Coisas

IA Inteligência Artificial

IDS Sistema de Detecção de Intrusão

IPS Sistema de Prevenção de Intrusão

KNN K-nearest neighbor

LogReg Logistic Regression

MINAS MultI-class learNing Algorithm for data Streams

MD Mineração de Dados

SGDC Stochastic Gradient Descent

SVM Support Vector Machine

Sumário

Capítulo 1

Introdução

1.1 Contextualização

A Internet está constantemente em avanço, e com a evolução da Internet das Coisas (IoT), que reúne uma ampla variedade de dispositivos, incluindo dispositivos móveis, eletrônicos de consumo, automóveis e vários tipos de sensores, a geração de fluxos de dados que produzem quantidades massivas de informações de forma contínua tem crescido de forma exponencial (AL-AMRI et al., 2021; ALJABRI et al., 2022). Esse substancial crescimento eleva o risco de danos causados por ameaças cibernéticas. Quando ocorre uma invasão em um dispositivo da rede, este pode ser utilizado para atacar outros dispositivos e sistemas, roubar informações, causar danos físicos imediatos ou executar várias outras ações maliciosas em empresas de todos os setores. Ameaças cibernéticas e ameaças internas colocam as organizações em posição de risco. Isso implica a necessidade de aplicar mecanismos para proteger a integridade e usabilidade dos dados (CASSALES et al., 2019; NEUPANE; HADDAD; CHEN, 2018).

Na maioria dos ataques cibernéticos, o atacante possui conhecimento dos mecanismos de defesa empregados pela organização, permitindo-lhe ocultar seu ataque (ALJABRI et al., 2022). Os *firewalls* desempenham um papel crucial na segurança da rede organizacional, representando a primeira barreira de defesa. Sua função não apenas consiste em proteger contra ameaças vindas de fontes externas, mas também internas. Para identificar tais ataques, é necessário analisar continuamente todos os registros de tráfego, criando um perfil que auxilie na determinação de regras para que o *firewall* tome as medidas apropriadas em resposta aos pacotes recebidos. Entretanto, essas regras estão em constante evolução para se adaptar às variações nos ataques, ao avanço das ferramentas e

à sofisticação dos métodos dos atacantes (SCHINDLER, 2018). Essa constante mutação das regras é uma questão significativa, uma vez que as regras são manualmente definidas por engenheiros e pessoal de segurança das organizações, com base em suas políticas e requisitos.

1.2 Motivação

Os gestores de sistemas configuram *firewalls* conforme as demandas específicas de suas organizações (ERTAM; KAYA, 2018). Dada a sua importância fundamental na proteção das redes contra ameaças, sejam internas ou externas, os *firewalls* se revelam como uma parte vital das atuais infraestruturas de comunicação. Basicamente, esses dispositivos organizam os registros de *log* da rede com base em regras que são estabelecidas manualmente, levando em consideração critérios como a finalidade da conexão, portas autorizadas para comunicação, sub-redes permitidas, entre outros, de acordo com as políticas vigentes na organização que faz uso do *firewall*. Adicionalmente, em virtude dos avanços tecnológicos e das frequentes mudanças no cenário ambiental, a atualização constante dessas regras se torna um processo desafiador e contínuo (UCAR; OZHAN, 2017). A partir dessas regras e de vários outros atributos presentes nos registros de *log* do *firewall*, são executadas ações, tais como “Permitir”, “Descartar”, “Negar” ou “Redefinir ambos”. A escolha inadequada de uma ação para lidar com uma sessão pode resultar em vulnerabilidades de segurança, possibilitando eventos indesejados, como desativação de dispositivos, perda de serviços e, indiretamente, prejuízos financeiros ou divulgação de informações confidenciais.

Portanto, os *firewalls* constituem componentes fundamentais para a segurança de qualquer infraestrutura de rede. No entanto, a administração e estabelecimento de regras que determinam as ações apropriadas têm-se tornado uma tarefa complexa e suscetível a erros (UCAR; OZHAN, 2017). Algoritmos de Inteligência Artificial (IA), como os de Aprendizado de Máquina (AM), revelam um potencial vasto em diversas áreas, inclusive na segurança cibernética (TU, 2019). A área de AM dedica-se a projetar algoritmos que executam tarefas de maneira mais eficiente à medida que adquirem mais experiência e são capazes de aprender e melhorar automaticamente sem serem explicitamente programados (MITCHELL, 1997). No AM tradicional, um conjunto de treinamento está disponível previamente, e uma vez treinado, o modelo de decisão não muda com o passar do tempo; esses modelos são estáticos, ou seja, uma vez que o modelo é gerado, ele não é atualizado com novos dados (GAMA, 2010).

Os sistemas de segurança de rede, a exemplo dos Sistema de Detecção de Intrusão (IDS) e *firewalls*, também geram grandes volumes de registros de *log*. Dentro desses registros, um conhecimento valioso está oculto. Mediante o uso de AM, esse conhecimento, aliado aos padrões nos atributos de tráfego da rede, pode ser desvendado e explo-

rado para desenvolver modelos que suportam a detecção de ameaças à rede (WINDING; WRIGHT; CHAPPLE, 2006).

Até o momento, todos os trabalhos relacionados fazem uso de técnicas de Mineração de Dados e AM para auxiliar e automatizar o processo de previsão da ação recomendada para as sessões de tráfego de rede, utilizando os métodos tradicionais de AM, aprendizado em *batch*, onde o conjunto de dados é estático e pode ser percorrido repetidamente. Nesse contexto, a detecção de novos padrões de ataque requer um novo ciclo de treinamento, teste e implementação de novos modelos.

No entanto, para problemas que possuem uma natureza dinâmica, a utilização de algoritmos de aprendizado em *batch* não é apropriada. A vasta quantidade de dispositivos e a subsequente geração de dados em volumes e velocidades elevadas impedem a aplicação de métodos tradicionais de AM.

Firewalls geram grandes volumes de arquivos de *log* que fluem continuamente ao longo do tempo. Gerenciar essas imensas quantidades de dados, frequentemente em alta velocidade, representando sequências infinitas de dados gerados de maneira contínua, é uma tarefa complexa. Além disso, realizar atividades como encontrar a média ou a entropia torna-se desafiador, dado que o fluxo não é finito, há falta de conhecimento prévio sobre todos os possíveis exemplos e há possibilidade de mudanças nas classes do problema ao longo do tempo. Dados com essas características são denominados Fluxo Contínuo de Dados (FCDs) (GAMA, 2010). Conhecidos em inglês como *data streams*, FCDs consistem em uma sequência ordenada de exemplos potencialmente infinita $(x_1, x_2, x_3, \ldots, x_i, \ldots)$ lidos em ordem crescente dos índices i. Eles são gerados de forma não estacionária e, na maioria das vezes, em alta velocidade (GUHA et al., 2000).

Ao contrário dos métodos tradicionais, para esses FCDs, técnicas de mineração de fluxos de dados, entre outras, são promissoras e podem ser aplicadas para auxiliar na escolha de uma ação mais adequada para o tráfego de rede do *firewall*, tomando decisões como “Permitir”, “Descartar”, “Negar” ou “Redefinir ambos”, de acordo com os atributos fornecidos pelos *logs* de cada sessão do tráfego de rede, lidando com diversas vantagens que os algoritmos de FCDs podem proporcionar. Dentre essas vantagens, podem ser destacadas (CASSALES et al., 2019; PINTO, 2005):

- ❏ Trabalhar com memória limitada;
- ❏ Processamento de dados de tráfego com uma única leitura;
- ❏ Produção de resposta em tempo real;
- ❏ Detectar novidades e mudanças em conceitos já aprendidos.

Dentre as técnicas de mineração de fluxo de dados, os algoritmos de Detecção de Novidade (DN) em FCDs permitem identificar mudanças de conceito e padrões emergentes.

Neste trabalho, propõe-se a utilização de dois algoritmos de mineração de FCDs não supervisionados para auxiliar na previsão da ação recomendada, utilizando *logs* de sessão de tráfego de rede de um *firewall*. Esses algoritmos foram escolhidos por serem bem estabelecidos na mineração de fluxo de dados e por representarem abordagens distintas.

Para alcançar esse objetivo, será utilizado o algoritmo de DN conhecido como MultI-class learNing Algorithm for data Streams (MINAS) (FARIA; CARVALHO; GAMA, 2016), que funciona em duas fases: offline e online. Além disso, adaptações foram realizadas para empregar o algoritmo CluStream, um framework desenvolvido para agrupamento em FCDs, baseado no conceito de microgrupos (AGGARWAL et al., 2003). A versão modificada do CluStream original foi direcionada especificamente à classificação de *logs* de sessões de tráfego de rede. Esse algoritmo opera em três fases principais: (i) uma fase offline, responsável pela inicialização dos microgrupos rotulados; (ii) uma fase online, dedicada ao processamento contínuo dos dados; e (iii) uma fase de macroagrupamento, que consolida os microgrupos em macrogrupos (Grupos Maiores), permitindo rotular as instâncias em uma das diferentes classes do problema.

Um *firewall* representa um componente essencial na segurança de redes, desempenhando a função de monitorar de maneira contínua o tráfego de entrada e saída, tomando decisões sobre a permissão ou bloqueio de fluxos específicos. Dessa operação contínua, surgem registros de tráfego de rede, sendo cada entrada no *log* uma sessão contendo atributos, tais como endereço IP de origem e destino, a aplicação utilizada, a porta designada, bytes enviados e recebidos, a regra que originou a ação e qual medida o *firewall* adota em resposta a essa regra, seja Permitir, Negar, Descartar ou Resetar ambos.

Os *logs* de sessões de tráfego de rede em *firewalls* geram FCDs, que dificultam o uso de algoritmos supervisionados ou não supervisionados tradicionais. Esses algoritmos, por dependerem de dados estáticos, são inadequados para lidar com a natureza dinâmica dos FCDs. Em contraste, os algoritmos MINAS e CluStream Adaptado se destacam por sua capacidade de ajustar continuamente seus modelos à medida que novos dados chegam. O MINAS ainda possui a capacidade de DN, o que proporciona uma abordagem diferente para a análise dos mesmos dados, que são altamente desbalanceados, e permite uma comparação direta com o CluStream Adaptado. Além disso, esses algoritmos são baseados em microgrupos e são utilizados para classificar fluxos de dados em evolução em cenários nos quais os rótulos verdadeiros são considerados sujeitos a um atraso infinito, ou seja, nunca são fornecidos. Dessa forma, o modelo não depende de rótulos verdadeiros para realizar atualizações durante a fase online. Essas características tornam o MINAS e o CluStream Adaptado mais adequados para trabalhar com dados contínuos para prever a ação recomendada para o tráfego de rede, diferenciando-os dos métodos convencionais.

Outro motivo para a escolha do MINAS e do CluStream Adaptado é que ambos apresentam baixa dependência de rótulos corretos, o que significa que o desempenho não é significativamente impactado por erros nas regras do *firewall*. Isso ocorre porque sua

base de funcionamento é centrada na formação de microgrupos, que são determinados por padrões intrínsecos dos dados e não diretamente pelos rótulos. Dessa forma, erros específicos nas regras não comprometem a identificação desses padrões. Além disso, com o desempenho satisfatório dos modelos, é possível realizar uma análise comparativa entre as divergências observadas nas ações recomendadas pelos algoritmos e aquelas tomadas pelas regras configuradas. Esse processo auxilia na identificação e correção de possíveis erros de configuração, garantindo maior eficácia na definição das regras de *firewall*.

No estudo de Komadina et al. (2024), que foca na análise de *logs* de *firewall* de uma grande rede de controle industrial, é apresentado um método inovador para gerar anomalias que simulam ações reais de atacantes. O objetivo é possibilitar a detecção dessas anomalias, deliberadamente inseridas nos *logs* do *firewall*. Diversos algoritmos de aprendizado em *batch*, tanto supervisionado quanto não supervisionado, foram testados. Os resultados indicam que nenhum dos algoritmos de aprendizado não supervisionado investigados alcançou desempenho satisfatório na detecção de anomalias. Isso confirma que as anomalias injetadas representam etapas de ataque bem integradas e disfarçadas nos *logs* de *firewall* existentes. Em contrapartida, métodos de aprendizado supervisionado demonstraram resultados significativamente melhores e mais satisfatórios, destacando o potencial desses métodos para identificar e integrar anomalias.

Ainda assim, todos os trabalhos encontrados até o momento na literatura foram baseados em algoritmos de AM tradicional para classificar a ação recomendada em quatro rótulos distintos (Permitir, Negar, Descartar ou Redefinir-Ambos) e não adaptados para FCDs. Outros trabalhos abordam o problema no contexto de FCDs, porém são voltados especificamente para detecção de intrusões, tratando-o como um problema clássico de classificação binária, prevendo a ação como normal ou ameaça, como em um Sistema de Detecção de Intrusão (IDS), ou separando entre a classe normal e diferentes classes de ameaça, ao invés de um *firewall* baseado em regras que, além de bloquear intrusões, bloqueia tráfego que não são ameaças, mas que, por políticas internas de segurança, aplicam o bloqueio do tráfego de rede. Portanto, nenhuma abordagem com um algoritmo de mineração de FCDs foi aplicada a esse contexto específico. Assim, o MINAS e CluStream Adaptado se posicionam como duas abordagens diferentes das já aplicadas para a previsão da ação recomendada em sessões de tráfego de rede, possuindo a capacidade de operar de forma a aprender com novos dados à medida que chegam, ajustando seu modelo continuamente com cada nova amostra de dados, o que pode ser essencial em aplicações onde a rapidez na adaptação aos dados mais recentes é fundamental e há necessidades de identificar padrões não convencionais em FCDs. Além disso, como o fluxo de dados ocorre a partir do tráfego de rede capturado por um *firewall*, seu comportamento pode mudar tanto de forma abrupta quanto gradualmente. Isso é comum em ambientes de rede, onde é necessário adaptar-se a mudanças frequentes nas políticas de segurança, alterações na rede ou novas ameaças. Nesse contexto, é importante investigar se um algoritmo com

capacidade de DN pode ajudar a identificar padrões de variação abrupta nos dados ou em cenários com classes altamente desbalanceadas.

Nesse cenário, o modelo proposto deve prever a ação recomendada a ser tomada em relação a cada sessão à medida que o tráfego flui pela rede. Essas ações podem ser classificadas como (*Allow/Permitir, Drop/Descartar, Deny/Negar e Rest-Both/Reiniciar-Ambos*), determinando se cada sessão de tráfego de rede será permitida, descartada ou bloqueada. Essas ações são tomadas com base em regras fixas, previamente configuradas no *firewall* pela equipe de segurança da informação da instituição de ensino. Tais regras só são alteradas em casos de ajustes nas políticas de segurança, mudanças no cenário ambiental ou quando se identificam necessidades de correção, não possuindo caráter dinâmico. Nessa situação, os algoritmos e modelos deste estudo são treinados a partir dos *logs* de tráfego de rede gerados pelo *firewall*. Esses *logs* registram as ações tomadas com base em regras estáticas, que determinam se um determinado tráfego deve ser permitido (*Allow*) ou bloqueado (*Drop, Deny e Reset-Both*). A partir dessa base de dados, os modelos aprendem a identificar padrões de comportamento conforme as decisões tomadas pelo *firewall*.

1.3 Hipótese e Objetivos

Este trabalho tem como objetivo principal a aplicação dos algoritmos MINAS e Clustream Adaptado para a previsão da ação recomendada em tráfego de rede. Os *logs* são sessões de tráfego de rede capturadas por um *firewall* de uma instituição de ensino. A adaptação é necessária para que esses algoritmos sejam capazes de classificar as ações recomendadas *(Allow, Deny, Drop e Reset-Both)* em um cenário de FCDs.

Dado o objetivo principal desta pesquisa, a hipótese é apresentada da seguinte forma:

- Métodos de mineração de FCDs que utilizam técnicas de agrupamento incremental não supervisionado para atualizar modelos de decisão à medida que os dados chegam do fluxo, serão capazes de classificar ações recomendadas para o tráfego de rede, mantendo a precisão e a eficiência mesmo em cenários onde os dados geralmente chegam em alta velocidade, são sequências infinitas gerados continuamente e podem ter alta variação no padrão de dados. A performance (precisão e eficiência) será avaliada por meio de métricas como acurácia, F1-Score, entre outras.

Formulada a hipótese, o objetivo principal foi concretizado por meio das seguintes adaptações e implementações específicas:

- **Adaptação do CluStream:** Desenvolver um método de agrupamento incremental que permita a criação de microgrupos rotulados para cada classe do problema durante a fase offline, e o ajuste contínuo desses microgrupos na fase online, de modo a refletir mudanças no tráfego de rede sem a necessidade de reprocessamento

completo dos dados. Por fim, realizar uma fase de macroagrupamento, onde os microgrupos são agrupados em conjuntos maiores (macrogrupos) com o intuito de rotular os exemplos em uma das classes do atributo alvo: Action (Allow, Deny, Drop e Reset-Both);

- **Adaptação do MINAS:** Modificar os hiperparâmetros do algoritmo MINAS para garantir que ele reconheça e classifique os exemplos em uma das classes fixas existentes, evitando a criação de novas classes e ajustando os microgrupos para refletir as mudanças dos conceitos conhecidos. Dessa forma, novos padrões e conceitos devem ser identificados, mas as ações a serem tomadas continuam sendo restritas a permitir ou negar o tráfego;

- **Comparação dos Resultados:** Por fim, será realizada uma comparação com os trabalhos relacionados, algoritmos supervisionados voltados para FCDs e entre o desempenho da abordagem que utiliza DN para criar extensões de conceitos conhecidos, como no caso do MINAS, e o algoritmo CluStream Adaptado, que não emprega DN. Essa comparação visa avaliar a eficácia de cada abordagem na adaptação a padrões de tráfego e na manutenção da precisão e eficiência ao longo do tempo.

1.4 Organização do Documento

O restante do documento está organizado da seguinte forma:

- **Capítulo 2 - Segurança da Informação: Proteção de Redes e Sistemas:** Neste capitulo será apresentada uma visão geral dos principais sistemas de segurança. Serão abordadas as características dos sistemas de detecção e prevenção de intrusões, além dos principais tipos de *firewalls* existentes. Por fim, serão detalhadas as principais características do *firewall* Palo Alto 3260, que será utilizado para a aquisição dos *logs* utilizados nos experimentos.

- **Capítulo 3 - Fluxo Contínuo de Dados:** Será fornecida uma visão geral sobre FCDs. Também são destacados os conceitos de Mineração, Classificação e Agrupamento em FCDs.

- **Capítulo 4 - Detecção de Novidade em FCDs:** Serão apresentadas as características fundamentais de DN, juntamente com a Mudança de Conceito e Evolução de Conceito. Também será apresentada uma visão geral dos algoritmos desenvolvidos para DN, abrangendo suas fases online e offline. Por último, serão apresentados os Métodos de Avaliação em Detecção de Novidade.

- **Capítulo** 5 **- Trabalhos Relacionados:** Serão apresentados os principais trabalhos da literatura que estão relacionados aos estudos desta pesquisa.

- **Capítulo** 6 **- Classificação da Ação Recomendada para Tráfego de Rede em Fluxos Contínuos de Dados: MINAS e CluStream:** Serão apresentados os métodos, e o algoritmo MINAS, fornecendo uma visão abrangente de suas fases offline, online, bem como dos procedimentos de Detecção de Extensões, Detecção de Padrões Novidade e das abordagens para lidar com recorrências. Além disso, serão apresentados os métodos e funcionamento do algoritmo para agrupamento em FCDs, conhecido como *Clustream*. Além da versão original do algoritmo, será discutida a adaptação realizada para permitir sua aplicação na recomendação de ações para o tráfego de rede em *firewalls*.

- **Capítulo** 7 **- Experimentos:** Serão mostrados os detalhes sobre os experimentos realizados e seus respectivos resultados para a previsão da ação recomendada para o tráfego de rede em *firewalls*. Além disso, serão discutidas as comparações entre os experimentos deste estudo e os desafios que dificultam a comparação com outras pesquisas relacionadas.

- **Capítulo** 8 **- Conclusões:** Serão apresentadas as principais contribuições desta pesquisa, destacando as principais limitações dos métodos sugeridos e propondo direções para trabalhos futuros.

Capítulo 2

Segurança da Informação: Proteção de Redes e Sistemas

A informação tornou-se um dos ativos mais valiosos de uma organização, exigindo proteção adequada para assegurar a realização dos negócios e o alcance de metas. A preocupação com a segurança da informação tornou-se ainda mais essencial com o avanço da tecnologia na comunicação de dados, que evidenciou novas ameaças. As práticas de segurança da informação abrangem diretrizes, normas, procedimentos e políticas destinadas a preservar a confidencialidade, a integridade e a disponibilidade das informações. O objetivo é garantir que as informações não sejam acessadas por pessoas não autorizadas, que estejam armazenadas de forma adequada e disponíveis quando necessário (FACHINELLI; AHLERT, 2019; JUNIOR, 2018).

2.1 Sistema de Detecção de Intrusão e Sistema de Prevenção de Intrusão

Sistema de Detecção de Intrusão (IDS) e Sistema de Prevenção de Intrusão (IPS) são tecnologias de segurança de rede projetadas para detectar e prevenir intrusões maliciosas. No entanto, elas operam em locais diferentes e de maneiras distintas. Na Tabela 1, são destacadas as principais diferenças entre esses dois sistemas (SCARFONE; MELL, 2007).

O IDS identifica invasões maliciosas ou acessos não autorizados na rede, analisando cópias de pacotes de dados sem realizar uma varredura direta. Ao observar padrões predefinidos nos pacotes, o IDS pode alertar sobre possíveis ameaças (FACHINELLI; AHLERT, 2019).

Tabela 1 – Diferenças entre IDS e IPS

	IDS	**IPS**
Definição	Detecção de Intrusões	Prevenção de Intrusões
Função	Detectar atividades maliciosas	Bloquear atividades maliciosas
Ação	Monitoramento e alerta	Monitoramento, alerta e resposta
Localização	Fora da linha de tráfego	Na linha de tráfego
Impacto no Tráfego	Não interfere diretamente	Bloqueia ou modifica pacotes
Reação a Ameaças	Alerta administradores	Alerta e toma medidas
Complexidade	Menos complexo	Mais complexo

Fonte: Elaborado pelo autor.

Para Fernandes e Rothenberg (2014), Gonçalves et al. (2019), Splunk (2024), Scarfone e Mell (2007), existem dois tipos principais de IDS:

- **Baseado em Rede**: Monitora o tráfego de rede em tempo real para detectar atividades maliciosas. Normalmente é implementado em pontos estratégicos da rede, como switches ou roteadores. Geralmente ele é implementado fora da linha de tráfego de rede, o que significa que ele monitora o tráfego de rede passivamente. Ele é conectado a um ponto de espelhamento de rede, permitindo uma observação e analise os dados que passam sem interferir diretamente no fluxo de tráfego. Assim, um IDS pode detectar atividades suspeitas e alertar administradores sem impactar o desempenho ou a disponibilidade da rede;

- **Baseado em Host**: É instalado diretamente nos dispositivos ou servidores individuais (hosts) que ele monitora. Observa a atividade, analisando *logs* do sistema, arquivos de configuração e outras atividades internas. Um IDS pode detectar alterações suspeitas no sistema de arquivos, tentativas de acesso não autorizado, e outros comportamentos anômalos específicos ao host. Como o IDS baseado em host opera diretamente no host, ele pode fornecer uma visão detalhada das ações que ocorrem dentro daquele dispositivo específico, mas não monitora o tráfego de rede em geral.

Além disso, o IDS pode operar em dois modos:

- **Detecção de Assinaturas**: Compara o tráfego de rede contra uma base de dados de assinaturas conhecidas de ataques;

- **Detecção de Anomalias:** Analisa o tráfego de rede em busca de atividades que desviem do comportamento normal.

Um IDS apenas alerta os administradores sobre uma possível intrusão, mas não toma nenhuma ação para mitigar o ataque (HAT, 2023).

Assim como o IDS, o Sistema de Prevenção de Intrusão (IPS) pode ser implementado em dois formatos: baseado em rede ou baseado em host. No entanto, o IPS vai além do simples monitoramento; ele atua de forma proativa para bloquear tráfego malicioso e tomar medidas de intervenção na rede. Diferentemente de um *firewall*, que baseia suas decisões apenas em endereços IP e portas, o IPS analisa também o conteúdo das aplicações. Sua implementação ocorre diretamente na linha de tráfego de rede, ou seja, ele está inserido no caminho que os dados percorrem. Entre as ações preventivas mais comuns do IPS estão (HAT, 2023; SCARFONE; MELL, 2007):

- **Bloqueio de Pacotes Maliciosos:** O IPS pode bloquear pacotes de dados que sejam identificados como maliciosos, evitando que cheguem ao destino final;
- **Resetar Conexões:** Pode encerrar conexões de rede suspeitas para interromper atividades maliciosas;
- **Reconfiguração de Firewall:** Pode ajustar dinamicamente as regras do *firewall* para reforçar a segurança contra ataques detectados.

Uma solução integrada de IDS e IPS é denominada NIDS/NIPS (*Network Intrusion Detection System/Network Intrusion Prevention System*). Esse conjunto não apenas monitora a rede, mas também pode realizar intervenções como bloqueios diretamente na infraestrutura em que estão implantados. Eles monitoram o tráfego de rede em busca de atividades suspeitas ou maliciosas que possam indicar uma tentativa de intrusão ou ataque. Ao contrário dos sistemas baseados em host, que monitoram atividades em um único dispositivo, ou dos IPS baseados em rede, cuja a ênfase está na capacidade de resposta ativa, o NIDS/NIPS é projetado para proteger toda a rede. O NIDS/NIPS equilibra a detecção de ameaças com a capacidade de intervir para prevenir ataques, oferecendo uma proteção abrangente e eficaz (FERNANDES; ROTHENBERG, 2014; GONÇALVES et al., 2019).

2.2 Firewall

O *firewall* é posicionado na fronteira entre redes privadas ou entre uma rede privada e a Internet. Em uma empresa com várias redes de computadores interconectadas, todo o tráfego de entrada e saída para a Internet deve passar pelo *firewall*. Isso possibilita ao *firewall* permitir ou bloquear pacotes com base nas políticas de segurança configuradas. Seu objetivo principal é controlar o tráfego de rede com base em um conjunto de regras de segurança predefinidas, permitindo ou bloqueando o tráfego com base em critérios

como endereço IP, porta de origem/destino, protocolo, entre outros (TANENBAUM; FEAMSTER; WETHERALL, 2021).

Para Kurose e Ross (2006), um *firewall* é uma solução composta por hardware e software que separa a rede interna de uma empresa da Internet, permitindo o controle seletivo do tráfego. Alguns pacotes são autorizados a passar, enquanto outros são bloqueados, proporcionando ao administrador da rede um controle completo sobre o sistema gerenciado, utilizando regras pré-configuradas. Os principais objetivos de um *firewall* incluem ser resistente a invasões, gerenciar todo o tráfego de entrada e saída, e servir como o único ponto de entrada e saída dos dados da rede. O termo *firewall* é uma analogia com a barreira física à prova de fogo usada para impedir a propagação do fogo entre estruturas. Da mesma forma, um *firewall* de rede atua como uma barreira virtual que controla e monitora o tráfego de dados entre redes, protegendo contra acessos não autorizados e possíveis ameaças digitais.

De acordo com Fiorenza e Kreutz (2019), dentre os tipos de *firewall*, destacam-se os filtros de pacotes, filtros de estado de sessão, *firewall* de aplicação (Proxy) e *firewalls* de nova geração como os mais comuns. Na Tabela 2 é mostrada uma comparação entre os principais tipos de *firewalls*, destacando suas características e diferenças.

Para Fachinelli e Ahlert (2019), atualmente os *firewalls* podem variar em funcionalidade, desde configurações simples, como um roteador que utiliza filtros de pacotes, até soluções mais complexas, como *gateways* que integram filtros de pacotes e funções de Proxy na camada de aplicação. Enquanto os primeiros modelos de *firewall* se concentravam principalmente em isolar redes em um nível lógico, os sistemas modernos de segurança são predominantemente baseados em filtros de pacotes, *firewalls* de aplicação e inspeção de estados.

De acordo com Hoyer, Meirelles e Protector (2013), Cisco (2022), Palo Alto Networks (2023), os principais tipos de *firewall* são:

- ❑ ***Firewall* de aplicação ou proxy de serviços:** Podem inspecionar o conteúdo dos pacotes até o nível de aplicação, o que os tornas ideais para implementar políticas de segurança específicas de aplicativos, como inspeção profunda de pacotes (DPI) e filtragem de conteúdo. Eles analisam todo o tráfego de aplicativos e atuam como intermediários entre um dispositivo e a rede para a qual os dados estão sendo enviados. Em vez de permitir que o dispositivo se conecte diretamente à rede de destino, o *firewall* de aplicação intermedeia essa conexão. Isso proporciona uma camada adicional de segurança, pois todas as comunicações passam pelo *firewall*, que pode controlar e registrar o tráfego de dados. Além disso, o *firewall* pode armazenar em cache informações frequentemente acessadas, melhorando a eficiência da navegação na web ao reduzir o tempo de resposta para acessos repetidos;

- ❑ ***Firewall* de filtragem de pacotes:** É um método de controle de acesso em redes

onde um conjunto de regras definidas pelo administrador determina quais pacotes de dados podem passar através do *firewall*. Essas regras analisam principalmente informações nos cabeçalhos das camadas de rede e transporte dos pacotes. Filtram o tráfego com base em informações como endereços IP de origem/destino, portas TCP/UDP e protocolos. Existem dois tipos principais de filtragem: *stateless* (filtragem estática), que aplica regras fixas sem considerar o estado da conexão, e *stateful* (filtragem dinâmica), que leva em conta o estado da comunicação para tomar decisões. O objetivo é permitir ou bloquear o tráfego com base na política de segurança da rede, seguindo o princípio de "permitir por padrão" (permitir tudo que não é explicitamente proibido) ou "negar por padrão" (negar tudo que não é explicitamente permitido);

- ***Firewall Stateful Inspection* ou Inspeção de Estados:** Monitora continuamente o tráfego de dados para identificar padrões permitidos conforme suas regras. Ele realiza a inspeção de pacotes. Isso significa que é analisado o conteúdo dos pacotes de dados que passam por ele para identificar e bloquear atividades maliciosas. Esses padrões são armazenados e utilizados como referência para validar o tráfego subsequente, impedindo a passagem de pacotes não autorizados. É mantido um registro do estado das conexões de rede, permitindo um controle mais detalhado do tráfego. Os *firewalls stateful* são capazes de analisar o contexto das conexões e tomar decisões com base no estado da sessão (se a conexão está estabelecida ou não). Com base nos registros de estado das conexões e nas políticas de segurança configuradas, o *firewall Stateful* toma decisões inteligentes sobre permitir ou bloquear o tráfego de rede. Por exemplo, ele pode permitir que pacotes de dados associados a conexões estabelecidas continuem a passar sem novas verificações intensivas, o que melhora o desempenho da rede. Ao contrário da filtragem de pacotes simples, a inspeção de estados mantém uma tabela de estado para cada conexão, permitindo um controle mais eficaz e dinâmico do tráfego de dados;

- ***Firewall* de nova geração:** Representam uma evolução significativa nas arquiteturas de segurança. Esses *firewalls* integram funcionalidades de diversas gerações anteriores, incluindo IDS, controle de aplicativos, VPN (Virtual Private Network), entre outros em um único produto. Prometem alto throughput, uma interface de gerenciamento unificada, maior controle e visibilidade, além de melhor segurança e simplificação das regras de configuração. São projetados para suceder *firewalls stateful* e baseados em portas, agregando capacidades como prevenção de intrusão, filtragem de malware, controle de aplicações e usuários. Realizam inspeção profunda de pacotes, indo além da simples análise de porta e protocolo, incluindo inspeção de aplicativos e prevenção de intrusão com inteligência externa integrada, sendo projetados para oferecer uma segurança mais abrangente e contextual, capaz

de lidar com ameaças modernas e tráfego de aplicativos complexos. *Firewalls* de nova geração não são apenas dispositivos de segurança, mas também atuam como gateways de rede, gerenciando e protegendo o tráfego de dados que passa entre diferentes redes. Eles combinam as funcionalidades de roteamento e controle de tráfego com capacidades avançadas de segurança, tornando-se componentes importantes na infraestrutura de rede moderna.

Tabela 2 – Diferenças entre os principais tipos de *firewall*

Tipo de Firewall	**Características**
Firewall de filtragem de Pacotes	Filtra pacotes de dados com base em endereços IP, portas e protocolos. Menor overhead de processamento. Limitada capacidade de inspeção profunda.
Firewall de Aplicação (Proxy)	Atua como intermediário entre usuários e a Internet, filtrando tráfego de aplicativos específicos. Melhor controle de acesso e segurança para aplicações. Pode armazenar em cache para melhorar desempenho.
Firewall de Inspeção de Estado (Stateful)	Mantém tabelas de estado para monitorar conexões de rede. Permite ou bloqueia pacotes com base no estado da conexão. Maior capacidade de análise e controle comparado ao firewall de filtragem de pacotes (*stateless*).
Firewall de Próxima Geração	Integra funcionalidades de várias gerações de *firewalls* e sistemas de detecção de intrusão. Realiza inspeção profunda de pacotes, aplicativos e prevenção de intrusão. Atua como gateway da rede, analisando tráfego entre sub-redes. Oferece interface de gerenciamento unificado e funcionalidades *all-in-one*.

Fonte: Elaborado pelo autor.

2.3 Firewall Palo Alto 3260

Neste trabalho, o *firewall* da marca Palo Alto, modelo 3260, será utilizado para a aquisição dos *logs* que serão utilizados nos experimentos, sendo ele considerado um *Firewall de Próxima Geração*. Todo o conteúdo desta seção foi baseado nos PDFs do Guia do Administrador do Firewall (Palo Alto Networks, 2024a) e na Ficha Técnica da Série Palo Alto 3200 (Palo Alto Networks, 2024b).

O *firewall* Palo Alto, modelo 3260, é uma solução de *firewall* amplamente reconhecida. Projetado especialmente para implantações de gateway de Internet de alta velocidade, ele oferece capacidades avançadas de segurança e proteção. Este modelo é capaz de inspecionar e proteger todo o tráfego de rede, inclusive o tráfego criptografado, utilizando

recursos dedicados de processamento e memória para funções de rede, segurança, prevenção de ameaças e gerenciamento. Conhecido por sua eficácia em aplicar políticas de segurança granulares (permitem que administradores de rede definam quem pode acessar quais recursos), inspeção profunda de pacotes e detecção de ameaças em tempo real. Além disso, ele também suporta integração com outras soluções de segurança e oferece uma interface de gerenciamento centralizada para facilitar a administração e o monitoramento da segurança da rede.

O *firewall* vem equipado com a tecnologia App-ID, que identifica as aplicações que trafegam na rede, independentemente do protocolo, criptografia ou tática evasiva. Se uma aplicação mudar de portas ou comportamento, o App-ID consegue identificar esse tráfego de rede.

2.3.1 Logs de Firewall

Um *log* é um arquivo gerado automaticamente e carimbado no tempo, que fornece um rastro de auditoria para eventos do sistema no *firewall* ou eventos de tráfego de rede monitorados pelo *firewall*. As entradas de *log* contêm atributos, que são propriedades, atividades ou comportamentos associados ao evento registrado, como o tipo de aplicação ou o endereço IP de um atacante. Cada tipo de log registra informações para um tipo de evento específico. Por exemplo, o *firewall* gera um *Threat log* (*Logs* de Ameaças) para registrar o tráfego que corresponde a uma assinatura de spyware, vulnerabilidade ou vírus, ou um ataque de negação de serviço que corresponde aos limiares configurados para um escaneamento de porta ou atividade de varredura de host no *firewall*.

Já os *logs* de tráfego, que serão usados nos experimentos deste trabalho, exibem uma entrada para o início e o fim de cada sessão. Cada entrada inclui as seguintes informações: data e hora; zonas de origem e destino, grupos dinâmicos de endereços de origem e destino, endereços e portas; nome da aplicação; regra de segurança aplicada ao tráfego de rede; ação da regra (*Allow, Deny, Drop e Reset-Both*); interface de entrada e saída; número de bytes; e motivo do fim da sessão.

A coluna Ação indica se o *firewall* permitiu, negou ou descartou a sessão. Um *Drop* indica que a regra de segurança bloqueou o tráfego especificando qualquer aplicação, enquanto uma *Deny/Reset-Both* indica que a regra identificou uma aplicação específica.

Neste trabalho serão utilizados os *logs* de tráfego rede de uma instituição de ensino com mais de 3500 servidores públicos federais e mais de 15.000 alunos matriculados em cursos de graduação, mestrado e doutorado. Esses *logs* são capturados pelo *firewall* Palo Alto, modelo 3260. Cada *logs* representa uma única instância de comunicação entre dois dispositivos na rede, denominada sessão de tráfego de rede, sendo que cada sessão é identificada por uma combinação única de parâmetros, contendo no total 113 atributos, incluindo (Palo Alto Networks, 2024a; Palo Alto Networks, 2024b):

- **Endereço IP de Origem e Destino:** Os endereços IP dos dispositivos que iniciam e recebem a comunicação;
- **Portas de Origem e Destino:** Números da portas utilizadas pelos dispositivos para enviar e receber dados;
- **Protocolo:** O protocolo de rede usado na comunicação, como TCP, UDP, ICMP, etc;
- **Aplicação:** A aplicação específica que está sendo utilizada na comunicação, identificada pelo App-ID do Palo Alto;
- **Regra de Segurança Aplicada:** A política de segurança ou regra que se aplica a essa sessão, determinando se o tráfego é permitido, negado ou descartado;
- **Ação Tomada pelo Firewall:** A resposta do *firewall* à sessão (Allow, Deny, Drop e Reset-Both), como permitir, negar ou descartar o tráfego com base nas regras configuradas;
- **Interfaces de Rede:** As interfaces de entrada e saída no *firewall* através das quais o tráfego entra e sai;
- **Bytes Transferidos:** A quantidade total de dados transferidos durante a sessão;
- **Data e Hora:** O *timestamp* que indica quando a sessão foi iniciada e, opcionalmente, quando foi encerrada.

Cada entrada nos *logs* de sessões de tráfego de rede representa uma sessão única de comunicação entre dispositivos, permitindo aos administradores de rede monitorar e controlar efetivamente o tráfego, diagnosticar problemas de rede, analisar padrões de uso e garantir a conformidade com políticas de segurança estabelecidas.

É importante ressaltar que este trabalho se baseia na análise de *logs* de sessão de tráfego de rede, capturados do *firewall* Palo Alto, onde são tomadas ações (classe alvo Action: Allow, Drop, Deny e Reset-Both) como permitir, descartar e negar sobre esse tráfego de rede. Essas ações são tomadas com base em regras fixas, previamente configuradas no *firewall* pela instituição de ensino. Tais regras só são alteradas em casos de ajustes nas políticas de segurança, mudanças no cenário ambiental ou quando se identificam necessidades de correção, não possuindo caráter dinâmico.

Diferentemente da captura e análise de pacotes de rede, que examina detalhadamente cada pacote de dados transmitido, a análise de *logs* de sessão foca nas informações agregadas sobre as conexões estabelecidas, proporcionando uma visão mais macro do comportamento do tráfego. Dessa forma, analisam-se as sessões completas de comunicação em vez de inspecionar cada pacote individualmente, função geralmente desempenhada por

IDS e IPS. Esses sistemas realizam uma análise mais profunda e específica para detectar intrusões e comportamentos maliciosos. Nesse contexto, experimentos frequentemente utilizam conjuntos de dados como o *Kyoto 2006+* e abordam a detecção de ameaças como um problema clássico de classificação binária, diferenciando entre a classe normal e anomalias ou entre a classe normal e diferentes classes que representam padrões de ataque.

Nesse cenário, os algoritmos e modelos deste estudo são treinados a partir dos *logs* de tráfego de rede gerados pelo *firewall*. Esses *logs* registram as ações tomadas com base em regras estáticas, que determinam se um determinado tráfego deve ser permitido ou bloqueado. A partir dessa base de dados, os modelos aprendem a identificar padrões de comportamento conforme as decisões tomadas pelo *firewall*. A avaliação tem como objetivo medir a precisão dos algoritmos na previsão de tráfego de rede, determinando a ação mais adequada para cada sessão.

Por fim, há uma distinção importante no bloqueio de tráfego por *firewalls*: um tráfego pode ser bloqueado não por representar uma ameaça, mas por políticas de segurança definidas pelo administrador da rede. Estas políticas podem restringir o acesso a determinados servidores ou serviços sem que isso implique em uma tentativa de invasão ou atividade maliciosa, sendo apenas uma questão de conformidade com as regras da organização.

Além disso, quando um novo padrão de ataque é detectado e reconhecido como uma ameaça emergente, ele deve ser identificado e bloqueado. Contudo, o foco principal reside no bloqueio efetivo do tráfego malicioso - utilizando as ações padrão *drop, deny e reset-both* - e não na classificação detalhada da nova ameaça. O objetivo central é garantir a eficácia do bloqueio e não necessariamente catalogar novos padrões de ataque em classes específicas.

2.4 Considerações Finais

Neste capítulo, foi apresentada uma visão geral dos principais sistemas de segurança. Foram abordadas as características dos sistemas de detecção e prevenção de intrusões, além dos principais tipos de *firewalls* existentes. Por fim, foram detalhadas as principais características do *firewall* Palo Alto 3260, que será utilizado para a aquisição dos *logs* utilizados nos experimentos.

No próximo capítulo será fornecida uma visão geral sobre FCDs, destacando os conceitos de Mineração, Classificação e Agrupamento em FCDs.

Capítulo 3

Fluxo Contínuo de Dados

Durante muito tempo o Aprendizado de Máquina (AM) foi focado principalmente no aprendizado em *batch* (GAMA, 2010). Nesse cenário, um conjunto de dados representativos estava disponível para a fase de aprendizado, e um modelo de decisão era gerado e usado para fazer previsões futuras na tarefa para a qual foi desenvolvido. Nesse tipo de contexto, os exemplos são gerados de acordo com uma distribuição de probabilidade estacionária (GAMA, 2010). No AM tradicional, um conjunto de treinamento é disponibilizado previamente e, uma vez construído, o modelo de decisão permanece inalterado ao longo do tempo. Esses modelos são estáticos, ou seja, não incorporam novos dados após o treinamento inicial (GAMA, 2010).

Graças ao avanço da tecnologia, houve um aumento em larga escala na aquisição de dados nas mais diversas áreas. Contudo, os algoritmos tradicionais de AM possuem dificuldades em processar esse número elevado e contínuo de dados (SILVA et al., 2013). Gerenciar essas grandes quantidades de dados, geralmente em alta velocidade, que são sequências infinitas de dados gerados continuamente, é altamente complexo. Além disso, é muito difícil realizar tarefas como encontrar a média ou a entropia, pelo fato de que o fluxo não é finito e não há conhecimento prévio sobre o número de classes do problema. Os dados que possuem essas características são conhecidos como Fluxo Contínuo de Dados (FCDs) (GAMA, 2010). FCDs, do inglês, *data streams*, referem-se a uma sequência ordenada, em alta velocidade e potencialmente infinita, de exemplos $(x_1, x_2, x_3, \ldots, x_i, \ldots)$ lidos em ordem crescente dos índices i (GUHA et al., 2000; PAIVA, 2014).

Obter conhecimento útil a partir de FCDs tornou-se um grande desafio, e uma característica importante dos FCDs é a mudança de conceito ao longo do tempo, onde novos

conceitos podem surgir e desaparecer durante o fluxo. Por isso, houve a necessidade da criação de algoritmos específicos para lidar com FCDs.

Para Babcock et al. (2002), Aggarwal (2007) e Gama (2010) as principais características dos FCDs são:

- Dados potencialmente ilimitados;
- Dados chegam de forma contínua, e em geral, em alta velocidade;
- O sistema não tem controle sobre a ordem de chegada dos exemplos;
- O sistema deve estar apto a fazer predições a qualquer momento;
- Os dados não podem ser armazenados na memória, sendo assim, um exemplo deve ser processado uma única vez;
- Os dados possuem uma distribuição não-estacionária, ou seja, as características dos dados mudam com o passar do tempo.

3.0.1 Exemplos de Aplicações em Fluxo Contínuo de Dados

Atualmente há uma grande quantidade de dados em fluxo contínuo gerados por diferentes aplicações do mundo real. São exemplos de aplicações:

- **Jogos e eSports:** coleta de dados durante partidas online para análise de desempenho, detecção de trapaças ou para personalização de experiências de jogo em tempo real.
- **Saúde e monitoramento de pacientes:** dispositivos vestíveis que enviam continuamente dados sobre sinais vitais (frequência cardíaca, pressão arterial, oxigenação, etc.) para avaliação médica em tempo real;
- **Meteorologia:** compreender e prever os fenômenos meteorológicos que acontecem com a evolução do tempo;
- **Redes e telecomunicações:** registros de chamadas telefônicas, fluxos de cliques em páginas da web, detecção de intrusão em redes de computadores e sistemas de monitoramento de rede;
- **Redes sociais:** análise em tempo real de postagens, curtidas, compartilhamentos e comentários para detectar tendências, eventos ou sentimentos predominantes;
- **Segurança pública e vigilância:** processamento contínuo de imagens de câmeras de segurança e sensores urbanos para identificar comportamentos suspeitos, aglomerações ou situações de emergência;

- ❏ **Finanças:** detecção de fraudes em cartões de crédito e previsão de tendências do mercado da bolsa de valores.

3.0.2 Mineração em Fluxo de Dados

Muitos algoritmos de AM adotam a abordagem de algoritmos não incrementais, assumindo que todos os dados de treinamento estão disponíveis antecipadamente. Além disso, eles pressupõem que uma vez que o classificador é construído com base nos exemplos fornecidos durante a fase de treinamento, ele não será mais alterado. Esses algoritmos operam sob a premissa de que o conhecimento não pode ser continuamente atualizado e, em geral, preocupam-se mais com a precisão do que com a eficiência (PAIVA, 2014).

No mundo real, muitas aplicações são dinâmicas, onde os conceitos são atualizados diariamente, e consequentemente a aprendizagem também é contínua. A ideia de aprendizagem incremental ocorre a partir da observação do mundo real, e o conhecimento é sempre atualizado com base em novas observações, mantendo uma base contínua para reagir a novos estímulos. A aprendizagem incremental acredita que o ambiente pode se transformar ao longo do processo de aprendizado e requer que o conhecimento adquirido seja continuamente atualizado. Portanto, para o modelo se adaptar a essas mudanças, ele deve ser preciso e eficiente (FISHER, 1987; GEPPERTH; HAMMER, 2016).

De acordo com Langley (1995), Paiva (2014), Losing, Hammer e Wersing (2018) um algoritmo L é incremental somente se L recebe como entrada um exemplo de treinamento por vez e não reprocessa qualquer exemplo já processado, mantendo apenas uma estrutura de conhecimento na memória. Ainda de acordo com Langley (1995), um algoritmo incremental apresenta as seguintes características:

- ❏ Não retém na memória os dados previamente analisados;
- ❏ Pode ter sua fase de treinamento interrompida a qualquer momento para fazer uma predição.

Expandindo as características dos algoritmos incrementais, Pinto (2005) destaca algumas de suas vantagens:

- ❏ Possuem maior rapidez de resposta;
- ❏ Poupam memória do computador;
- ❏ O tempo de processamento para a geração do modelo de aprendizado não é desperdiçado.

O aprendizado incremental refere-se à capacidade de um modelo de atualizar seu conhecimento incorporando novos dados periodicamente, sem a necessidade de treinar

o modelo do zero com todos os dados disponíveis anteriormente, reduzindo o tempo de treinamento (LOSING; HAMMER; WERSING, 2018; LUO et al., 2020).

Para Dong et al. (2003), a mineração incremental estuda como atualizar os modelos e padrões considerando a parte incremental dos dados. No entanto, a mineração de fluxos de dados muitas vezes requer detecção online e dinâmica e resumo de mudanças interessantes.

Um algoritmo online precisa trabalhar continuamente no fluxo de dados e responder rapidamente às consultas solicitadas com base na parte dos dados recebidos até o momento (ASAI et al., 2002). Para Hoi et al. (2021), o aprendizado online é capaz de superar as desvantagens do aprendizado batch, pois o modelo preditivo pode ser atualizado instantaneamente para quaisquer novas instâncias de dados. Assim, os algoritmos de aprendizado online são muito mais eficientes e escaláveis para tarefas de aprendizado de máquina em grande escala em aplicações de análise de dados do mundo real, onde os dados não são apenas grandes em volume, mas também chegam a uma alta velocidade.

No Capítulo 4, será apresentado de maneira mais detalhada técnicas de agrupamento incremental e algoritmos que utilizam uma abordagem combinada das fases offline e online.

A Mineração de Dados (MD) tradicional emprega métodos como, classificação, mineração de regras de associação (padrões frequentes) e agrupamento (clustering). Novos algoritmos têm sido propostos para cada uma dessas abordagens, visando tratar os desafios associados a FCDs.

3.1 Classificação em Fluxo Contínuo de Dados

Na tarefa de classificação, inicialmente constrói-se um modelo de decisão com base em um conjunto de dados de treinamento. Esse modelo é então avaliado utilizando um conjunto de testes. No entanto, grande parte das técnicas voltadas para classificação operam em ambientes estáticos, onde uma vez criado o modelo de decisão, ele permanece inalterado. Além disso, na construção do modelo, os algoritmos presumem que todos os dados estão na memória, o que leva a diversas varreduras repetidas sobre eles. Dessa forma, os algoritmos de classificação concebidos para ambientes estáticos precisam ser reavaliados para se adequarem às características dos FCDs (AGGARWAL, 2007).

Ling, Ling-Jun e Li (2009) definem o problema de classificação em FCDs da seguinte forma: O FCDs é composto por uma série de exemplos, em que cada exemplo é constituído por um conjunto de d atributos. Cada atributo pode ser numérico ou categórico. Cada exemplo está associado a um rótulo, representado por um valor discreto que indica a classe à qual o exemplo pertence. O objetivo da classificação em FCDs é prever com alta precisão a classe dos exemplos desconhecidos que chegam continuamente.

Para Aggarwal (2007), um grande desafio na classificação em FCDs está relacionado à mudança de conceito, ou seja, à evolução dos dados ao longo do tempo. O principal impulsionador da mineração de FCDs no contexto de classificação é a necessidade de realizar apenas uma única varredura nos dados. Consequentemente, torna-se inviável utilizar os algoritmos convencionais de classificação, os quais pressupõem que os dados estão completamente disponíveis na memória, realizam várias varreduras e geram modelos de decisão estáticos (LING; LING-JUN; LI, 2009).

Segundo Gama e Rodrigues (2009), outro desafio significativo no aprendizado a partir de FCDs é manter constantemente um modelo de decisão com boa acurácia. Para isso, é necessário que os algoritmos sejam incrementais, pois sempre que novos dados são analisados, o algoritmo deve ser capaz de modificar o modelo de decisão atual. Novas classes podem surgir ou desaparecer; além disso, como as classes podem evoluir ao longo do tempo, torna-se imprescindível que os algoritmos para FCDs possuam a capacidade de decidir quais dados são relevantes, quais dados podem ser esquecidos e quais dados tornaram-se obsoletos para o problema atual.

3.2 Agrupamento em Fluxo Contínuo de Dados

O objetivo da tarefa de agrupamento de dados é separar os dados em diferentes grupos levando em consideração as suas características semelhantes (NGUYEN; WOON; NG, 2015). Portanto espera-se que os objetos de um grupo possuam alta similaridade e objetos de grupos diferentes possuam baixa similaridade (PAIVA, 2014).

O conceito de agrupamento de dados tem sido aplicado em diversas áreas do conhecimento, levando ao desenvolvimento de uma ampla variedade de algoritmos. A grande diversidade de algoritmos de agrupamento pode ter ocorrido devido ao fato de que não há uma definição geral de grupos, e, em geral, diferentes métodos de agrupamento se adaptam a diferentes tipos de aplicação (KAUFMAN; ROUSSEEUW, 2005; PAIVA, 2014). Dentre os algoritmos de agrupamento em cenários em *batch*, destacam-se o K-Means (LLOYD, 1982; MACQUEEN, 1967), o HDDBSCAN (CAMPELLO et al., 2015), o PAM (KAUFMAN; ROUSSEEUW, 2005), entre outros.

De acordo com Gama (2010), a maior dificuldade em realizar o agrupamento em FCDs está em manter de forma contínua e consistente bons grupos seguindo um critério de validação de agrupamento, bem como atender às restrições de memória e tempo. Isso ocorre devido à alta velocidade e à distribuição não estacionária dos dados.

Diversos algoritmos têm sido desenvolvidos ao longo dos anos com o objetivo de realizar o agrupamento em FCDs. Entre os principais, destacam-se o CluStream (AGGARWAL et al., 2003), o DenStream (CAO et al., 2006), o StreamKM++ (ACKERMANN et al., 2012), o D-Stream (CHEN; TU, 2007) e o Clustree (KRANEN et al., 2011).

Na maioria dos casos, é necessário realizar o processo de agrupamento em duas etapas distintas nos algoritmos de agrupamento em FCDs baseados em objetos: a etapa online (ou microagrupamento, que se refere à abstração dos dados) e a etapa offline (ou macroagrupamento, relacionada ao agrupamento propriamente dito), como ilustrado na Figura 1 (SILVA et al., 2013).

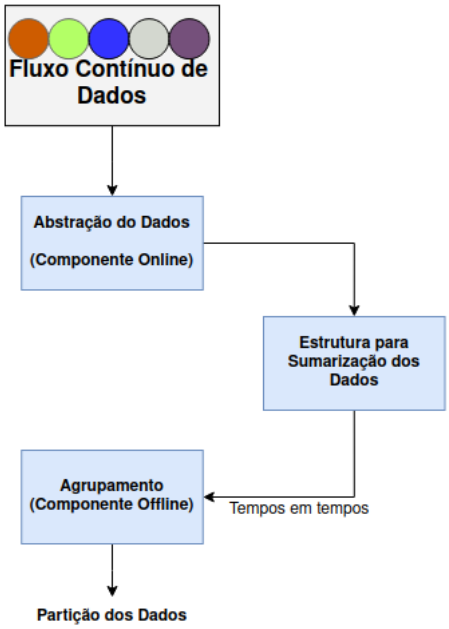


Figura 1 – Componentes online e offline aplicadas ao processo de agrupamento em FCDs.

Fonte: Adaptado de (SILVA et al., 2013)

A etapa online tem a responsabilidade de resumir os objetos que chegam ao fluxo, utilizando estratégias adicionais para capturar as principais características desses objetos. Essas estratégias permitem manter um conhecimento sobre os objetos originais sem a necessidade de armazená-los completamente em memória, o que é fundamental para lidar com as restrições de espaço e memória impostas pelos FCDs (AGGARWAL, 2003).

É fundamental, segundo Silva et al. (2013), que esse processo, além de resumir os objetos do fluxo, seja executado continuamente e, sobretudo, que os novos objetos sejam diferenciados dos antigos ao longo do tempo. O uso de estruturas como as janelas de tempo é então aplicado para gerenciar a atualização incremental do modelo. Assim, as mudanças nos FCDs podem ser acompanhadas e analisadas pelo usuário ao longo do tempo, mostrando que o modelo gerado permanece constantemente atualizado e consistente.

Segundo Silva et al. (2013), para preservar os dados originais é necessário utilizar estruturas de sumarização para contornar o problema de memória e espaço limitado. Na literatura, as estruturas mais comuns são: Vetor de Características, Vetor de Protótipos, Árvore de Conjunto de Objetos Representativos e Malha. Neste capítulo, serão abordados apenas os Vetores de Características, pois somente esta estrutura será utilizada no

trabalho. É possível encontrar uma descrição mais detalhada das demais estruturas de sumarização em (SILVA et al., 2013).

Um algoritmo de detecção de agrupamentos, conhecido como Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH), foi projetado para lidar com grandes volumes de dados, especialmente quando estes não cabem inteiramente na memória principal, sendo necessário, em muitos casos, uma análise única sobre cada instância de dado do conjunto. Este algoritmo é capaz de lidar com grandes conjuntos de dados através do uso de Vetores de Características, do inglês Cluster Feature Vector (CFV), que resumem os exemplos de grandes quantidades de dados em grupos, mantendo uma tripla: (ZHANG; RAMAKRISHNAN; LIVNY, 1996):

- **N:** número de exemplos no grupo;
- **LS:** soma linear dos N exemplos do grupo;
- **SS:** soma quadrada dos N exemplos do grupo.

Estes três componentes permitem obter medidas como *centroide*, representado pela Equação 1, e *raio* do grupo, apresentado na Equação 2:

$$\text{Centroide} = \frac{LS}{N} \tag{1}$$

$$\text{Raio} = \sqrt{\left(\frac{SS}{N}\right) - \left(\frac{LS}{N}\right)^2} \tag{2}$$

Segundo Gama (2010), um CFV possui as propriedades de *Aditividade* e *Incrementalidade*:

- Aditividade - Onde CF1 e CF2 são CFVs de dois grupos disjuntos, como mostrado na Equação 3.

$$CF1 + CF2 = (N_1 + N_2, LS_1 + LS_2, SS_1 + SS_2) \tag{3}$$

- Incrementalidade - Onde x é um exemplo adicionado ao CFV, como mostrado na Equação 4.

$$\begin{aligned} LS &\leftarrow LS + x \\ SS &\leftarrow SS + x^2 \\ N &\leftarrow N + 1 \end{aligned} \tag{4}$$

A utilização de CFVs para sumarizar grandes quantidades de dados foi introduzida no algoritmo CluStream, no estudo de Aggarwal et al. (2003), que faz uso de uma extensão dos CFVs denominada microgrupos. Além das três informações do CFV, cada microgrupo armazena a soma dos marcadores de tempo ($CF1^t$) e a soma quadrática dos marcadores de tempo ($CF2^t$).

Outro algoritmo tradicional de agrupamento em FCDs que utiliza o conceito de CFVs é o DenStream. Para manter e distinguir entre microgrupos e ruídos, são construídas estruturas de core-micro-cluster e outlier-micro-cluster. Cada uma dessas estruturas possui um peso associado que indica sua importância com base no tempo (CAO et al., 2006).

3.3 Considerações Finais

O objetivo deste capítulo foi apresentar uma visão geral sobre FCDs destacando exemplos de aplicações e suas diferenças em relação à abordagem de AM tradicional. Foi apresentado o processo de mineração em FCDs, incluindo as tarefas de classificação e agrupamento, suas etapas de operação e os principais desafios encontrados.

O próximo capítulo tem o objetivo de introduzir as características e conceitos fundamentais de Detecção de Novidade em FCDs e uma visão geral dos algoritmos desenvolvidos.

Capítulo 4

Detecção de Novidade em FCDs

Com o surgimento da mineração em FCDs, a tarefa de Detecção de Novidade (DN) traz novos desafios a serem abordados. DN é a habilidade de reconhecer quando uma instância não rotulada, ou um grupo delas, difere de maneira significativa dos conceitos previamente aprendidos (FARIA et al., 2016). DN possibilita o reconhecimento de novos conceitos em dados não rotulados.

Para Marsland, Shapiro e Nehmzow (2002) a DN é considerada um tópico essencial na Mineração de Dados (MD) e tem sido amplamente estudada em contextos batch, onde se parte do pressuposto de que um conjunto de dados representativo está disponível (MARKOU; SINGH, 2003; PIMENTEL et al., 2014).

Segundo Albertini e Mello (2012), muitas aplicações do mundo real têm motivado pesquisas de DN, que buscam identificar dados raros e desconhecidos onde há necessidade de criar modelos capazes de extrair informações de dados imprecisos devido a ruídos, anomalias, padrões escassos e dados sujeitos a mudanças nas características ao longo do tempo.

Para Spinosa, Carvalho e Gama (2009), em situações tradicionais de classificação, o modelo gerado sempre designa cada novo exemplo a uma das classes previamente conhecidas. Contudo, essa escolha pode ser inapropriada se o modelo gerado dos exemplos do conjunto de treinamento não refletir adequadamente a distribuição dos dados no momento da classificação. Esse problema pode ocorrer, entre outros motivos, devido a:

- **Mudança na distribuição dos dados:** Caso a distribuição dos dados tenha se alterado após a fase de treinamento, novos conceitos podem ter surgido, e as características dos conceitos antigos podem ter se modificado (KLINKENBERG, 2004). Esta é uma situação comum em várias aplicações reais;

- **Conjunto de treino pouco representativo:** É possível que algum conceito tenha sempre existido, mas não tenha sido incluído no modelo por não estar bem representado na fase de treinamento. Conseguir um conjunto de treino representativo, onde todos os conceitos estejam bem representados, geralmente não é uma tarefa fácil;

- **Dificuldade em obter exemplos representativos de todas as classes:** Em algumas aplicações, é impraticável obter exemplos de todas as condições possíveis relacionadas a uma determinada classe. Por exemplo, em problemas de detecção de intrusão, representar todas as ameaças existentes é inviável, dificultando a obtenção de uma boa descrição para a classe que representa a intrusão;

- **Presença de outliers:** Exemplos esparsos que não representam nenhum conceito não devem ser atribuídos a nenhuma classe.

Classificadores que apenas rejeitam ou deixam de classificar exemplos não permitem a identificação de mudanças ou novos conceitos, mantendo o classificador inadequado para a classificação de novos exemplos (LANDGREBE et al., 2006).

Para Gama (2010), a aplicação de técnicas de DN em FCDs enfrenta inúmeros desafios, uma vez que os conceitos podem variar ao longo do tempo. Diferentes formas de evolução dos padrões podem ocorrer nesse contexto, incluindo:

- Evolução de conceitos, onde novos padrões podem surgir;

- Desaparecimento ou Recorrência de Conceito, onde padrões podem desaparecer e, eventualmente, reaparecer;

- Mudança de conceito, também conhecida como deriva ou desvio, onde um padrão se transforma gradualmente ao longo do tempo;

- Ruído ou outlier, que são casos isolados e devem ser identificados, mas não utilizados.

4.1 Mudança de Conceito e Evolução de Conceito

Em aplicações que utilizam FCDs, a distribuição dos dados que representam os conceitos aprendidos pelo modelo pode mudar ao longo do tempo devido à natureza dinâmica do ambiente gerador (WIDMER; KUBAT, 1996). Como os conceitos evoluem ao longo do tempo, podem ocorrer mudanças e evoluções nos próprios conceitos, como evidenciado nos estudos de Roshan et al. (2018), Yuan et al. (2018) e Seth, Singh e Chahal (2021), focados na detecção de intrusões em redes de computadores.

Para Masud et al. (2010) uma das tarefas mais complexas em FCDs é a mudança de conceito, na qual o perfil das classes existentes se altera ao longo do tempo. Alguns exemplos de mudança de conceito são: mudanças no padrão de uma doença, mudanças de temperatura, mudanças no perfil de compras de um cliente ao longo dos anos, etc (FARIA et al., 2016). DN é valiosa em situações em que uma classe fundamental do problema está sub-representada no conjunto de treinamento. Conforme afirmam Markou e Singh (2003), essa é uma tarefa essencial, uma vez que para muitos problemas, é impossível determinar se os dados de treinamento disponíveis abrangem todas as possíveis classes de exemplos.

Ao contrário dos algoritmos de AM tradicionais, nos quais o número de classes é estabelecido previamente, e exemplos são associados a pelo menos uma das classes em um conjunto de classes predefinidas, em situações envolvendo FCDs essa suposição não se aplica. Na fase de treinamento, nem todas as classes são conhecidas, e exemplos de novas classes podem surgir ao longo do tempo. O fenômeno de evolução de conceito, também conhecido como classes emergentes, pode se manifestar em problemas como filtragem de spam, detecção de intrusão em redes de computadores e classificação de texto (FARIA et al., 2016).

Para Gama (2010), a detecção das mudanças na distribuição dos dados é fundamental porque observações antigas, que representam comportamentos passados, podem tornar-se irrelevantes para os modelos de aprendizado de máquina, afetando os resultados obtidos. Portanto, os algoritmos precisam descartar essas informações. Essas mudanças também podem ser classificadas de acordo com a rapidez com que ocorrem (GAMA, 2010):

- **Mudanças repentinas:** Manifestam-se abruptamente e de forma irreversível nos dados de exemplo;
- **Mudanças incrementais ou graduais:** Ocorrem gradualmente ao longo do tempo;
- **Mudanças recorrentes:** Alterações temporárias que tendem a ser revertidas com o tempo;
- **Ruído ou outliers:** São casos isolados que não devem ser considerados como mudanças significativas.

Em ambientes de FCDs, a latência refere-se ao intervalo de tempo entre a solicitação e a disponibilidade do rótulo de uma instância. Segundo Souza et al. (2015), três cenários de latência podem ser definidos:

- **Latência Nula/Zero**: os rótulos são disponibilizados imediatamente, sendo ideal para lidar com mudanças de conceito e evolução de conceito, devido à possibilidade de atualização imediata do modelo.

- **Latência Extrema/Infinita**: os rótulos nunca são disponibilizados, inviabilizando a atualização online do modelo e comprometendo a qualidade da classificação.
- **Latência Intermediária**: os rótulos são disponibilizados em algum momento futuro, possivelmente em ordem diferente da chegada dos dados. Este cenário reflete melhor as condições reais, pois a disponibilidade dos rótulos pode depender de fatores como a intervenção de especialistas.

A latência nula permite atualizações imediatas do modelo, sendo ideal para lidar com mudanças e evolução de conceitos. Entretanto, essa condição é rara em aplicações reais. Por outro lado, a latência extrema impede atualizações online, comprometendo gravemente a qualidade da classificação (SOUZA et al., 2015).

4.2 Visão Geral de DN em FCDs

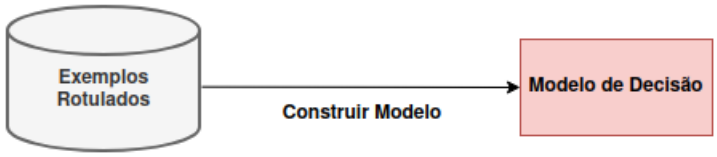
A DN pode ser abordada de várias maneiras, incluindo classificações binárias simples, onde um conceito representa a classe normal e as anomalias são identificadas pela falta de um conceito no modelo ou pela presença de um segundo conceito. No entanto, além dessa abordagem binária, é importante considerar a DN como uma classificação multi-classe, especialmente em cenários onde múltiplos conceitos podem coexistir nos dados. Contudo, alguns métodos que adotam a classificação multi-classe podem não lidar completamente com características de conjuntos de dados que evoluem ao longo do tempo, como mudanças graduais nos padrões de dados ou mudanças abruptas de conceito. Essas mudanças podem incluir a evolução de conceitos ou a mudança de conceito, onde vários padrões podem surgir simultaneamente durante um intervalo de avaliação. Portanto, é importante considerar a complexidade temporal dos dados ao escolher um método de DN, garantindo que ele seja capaz de capturar e adaptar-se eficazmente a mudanças nos padrões de dados ao longo do tempo (GAMA, 2010).

Normalmente as abordagens desenvolvidas para DN adotam um modelo de aprendizado offline-online. Na fase offline, um modelo de decisão é criado, e na fase online, toda vez que um exemplo é recebido, ele é classificado ou rejeitado pelo modelo de decisão. A capacidade de rejeitar exemplos durante a fase online permite a identificação de conceitos novos, o que pode indicar o surgimento de novos padrões, mudanças nos conceitos conhecidos ou a presença de ruído (MENDELSON; LERNER, 2020). A aplicação dessa técnica é fundamental para a mineração de FCDs, uma vez que auxilia na atualização do modelo de decisão.

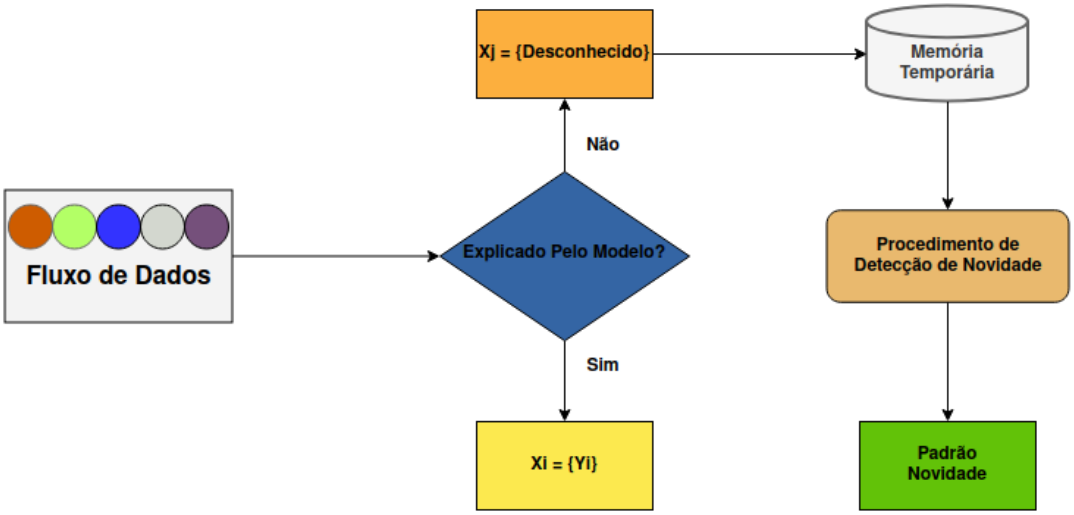
Na estratégia de aprendizado offline-online, muitos métodos de DN utilizam algoritmos de agrupamento não supervisionados. Esses algoritmos são usados tanto para criar o modelo inicial quanto para descobrir novos conceitos dos exemplos que o modelo não consegue explicar e que são marcados como desconhecidos (SPINOSA; CARVALHO; GAMA,

2009; GAMA; RODRIGUES, 2009; MASUD et al., 2011; FARIA; GAMA; CARVALHO, 2013).

Algumas abordagens adotam, na etapa inicial chamada offline, a premissa da disponibilidade de um conjunto de exemplos rotulados ou de conceitos conhecidos, os quais são utilizados para criar um modelo de decisão inicial (Figura 2a). Já na fase online, para cada exemplo não rotulado do FCDs, é verificado se o modelo consegue explicá-lo. Se sim, o exemplo é classificado; caso contrário, o exemplo é movido para uma memória temporária onde será utilizado para análises futuras, podendo representar um outlier, novidade ou extensão de um conceito já conhecido (Figura 2b).



(a) Fase Offline: Criação do modelo de decisão a partir de exemplos rotulados.



(b) Fase Online: Nesta etapa, ocorre a classificação de exemplos à medida que chegam ao longo do fluxo. Caso o modelo de decisão não seja capaz de classificar o exemplo, ele é categorizado como desconhecido e encaminhado para a Memória Temporária. Posteriormente, os exemplos desconhecidos passam por um processo de detecção de novidade para determinar se são Extensões, Outliers ou um Padrão Novidade.

Figura 2 – Visão geral da tarefa de DN.

Fonte: Adaptado de (FARIA et al., 2016)

É essencial descartar estruturas de classificação antigas que não desempenham um

papel relevante na tarefa atual do FCDs. Além disso, o modelo de decisão existente deve incorporar as mudanças geradas pela mudança de conceito; do contrário, será erroneamente identificada como um Padrão Novidade qualquer alteração no conceito previamente conhecido do problema (PAIVA, 2014).

4.2.0.1 Atualização do Modelo de Decisão e Mecanismo de Esquecimento

Para Spinosa, Carvalho e Gama (2009), Farid e Rahman (2012), Masud et al. (2011) e Paiva (2014), os algoritmos destinados à DN em FCDs devem manter a atualização contínua de seus modelos de decisão, visando lidar com evolução nos conceitos e a mudança de conceito. Existem três aspectos que devem ser levados em consideração nessa etapa:

- ❏ Tipo de atualização, podendo ser executado com ou sem feedback externo;
- ❏ Número de classificadores, o qual pode ser apenas um classificador ou um comitê de classificadores;
- ❏ Implementação de mecanismos de esquecimento para eliminar informações antigas.

Além de proceder com a atualização do modelo de decisão, algumas abordagens incorporam um mecanismo de esquecimento para dados desatualizados. Em propostas que se fundamentam em comitês de classificadores, identifica-se e elimina-se esses dados sempre que um novo classificador é treinado, substituindo um modelo anterior. Já em metodologias que empregam um único classificador, é integrado um mecanismo específico para efetuar esse esquecimento (FARIA; GAMA; CARVALHO, 2013; FARIA et al., 2016).

4.3 Métodos de Avaliação em Detecção de Novidade

Para avaliar a tarefa de classificação em FCDs, frequentemente são empregados dois métodos de amostragem:

- ❏ **Holdout:** Esta abordagem implica na divisão sequencial e dinâmica dos dados em conjuntos de treinamento e teste. O modelo é treinado nos dados iniciais (treinamento) e avaliado periodicamente em novos exemplos (teste), que simulam instâncias futuras não vistas. Conforme o fluxo evolui, o conjunto de treinamento pode ser atualizado com dados recentes, enquanto exemplos antigos que não contribuem mais para o processo de classificação são descartados (GAMA, 2010).
- ❏ **Prequential:** Neste método, as métricas de avaliação podem ser atualizadas de forma incremental, e cada exemplo individual pode ser empregado para testar o modelo antes do treinamento. Ele fornece uma avaliação contínua do desempenho do modelo, permitindo detectar mudanças na distribuição dos dados ao longo do

tempo e a adaptação do modelo em tempo real. De acordo com Bifet et al. (2009), o método *Prequential* tem a vantagem de não requerer um conjunto separado para testes, o que maximiza a utilização dos dados disponíveis.

Uma limitação observada na avaliação de algoritmos de DN em FCDs é a presença de exemplos não explicados pelo modelo, que são identificados como desconhecidos (FARIA et al., 2016). Em um estudo anterior de Faria et al. (2015), foi apresentada uma metodologia de avaliação que aborda essa questão, levando em consideração os exemplos desconhecidos e calculando medidas específicas para esses casos separadamente. Além disso, essa abordagem permite a associação de Padrões Novidade com as classes do problema, possibilitando a avaliação desses Padrões Novidade usando medidas de avaliação multi-classe.

Faria et al. (2015) e Faria et al. (2016) assumem que a matriz de confusão gerada pelos algoritmos de DN não é quadrada (Figura 3), já que o número de colunas aumenta sempre que um novo Padrão Novidade é detectado. Cada linha da matriz representa uma classe do problema, incluindo classes conhecidas e novidades, enquanto cada coluna representa uma das classes previstas pelo algoritmo de DN. Além disso, a última coluna da matriz representa os exemplos marcados como desconhecidos. É importante destacar que os Padrões Novidade detectados pelo algoritmo não têm uma correspondência direta com as classes do problema.

Para avaliar a matriz de confusão, Faria et al. (2015) e Faria et al. (2016) ressaltam que é essencial levar em conta os seguintes aspectos:

- ❏ Exemplos não explicados pelo modelo de decisão são marcados como desconhecidos;
- ❏ O algoritmo pode detectar menos Padrões Novidade do que o número de classes novidade. Isso ocorre quando o algoritmo não é capaz de distinguir os exemplos de todas as classes novidade. Além disso, uma classe pode ser representada por mais de um Padrão Novidade, o que implica na possibilidade de haver mais Padrões Novidade do que classes do problema;
- ❏ Para representar as mudanças ao longo do tempo, é necessário realizar avaliações periódicas da matriz de confusão.

4.4 Técnicas de agrupamento incremental e Algoritmos Online

Em geral, em um algoritmo típico de agrupamento incremental, cada exemplo pode ser associado a um grupo já existente ou formar um novo grupo. Algoritmos de agrupamento

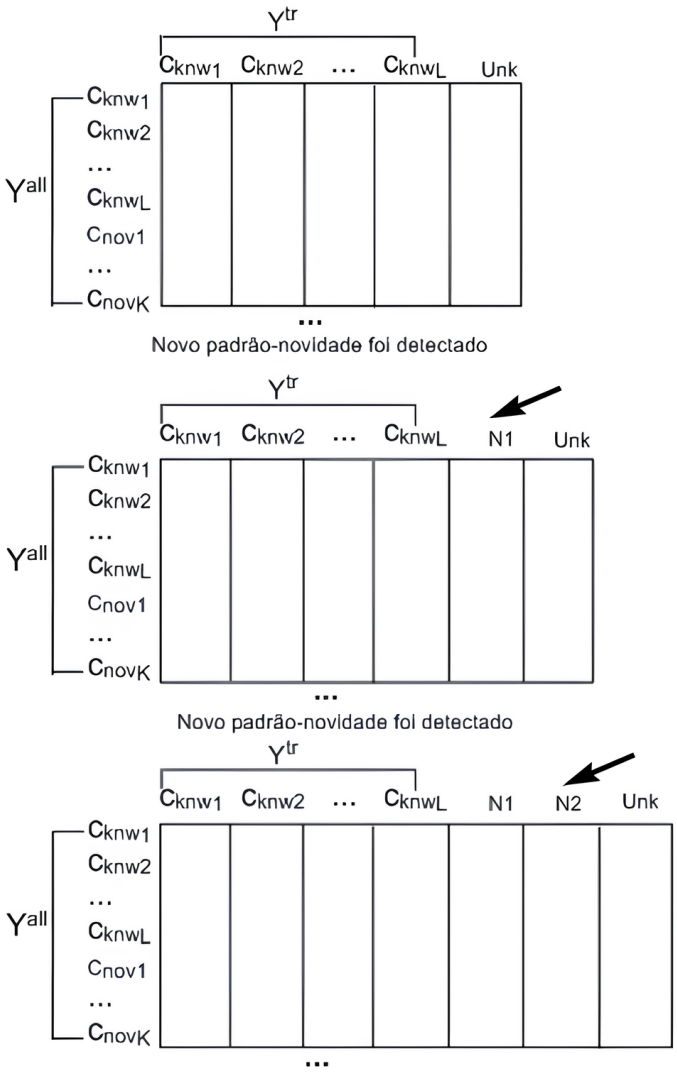


Figura 3 – Evolução da matriz de confusão ao longo do tempo.

Fonte: (PAIVA, 2014)

incremental constroem o modelo representativo dos dados à medida que novos exemplos são recebidos, permitindo sua aplicação em cenários de aprendizado (JAIN; MURTY; FLYNN, 1999).

Entre os diversos algoritmos de agrupamento incremental discutidos na literatura, o BIRCH (ZHANG; RAMAKRISHNAN; LIVNY, 1996) destaca-se como um dos mais abrangentes para lidar com grandes volumes de dados, considerando as restrições de armazenamento e tempo de processamento que impactam diretamente o processo de agrupamento. Segundo os autores, o BIRCH foi o primeiro algoritmo incremental a abordar outliers. Porém, é sensível à ordem de apresentação dos exemplos, semelhante ao COBWEB (FISHER, 1987).

Para Spinosa, Carvalho e Gama (2009), os algoritmos de agrupamento incremental são valorizados por sua capacidade de aprender continuamente à medida que novos exemplos

são incorporados. Isso significa que podem ajustar seus modelos e agrupamentos conforme novos dados são recebidos, o que é vantajoso em cenários onde a distribuição dos dados pode mudar ao longo do tempo.

Por outro lado, uma limitação desses algoritmos é que eles tendem a agrupar todos os exemplos disponíveis. Isso pode ser restritivo em problemas de DN, onde o interesse muitas vezes está em identificar exemplos que não se encaixam nos padrões existentes, como outliers ou novos padrões não previstos anteriormente. Portanto, embora úteis para aprendizado contínuo, sua aplicação direta na DN pode ser limitada devido à sua tendência de agrupar todos os exemplos sem distinção de novidade ou conformidade com padrões existentes. Além disso, um algoritmo incremental não é necessariamente um algoritmo online. Enquanto os algoritmos online geralmente processam os dados uma amostra por vez (ou em mini-batches), algoritmos incrementais podem operar em blocos maiores ou em dados armazenados, embora também possam ser usados em contextos online (HOI et al., 2021).

De acordo com Hoi et al. (2021), o aprendizado online é uma abordagem de aprendizado de máquina onde um modelo aprende a resolver tarefas preditivas ou de tomada de decisão processando instâncias de dados sequencialmente, uma de cada vez. O objetivo é maximizar a precisão das previsões ou decisões com base no conhecimento adquirido de tarefas anteriores e, possivelmente, em informações adicionais. Diferente dos métodos tradicionais de aprendizado em lote, que treinam um modelo com todo o conjunto de dados de uma só vez, o aprendizado online é ideal para lidar com FCDs, comuns em muitas aplicações reais.

Embora frequentemente associado aos algoritmos online, o aprendizado incremental representa uma categoria mais ampla. Algoritmos incrementais também têm a capacidade de atualizar seus modelos com novos dados sem a necessidade de reprocessar o conjunto completo anterior. No entanto, diferentemente dos algoritmos online — que são projetados para processar os dados amostra por amostra (ou em mini-batches), em tempo real e sob restrições computacionais — os algoritmos incrementais podem operar de maneira mais flexível, processando blocos de dados de tamanhos variados e, muitas vezes, sem exigência de atualização imediata. Dessa forma, todo algoritmo online é, por definição, incremental, pois atualiza o modelo de forma progressiva. Contudo, nem todo algoritmo incremental é online, pois pode não operar com dados sequenciais ou em tempo real. Essa distinção é importante na escolha da abordagem mais adequada para aplicações, como em sistemas de recomendação, detecção de anomalias e adaptação dinâmica a ambientes não estacionários Hoi et al. (2021).

Em geral, o aprendizado online pode ser classificado em três categorias principais:

- ❏ Aprendizado supervisionado online, onde há feedback completo;
- ❏ Aprendizado online com feedback limitado;

- ❑ Aprendizado não supervisionado online, onde não há feedback.

Os algoritmos que serão utilizados neste trabalho empregam, na fase online após a geração do modelo de decisão (fase offline), o aprendizado não supervisionado online. Este tipo de aprendizado lida com tarefas em que o modelo recebe apenas uma sequência de instâncias de dados sem qualquer feedback adicional (por exemplo, rótulos de classe verdadeiros) durante o processo de aprendizado online. O aprendizado não supervisionado online pode ser considerado uma extensão natural do aprendizado não supervisionado tradicional para lidar com fluxos de dados, que é tipicamente estudado de forma batch. Exemplos de aprendizado não supervisionado online incluem tarefas de agrupamento online, redução de dimensão online e detecção de anomalias online, entre outras. O aprendizado não supervisionado online tem suposições menos restritivas sobre os dados, não exigindo feedback explícito ou informações de rótulo, que podem ser difíceis ou caras de obter (HOI et al., 2021; AGGARWAL, 2014; BERKHIN, 2006).

Neste trabalho, a novidade não será abordada por meio do surgimento de novas classes (Evolução de Conceito). Em vez disso, a ideia é aprender com o fluxo à medida que seu comportamento se altera, utilizando mudanças de conceito para indicar possíveis ampliações dos conceitos já aprendidos através de extensões e recorrências. O objetivo é detectar novos padrões nos dados ou identificar padrões que aparecem, desaparecem e retornam durante o FCDs. Mudanças mais abruptas que indicam novos Padrões Novidade também serão consideradas uma extensão de um conceito conhecido e classificadas em uma das classes existentes, dado que o número de classes é fixo e não varia ao longo do FCDs. O objetivo é controlar o tráfego de rede, permitindo ou bloqueando o acesso, independentemente de se tratar de uma ameaça nova ou recorrente. Novos padrões são identificados, porém sempre serão classificados como permitido ou negado.

Como o fluxo de dados ocorre a partir do tráfego de rede capturado por um *firewall*, seu comportamento pode mudar tanto de forma abrupta quanto gradualmente. Isso é comum em ambientes de rede, onde é necessário adaptar-se a mudanças frequentes nas políticas de segurança, alterações na rede ou novas ameaças. Portanto, será utilizado um algoritmo offline-online (MINAS) que ajusta seu modelo continuamente à medida que novos dados chegam, com a capacidade de DN, mecanismos de esquecimentos para dados que já não contribuem para o fluxo e mecanismos que identificam recorrência de padrões nos dados.

Essa abordagem será comparada com um algoritmo incremental para FCDs que não utiliza DN, o Clustream Adaptado. A comparação visa avaliar se a detecção de novos padrões, por meio da extensão de conceito é mais eficaz para *logs* de sessão de tráfego de rede. Além disso, a análise pode ajudar na verificação de como as mudanças de conceito ocorrem, de forma mais suave e esperada ou de maneira mais abrupta.

4.5 Considerações Finais

Neste capítulo foi abordada a tarefa de DN em FCDs, foram apresentadas as características fundamentais juntamente com Mudança de Conceito e Evolução de Conceito. Também foi apresentada uma visão geral dos algoritmos desenvolvidos para DN, abrangendo suas fases online e offline. Por último, foram apresentados os Métodos de Avaliação em Detecção de Novidade.

No próximo capítulo serão apresentados os trabalhos relacionados e o modo como esta pesquisa se relaciona com eles.

Capítulo 5

Trabalhos Relacionados

A mineração em Fluxo Contínuo de Dados (FCDs) se revela de suma importância devido à enorme quantidade e à velocidade com que os dados são gerados atualmente. Nesse sentido, tem-se observado um crescente desenvolvimento de trabalhos voltados para enfrentar esse desafio. Entre esses esforços, a Detecção de Novidade (DN) desponta como uma tarefa de destaque em trabalhos voltados para detecção de intrusão, como exemplificado pelo *IDSA-IoT: an intrusion detection system architecture for IoT networks* (CASSALES et al., 2019) e pelo *ASTREAM: Data-stream-driven scalable anomaly detection with accuracy guarantee in IIoT environment* (YANG et al., 2022). No entanto, percebe-se a falta de algoritmos de classificação na literatura especializada que abordem de forma adequada a análise de registros de *log* do tráfego de rede em dispositivos de *firewall*, com o intuito de prever a ação recomendada *(Allow, Drop, Deny e Reset-Both)*, classificando as sessões de tráfego de rede de forma contínua, incremental e como uma tarefa de DN. Essa deficiência destaca a necessidade de explorar alternativas que identifiquem e monitorem continuamente as sessões de tráfego de rede. O objetivo é prever a ação recomendada para o tráfego de rede de forma mais próxima da realidade, pois os *logs* de sessão de tráfego de rede são gerados em alta velocidade e de maneira contínua, necessitando de abordagens e algoritmos voltados para FCDs.

Neste capítulo, são abordados os trabalhos relacionados e são apresentados os aspectos do estado da arte do tópico Detecção de Novidades em Fluxo de Dados, fazendo uma comparação com a Arquitetura IDSA-IoT proposta por (CASSALES et al., 2019), que, embora não esteja diretamente relacionada à previsão de ações recomendadas para o tráfego de rede em *firewalls*, a arquitetura utiliza o algoritmo de DN MINAS em FCDs para detecção de intrusão, mostrando capacidade de lidar com tráfego de rede e proteger

a infraestrutura de rede contra ameaças emergentes e ataques cibernéticos.

Também é discutido o trabalho de Komadina et al. (2024), intitulado "Análise Comparativa de Abordagens de Detecção de Anomalias em Logs de Firewall", que realiza uma avaliação comparativa de algoritmos de AM não-supervisionados e supervisionados na detecção de anomalias inseridas em *logs* de *firewall*.

Por fim, são conduzidas análises comparativas com outros três estudos (ERTAM; KAYA, 2018; SHARMA; WASON; JOHRI, 2021; ALJABRI et al., 2022) que abordam o tópico "previsão da ação recomendada para o tráfego de rede em *firewalls* classificados em (*Allow, Deny, Drop e Rest-Both*)". Porém, esses estudos não empregam algoritmos incrementais e de DN em FCDs; em vez disso, são utilizados algoritmos de aprendizado em *batch*, métodos tradicionais de AM. Até o presente momento, não foram encontrados na literatura trabalhos que integrem de forma conjunta esses dois tópicos. Este trabalho se difere por aplicar algoritmos baseados em microgrupos, de forma incremental e utilizando DN, em registros de *log* de tráfego de rede capturados por um *firewall*. O objetivo é classificar a ação recomendada para as sessões de tráfego, considerando os dados como um FCDs.

5.1 Arquitetura IDSA-IoT

Cassales et al. (2019) propõem uma arquitetura de detecção de intrusão para ambientes IoT chamada *An Intrusion Detection System Architecture for IoT Networks* (IDSA-IoT). O objetivo desta arquitetura é monitorar uma rede local composta por diversos tipos de dispositivos IoT e identificar atividades intrusivas. A proposta utiliza os recursos de borda da nuvem para aumentar a escalabilidade e reduzir a latência. Três métodos de detecção de novidade foram usados para aprender padrões emergentes de tráfego de rede. Os algoritmos utilizados incluem ECSMiner (MASUD et al., 2011), AnyNovel (ABDALLAH et al., 2016) e MINAS (FARIA et al., 2016). A abordagem aproveita os serviços da nuvem pública para realizar o processamento offline para operações mais precisas, como melhorias de modelo, e faz uso de recursos de rede de borda para coletar e analisar fluxos de dados. A detecção de intrusão é realizada por meio de algoritmos de processamento de fluxo que são executados próximos aos dispositivos e sensores IoT na borda da rede. Assim, o processamento acontece em locais como um *Single Board Computer* (SBC) ou roteadores, que possuem poder de processamento suficiente. Dessa forma, o tráfego de dados é analisado localmente, evitando a necessidade de transferir esses dados para o centro da rede, como uma nuvem pública, economizando tempo e recursos. Devem ser enviados para a nuvem pública apenas os dados que não foram explicados pelo modelo e que contribuem para a evolução dos modelos. Os dados armazenados na nuvem, originários de diferentes regiões da rede, são analisados para realizar ajustes nos modelos existentes ou para treinamento de novos modelos.

Nos experimentos, foi utilizado o conjunto de dados *Kyoto 2006+*, no qual dados foram coletados de 348 *Honeypots* (máquinas isoladas, equipadas com vários softwares que possuem vulnerabilidades conhecidas e são expostas à Internet, com intuito de atrair ataques), incluindo sistemas com Windows (várias versões), Linux/Unix (Solaris 8, MacOS X), impressora de rede, eletrodomésticos, aparelho de TV, gravador de HDD, entre outros. O conjunto de dados é composto por 24 atributos, com o atributo alvo classificado como normal, ataque conhecido ou ataque desconhecido. O treinamento offline foi realizado com 72.000 exemplos, o que equivale a 10% do conjunto de dados, usando a técnica de holdout. Foram utilizados apenas os exemplos de tipos de ataque conhecidos com um mínimo de 10.000 ocorrências.

O experimento avaliou a qualidade de DN para os três algoritmos propostos na arquitetura IDSA-IoT: ECSMiner, MINAS e AnyNovel. Para avaliar a arquitetura, foram utilizadas as métricas de qualidade de *Fnew*, *Mnew* e *erro*. A métrica *Fnew* (ou Falso Positivo) representa a proporção de exemplos de uma classe normal que foram classificados como novidade ou extensão, enquanto a métrica *Mnew* (ou Falso Negativo) representa a fração de exemplos da classe novidade que foram rotulados como normal. Por fim, a métrica *erro* é calculada como a soma dos valores de falso positivo e falso negativo dividida pelo número total de exemplos classificados (PAIVA, 2014). Além dessas métricas de qualidade, também foi registrada a quantidade de requisições de classificação realizadas por especialistas.

Os algoritmos também foram avaliados quanto ao uso de recursos computacionais e ao tempo total de processamento em dispositivos com recursos limitados.

Em seu estudo, Cassales et al. (2019) concluíram que o algoritmo ECSMiner é a melhor opção em um cenário em que existe um alto feedback especializado e grande poder de processamento. Porém, em cenários de IoT, raramente esses requisitos são facilmente atendidos.

Apesar de apresentar valores mais altos na métrica de *erro*, o algoritmo MINAS manteve a métrica Fnew constante e baixa. Demonstrou-se adequado para dispositivos limitados, com baixo consumo de recursos computacionais. Além disso, o algoritmo pode obter melhores resultados com o ajuste de seus hiperparâmetros, reduzindo a taxa de erro.

Por fim, os resultados obtidos no AnyNovel variaram consideravelmente, mas com uma melhor parametrização ou seleção de características pode-se obter melhores resultados. No entanto, não houve nenhuma configuração nos testes realizados que resultasse em uma boa métrica em uma categoria e fosse aceitável em outras.

A proposta de detectar tentativas de intrusão por meio de algoritmos de DN em FCDs em dispositivos IoT oferece informações sobre o desempenho do algoritmo MINAS na detecção de ameaças e na identificação de padrões de ataques não identificados anteriormente. Isso ajuda a orientar a implementação do MINAS em classificar registros de *log*

de *firewall*, considerando seu potencial para prever ações recomendadas, sua adaptabilidade em ambientes de rede e sua eficiência em dispositivos com recursos computacionais limitados.

5.2 Análise Comparativa de Abordagens de Detecção de Anomalias em Logs de Firewall: Integrando a Geração Simplificada de Logs de Segurança e Detecção de Ataques Gerados Artificialmente

Komadina et al. (2024) propõem um método para gerar *logs* contendo registros relacionados a ataques, eliminando a necessidade de um ambiente de teste dedicado ou de controles de segurança instalados. Foram utilizados registros de *firewall* internos de um operador nacional de sistema de transmissão de eletricidade. Como esses *logs* não continham anomalias em sua forma original, Komadina et al. (2024) aplicaram o método proposto para criar registros relacionados a ataques e integrá-los aos *logs* pré-existentes. O método possui a capacidade de gerar anomalias que se assemelham de perto ao comportamento real dos atacantes, permitindo uma integração com os *logs* de *firewall* existentes para testes realistas. Além disso, Komadina et al. (2024) realizaram uma avaliação comparativa de algoritmos de AM não supervisionados e supervisionados na detecção de anomalias injetadas nos *logs* do *firewall*.

Os *logs* de *firewall* usados por Komadina et al. (2024) foram adquiridos de um *firewall* Check Point e continham um total de 19 atributos. Desse total, foram selecionados os seguintes atributos: *timestamp* da conexão, endereço IP de origem e destino, porta de origem e destino, protocolo utilizado e ação do *firewall*. Os registros de dados foram inicialmente filtrados pelo atributo *action* do *firewall*. Esse atributo pode ter um dos quatro valores: *accept* (aceitar), *drop* (descartar), *bypass* (contornar) e *monitor* (monitorar). A ação *accept* foi o valor predominante e representou mais de 70% dos casos, enquanto *drop* ocorreu em cerca de 29% dos casos. Os outros dois valores tiveram uma participação mínima, somando aproximadamente 170 registros. Somente os registros com a ação *accept* foram preservados para integração aos *logs* contendo as anomalias injetadas. Um script foi utilizado para converter os *logs* de pacotes em *logs* compatíveis com os *logs* do *firewall* alvo. Komadina et al. (2024) destacam que, devido aos *logs* de *firewall* fornecerem informações significativamente menos detalhadas em comparação com capturas completas de pacotes, uma inspeção manual completa de todos os atributos foi realizada nos *logs* anômalos gerados. Essa inspeção foi importante para garantir que os logs refletissem com precisão o que seria esperado no caso de um ataque real à rede alvo. Como resultado, a

tarefa no estudo não foi de classificação multiclasse com as ações: *accept* (aceitar), *drop* (descartar), *bypass* (contornar) e *monitor* (monitorar). Em vez disso, foram inseridos os registros anômalos nos *logs* de *firewall* já existentes e introduzido um rótulo para o atributo alvo que distingue entre registros normais e anômalos. Dessa forma, o problema de detecção de anomalias foi transformado em uma tarefa de classificação binária.

Nos experimentos envolvendo algoritmos não supervisionados, Komadina et al. (2024) avaliaram um total de 13 modelos diferentes de AM, incluindo o *Copula-Based Outlier Detection* (COPOD), *Isolation Forest* e *Outlier Detection with Kernel Density Functions* (KDE). Para os experimentos com algoritmos supervisionados, foram igualmente avaliados 13 modelos diferentes, entre eles *Naive Bayes*, *Random Forest* e Perceptron.

Komadina et al. (2024) concluíram em seus estudos que nenhum dos algoritmos de aprendizado não supervisionado investigados apresentou desempenho satisfatório na detecção de anomalias. Esse resultado confirma que as anomalias injetadas representam etapas de ataque disfarçadas e bem integradas aos *logs* de *firewall* pré-existentes. Em contrapartida, a utilização de métodos de aprendizado supervisionado proporcionou resultados significativamente melhores e satisfatórios, mostrando o potencial do método apresentado para gerar e integrar anomalias na detecção de anomalias.

Apesar de utilizar logs de *firewall*, o trabalho de Komadina et al. (2024) empregou algoritmos de AM tradicionais que não são voltados para FCDs. Além disso, o problema foi transformado em uma classificação binária com o intuito de identificar anomalias, ao contrário do tema proposto neste trabalho, que realiza uma previsão multiclasse, permitindo quatro ações possíveis: *Allow (Permitir), Deny (Negar), Drop (Descartar) e Reset-Both (Resetar Ambos).* Outra diferença é que, além de identificar anomalias, o objetivo é determinar como cada tipo de tráfego deve ser classificado. Isso é importante porque um tráfego que não é uma invasão também pode ser bloqueado se as políticas de segurança determinarem que o firewall deve fazer esse tipo de bloqueio.

5.3 Classificação Multiclasse de arquivos de *log* de *firewall* usando Máquinas de Vetores de Suporte (SVM)

Para Ertam e Kaya (2018), posicionar-se estrategicamente em conformidade com as atividades dos usuários finais em uma rede corporativa é de suma importância para a gestão eficiente da infraestrutura de rede. A quantidade considerável de dados gerados pelos usuários, dependendo do tamanho da rede, destaca a relevância crítica desse alinhamento. Dispositivos de *firewall* desempenham um papel essencial ao permitir ou bloquear o tráfego com base nas políticas estabelecidas, mediante a análise dos dados gerados. A configuração precisa desses *firewalls* é essencial para assegurar o funcionamento

adequado e a segurança das redes de comunicação, contribuindo assim para a utilização ativa e eficaz das tecnologias de comunicação empresarial e outros dispositivos de rede existentes.

Em seu estudo, Ertam e Kaya (2018) destacam a importância vital de analisar os registros nos dispositivos de *firewall* para controlar o tráfego da Internet com base nessa análise. Os autores classificaram registros de *log*, provenientes do dispositivo de *firewall* Palo Alto 5020 na Universidade Firat, utilizando o classificador de Máquina de Vetores de Suporte (SVM) multiclasse. A avaliação do desempenho do classificador envolveu a obtenção de valores de *precisão*, *recall* e *F1-Score*. No total, 65532 exemplos foram examinados, e o atributo *Action* foi designado como o atributo alvo.

Ertam e Kaya (2018) utilizaram o classificador SVM para realizar a classificação em quatro classes distintas. Para isso, a abordagem adotada consistiu em dividir o problema em múltiplos classificadores binários, uma estratégia comum em problemas de classificação multiclasses, em que as classes são agrupadas e tratadas de forma binária. Diversas funções de kernel foram consideradas no processo de classificação, incluindo linear, polinomial, RBF e sigmoidal. A escolha da função de kernel adequada foi baseada na revisão de funções frequentemente utilizadas na literatura. Métricas de desempenho, como *precisão, recall e F1-Score*, foram empregadas para avaliar o desempenho do classificador, sendo a *precisão* usada para calcular a relevância da informação resultante em relação à informação desejada. Para classificar os dados de *log* do *firewall*, foram escolhidos 11 atributos do conjunto de dados (*Source Port, Destination Port, NAT Source Port, NAT Destination Port, Elapsed Time (sec), Bytes, Bytes Sent, Bytes Received, Packets, Packets Sent, Packets Received*). Ao selecionar dados, é preciso escolher atributos que possuam mais valores numéricos. O atributo *Action*, que é um dos atributos não numéricos, foi aceito como uma classe e pode ser rotulado como *(Allow, Deny, Drop e Reset-Both).*

Os experimentos conduzidos revelaram que a função de kernel Sigmoid obteve o melhor valor de *recall*. A maior *precisão* foi alcançada pelo classificador com a função de kernel linear, enquanto o *F1-Score*, que é calculado como a média harmônica de *precisão* e *recall*, indicou que o classificador mais eficaz foi aquele com a função de kernel RBF. Os classificadores utilizando ativação linear e sigmoidal também demonstraram desempenho satisfatório; por fim, a função de kernel polinomial resultou nos piores valores de *precisão*, *recall* e *F1-Score*.

A conclusão de Ertam e Kaya (2018) enfatiza a extrema importância de analisar os registros de *log* nos dispositivos de *firewall* para orientar o tráfego da Internet com base nessas análises. Como resultado, destacou que o SVM, com a função de kernel Sigmoid, alcançou uma *recall* de 98,5%, enquanto a função de kernel linear obteve uma *precisão* de 67,5%. O *F1-Score* mais elevado, atingindo 76,4%, foi observado no classificador utilizando a função de kernel RBF. Considerando valores médios, o estudo concluiu que o classificador mais eficiente é aquele que utiliza a função de kernel RBF.

5.4 Classificação Otimizada de Dados de *Log* de *firewall* utilizando Técnicas de ensemble heterogêneas

Em um estudo sobre aprimoramento da classificação de dados de *log* de *firewall*, Sharma, Wason e Johri (2021) abordaram a crescente necessidade de filtros mais inteligentes e automáticos para pacotes de dados na web. O trabalho propõe uma solução otimizada por meio do uso de técnicas de aprendizado de máquina. Este estudo tratou da categorização de pacotes de dados de *firewall*, onde o atributo "action" pode ser classificado em quatro rótulos distintos: *(accept, drop, reject, TCP reset)*. Esses rótulos, que definem as ações associadas aos dados, foram codificados e convertidos de valores categóricos para valores numéricos. Em seguida, nove atributos foram selecionados dentre todos os atributos disponíveis para cada sessão de tráfego. São eles: *NAT Source Port, NAT Destination Port, Source Port, Destination Port, Bytes Sent, Bytes Received, Elapsed Time (sec), Packet Sent e Packet Received*. Os dados foram padronizados, resultando em uma transformação do conjunto de dados. Esta abordagem garantiu que os valores de cada atributo fossem reescalonados para uma escala entre -1 e 1, utilizando normalização. Assim, os valores de cada atributo foram ajustados para que se encontrassem dentro desse intervalo específico.

Os pesquisadores analisaram 65532 exemplos de arquivos de *log*. Os dados que constituem os registros de *log* foram retirados da Universidade Firat, que utilizou o dispositivo de *firewall* Palo Alto 5020. Os dados processados foram divididos em uma proporção de 3:1 para treinamento e teste. Essa proporção indica que, dos dados disponíveis, 75% são usados para treinamento e 25% são usados para teste. Primeiramente, foram criados cinco modelos básicos de forma independente: *K-nearest neighbor (KNN), Decision Tree Clasifier (Dtree), Support Vector Machine (SVM), Logistic Regression (LogReg) e Stochastic Gradient Descent (SGDC)*. O modelo de classificação SVM utilizado no estudo aplicou a função de kernel linear para separar os pontos de dados. Posteriormente, esses modelos foram combinados para formar dois modelos de ensembles avançados: um classificador ensemble de votação heterogêneo e um classificador ensemble de empilhamento (*stacking ensemble*).

No modelo *stacking ensemble* os modelos base foram combinados e as saídas (meta previsões) dos modelos base foram alimentadas juntas no meta classificador para a previsão final. O classificador Random Forest, que é em si um modelo de ensemble, foi usado como meta classificador no modelo *stacking ensemble*.

No modelo ensemble de votação heterogêneo as saídas dos modelos base foram consideradas em conjunto e a previsão final foi feita com base no voto de consenso desses modelos. Por fim, o desempenho de todos os modelos considerados no trabalho de (SHARMA;

WASON; JOHRI, 2021) foi comparado com base em três métricas: *Precisão*, *Recall* e *F1-Score*.

Sharma, Wason e Johri (2021) avaliaram os modelos por meio de métricas de desempenho, revelando que o modelo *stacking ensemble* alcançou uma *precisão* de 91%, *Recall* de 82% e *F1-Score* de 85%, sendo ele portanto proposto como o modelo otimizado para a classificação de dados de *log* de *firewall*. Já o ensemble de votação heterogêneo alcançou uma *precisão* de 74%, *Recall* de 74% e *F1-Score* de 74%.

Sharma, Wason e Johri (2021) ressaltam que esses resultados superaram consideravelmente outras abordagens para a otimização da classificação de dados *firewall*, destacando a eficácia desta proposta no cenário atual de filtragem de dados em rede.

Sharma, Wason e Johri (2021) concluíram que analisar os arquivos de *log* do *firewall* é essencial para a geração de modelos de aprendizado de máquina que podem influenciar as ações inteligentes do *firewall* em pacotes de dados. Levando isso em consideração, o estudo concentrou-se em encontrar um modelo otimizado para a classificação de dados de *log* de *firewall*.

5.5 Classificação de logs de firewall usando modelos de aprendizado de máquina multiclasse

O estudo conduzido por Aljabri et al. (2022) busca simplificar a análise de *logs* de *firewall* por meio do emprego de técnicas de Aprendizado de Máquina (AM) e Aprendizado Profundo (AP). Foram criados modelos capazes de examinar esses registros e categorizar as ações a serem adotadas diante das sessões recebidas, como “Permitir”, “Descartar”, “Negar” ou “Redefinir ambos”. Eles destacaram ainda a avaliação do impacto da inclusão de dois atributos adicionais ao conjunto tradicionalmente usado em estudos correlatos.

Aljabri et al. (2022) utilizaram diversos algoritmos de AM e AP, como K-Nearest Neighbours (KNN), Naïve Bayes (NB), Árvores de Decisão (AD), Random Forest (RF) e Rede Neural Artificial (RNA), visando prever a ação recomendada em relação a cada sessão à medida que o tráfego flui pela rede.

Os classificadores foram avaliados por meio de métricas como *Acurácia*, *Precisão*, *Recall*, *F1-Score* e *Área sob a curva ROC*. Além disso, Aljabri et al. (2022) empregaram validação cruzada k-fold, utilizando k=10, para avaliar a eficácia dos modelos propostos.

Em seu estudo, Aljabri et al. (2022) utilizaram um conjunto de dados de *logs* de rede coletados entre os dias 18 e 27 de maio de 2021. Os registros de *log* empregados foram extraídos de um *firewall* utilizado pela organização. O arquivo CSV continha originalmente 1.048.576 instâncias, divididas em quatro classes diferentes. Foi realizado o pré-processamento do conjunto de dados antes de qualquer análise para deixar o arquivo pronto para uso pelos modelos para treinamento e teste.

O conjunto de dados de *log* continha mais de um milhão de entradas, das quais apenas 2,68% eram sessões em que a ação tomada foi *"Deny"*, enquanto as sessões para as quais a ação tomada foi *"Allow"* representaram 88,23% dos dados. Isso mostra que os dados sofreram desequilíbrio onde pelo menos um dos rótulos de classe não estava balanceado em número quando comparado com outros rótulos de classe.

Considerando que o conjunto de dados era altamente desbalanceado, Aljabri et al. (2022) aplicaram subamostragem para igualar o número de instâncias de cada tipo de ação. Assim, 28.133 instâncias de cada tipo de ação foram selecionadas aleatoriamente para ter um total de 112.532 instâncias. Quanto aos atributos categóricos, foi utilizada uma técnica de codificação de dados para convertê-los em valores numéricos. Além disso, as características numéricas foram normalizadas dentro de um intervalo de -1 e 1.

O conjunto de dados empregado engloba 25 atributos, combinando características categóricas e numéricas, as quais detalham as sessões de tráfego de rede, incluindo o atributo alvo e a ação correspondente. De acordo com a pesquisa conduzida por Aljabri et al. (2022), foram realizados dois experimentos. No primeiro experimento, um conjunto de 11 atributos foi selecionado com base em sua frequente utilização em outros estudos relacionados *(Bytes, Bytes Received, Bytes Sent, Destination Port, Elapsed Time, NAT Destination Port, NAT Source Port, Packets, Packets Received, Packets Sent e Source Port)*. Para o segundo experimento, foram adicionados os atributos *Application* e *Category*. O atributo *action* foi designado como o atributo alvo, com quatro valores possíveis: "Permitir", "Descartar", "Negar" ou "Redefinir ambos".

Dentre todos os algoritmos empregados nos experimentos, Aljabri et al. (2022) verificaram que o Random Forest (RF) alcançou o melhor desempenho em todas as métricas de avaliação nos dois experimentos, com taxas de *acurácia* e *precisão* de 99,11% e 99,64% para os experimentos um e dois, respectivamente. O KNN obteve uma *precisão* e *acurácia* de 98,8% no primeiro experimento e 99,2% no segundo. De maneira geral, todos os modelos apresentaram um desempenho satisfatório nas avaliações. Além disso, foi observado que todos os modelos obtiveram resultados superiores no segundo experimento em comparação ao desempenho no primeiro experimento. Isso enfatiza a importância da inclusão dos atributos *Application* e *Category*, sobretudo para o desempenho do algoritmo de Redes Neurais Artificiais (RNA).

Por fim, Aljabri et al. (2022) concluíram que seu estudo apoia o uso de técnicas de AM na classificação automatizada das ações registradas nos *logs* do *firewall*, visando aprimorar a segurança das redes organizacionais de forma ágil e confiável. Os resultados obtidos não apenas têm o potencial de fortalecer a segurança e a proteção oferecidas por *firewalls* e softwares antivírus, mas também de impulsionar o desenvolvimento de novas abordagens para prevenir ameaças cibernéticas.

5.6 Considerações Finais

Neste capitulo, foram abordados os trabalhos relacionados com foco na detecção de intrusões em tráfego de rede, fazendo uma comparação com a Arquitetura IDSA-IoT que utiliza o algoritmo de DN MINAS em FCDs para detecção de intrusão. Também foi discutido um estudo sobre detecção de anomalias em *logs* de *firewall*. Por fim, foi conduzida uma análise comparativa com três estudos que abordam o tópico "previsão da ação recomendada para o tráfego de rede em *firewalls*", classificados em (*Allow, Deny, Drop e Rest-Both*). Na Tabela 3 são apresentados os principais aspectos desses trabalhos, destacando objetivos e tipos de abordagem de aprendizado.

Tabela 3 – Resumo dos Trabalhos Relacionados

Autores/Título	**Ano**	**Objetivo**	**Abordagem de Aprendizado**
Cassales et al.	2019	Propor a arquitetura IDSA-IoT para detecção de intrusões em redes IoT, avaliando algoritmos de DN (ECSMiner, MINAS, AnyNovel)	Fluxo Contínuo de Dados (FCDs)
Komadina et al.	2024	Criar logs realistas com ataques simulados e identificar anomalias usando AM, em classificação binária (normal/ataque)	Aprendizado Batch
Ertam e Kaya	2018	Classificação de logs de firewall utilizando SVM com dados do firewall Palo Alto 5020	Aprendizado Batch
Sharma, Wason e Johri	2021	Classificação de logs de firewall uando técnicas de ensemble heterogêneas com dados do firewall Palo Alto 5020	Aprendizado Batch
Aljabri et al.	2022	Classificação de logs de firewall com Random Forest, KNN e outros algoritmos, utilizando dados de um firewall corporativo privado	Aprendizado Batch

Fonte: Elaborado pelo autor.

Ainda assim, todos os trabalhos encontrados até o momento na literatura foram baseados em algoritmos de AM tradicional para classificar a ação recomendada em quatro rótulos distintos (Permitir, Negar, Descartar ou Redefinir Ambos) e não adaptados para FCDs. Outros trabalhos abordaram o problema no contexto de FCDs, porém são volta-

dos especificamente para detecção de intrusão, tratando-o como um problema clássico de classificação binária, prevendo a ação como normal ou ameaça, como em um Sistema de Detecção de Intrusão (IDS), ou separando entre a classe normal e diferentes classes de ameaça, ao invés de um *firewall* baseado em regras que, além de bloquear intrusões, bloqueia tráfego que não são ameaças, mas que, por políticas internas de segurança, aplicam o bloqueio do tráfego de rede. Portanto, nenhuma abordagem foi feita com algoritmos voltados para trabalhar com FCDs e DN. Assim, os algoritmos MINAS e Clustream Adaptado, que serão utilizados nos experimentos, se posicionam como uma abordagem diferente das já aplicadas para a previsão da ação recomendada em sessões de tráfego de rede, possuindo a capacidade de operar de forma a aprender com novos dados à medida que chegam, ajustando seu modelo continuamente com cada nova amostra de dados, o que pode ser essencial em aplicações onde a rapidez na adaptação aos dados mais recentes é fundamental e há necessidades de identificar padrões não convencionais em FCDs. Além disso, como o fluxo de dados ocorre a partir do tráfego de rede capturado por um *firewall*, seu comportamento pode mudar tanto de forma abrupta quanto gradualmente. Isso é comum em ambientes de rede, onde é necessário adaptar-se a mudanças frequentes nas políticas de segurança, alterações na rede ou novas ameaças.

No próximo capítulo serão apresentados os métodos e funcionamento do MultI-class learNing Algorithm for data Streams (MINAS), um algoritmo de aprendizado para FCDs com DN. Também serão apresentados os métodos e o funcionamento do algoritmo incremental para agrupamento em FCDs, conhecido como *Clustream*. Além da versão original do algoritmo, será discutida a adaptação realizada para permitir sua aplicação na recomendação de ações para o tráfego de rede em *firewalls*.

Capítulo 6

Classificação da Ação Recomendada para Tráfego de Rede em Fluxos Contínuos de Dados: MINAS e CluStream

Neste capítulo, são descritos os métodos e algoritmos empregados na pesquisa para a classificação das ações recomendadas para o tráfego de rede em FCDs utilizando *logs* de *firewalls*. Dois algoritmos principais foram selecionados para esta tarefa. O primeiro é uma adaptação do CluStream, um algoritmo de agrupamento incremental que atualiza continuamente o modelo à medida que novos dados chegam, sendo eficaz na gestão e agrupamento de dados dinâmicos. O segundo é o MINAS, um algoritmo especializado em DN, que oferece uma abordagem para identificar e adaptar-se a novos padrões e anomalias em FCDs.

Esses dois algoritmos foram selecionados por serem algoritmos incrementais baseados em microgrupos e bem estabelecidos na área de mineração de fluxo de dados, representando abordagens distintas dentro desse domínio. Ambos possuem suas habilidades em aprender e evoluir o modelo à medida que novos dados chegam, além de sua eficácia em lidar com FCDs. Por essas razões, serão utilizados nos experimentos para a previsão das ações recomendadas para o tráfego de rede.

Adicionalmente, será comparado o desempenho da abordagem que utiliza DN para criar extensões de conceitos conhecidos, como no caso do MINAS, com o algoritmo CluStream Adaptado, que não emprega DN. Os dados experimentais serão obtidos a partir

de *logs* de sessões de tráfego de rede capturados pelo *firewall*.

A análise tem como objetivo avaliar o desempenho dos dois algoritmos na previsão de ações recomendadas para o tráfego de rede, tratando os dados como um FCDs. Além disso, busca identificar qual abordagem oferece o melhor desempenho na classificação desses dados.

6.1 Metodologia Utilizada

Para atingir os objetivos propostos, a metodologia adotada neste trabalho envolveu as seguintes etapas:

1. **Preparação dos Dados:** Inicialmente, os *logs* de sessões de tráfego de rede capturados pelo *firewall* foram divididos em duas fases: offline, utilizada para inicialização do modelo, e online, destinada ao processamento contínuo. Em seguida, os dados foram pré-processados para remover atributos irrelevantes e codificar variáveis categóricas, visando à adaptação dos dados aos algoritmos de aprendizado.

2. **Implementação e Adaptação dos Algoritmos:**

   - O algoritmo CluStream foi adaptado para criar microgrupos rotulados durante a fase offline, representando cada uma das classes do problema *(Allow, Deny, Drop e Reset-Both)*. Durante a fase online, o CluStream Adaptado foi ajustado para atualizar esses microgrupos a medida que cada exemplo chega durante o fluxo, lidando com mudanças de conceito sem a necessidade de retreinar o modelo por completo. Por fim, é realizado uma fase de macroagrupamento, onde os microgrupos são agrupados em conjuntos maiores (macrogrupos) com o intuito de rotular os exemplos em uma das classes do atributo alvo *action*.
   - O algoritmo MINAS teve seus hiperparâmetros ajustados para funcionar em um cenário de classes fixas, onde não surgem novas classes ao longo do tempo. Em vez de criar novas classes para acomodar novidades, o algoritmo foi configurado para expandir os microgrupos das classes existentes ao receber esses novos dados. Nesse contexto, os exemplos são incorporados por microgrupos já existentes que representam a classe do problema, ou novos microgrupos são criados, por meio do processo de DN, para ampliar o conceito das classes previamente conhecidas.

3. **Avaliação do Desempenho:** Os métodos de avaliação incluíram a geração de uma matriz de confusão detalhada, a partir da qual foram calculadas métricas de desempenho essenciais. Entre elas, a minimização de erros críticos de classificação, como a confusão entre permitir tráfegos que deveriam ser bloqueados e bloquear

tráfegos que deveriam ser permitidos. Também é calculado a taxa média de acerto das classes, levando em consideração o desempenho global, mesmo em cenários de classes desbalanceadas. Por fim, também são calculadas métricas clássicas, como precisão, recall e F1-score, para uma análise abrangente. Nos experimentos, foram realizadas simulações em diferentes cenários de fluxo de dados, abrangendo dados coletados em dias e horários distintos. Esse procedimento permitiu avaliar a robustez dos algoritmos frente às variações nos padrões de tráfego de rede.

4. **Comparação dos Algoritmos:** Foi realizada uma análise comparativa entre o CluStream Adaptado e o MINAS para identificar qual dos algoritmos é mais eficaz na previsão da ação recomendada para o tráfego de rede. A comparação avaliou se o MINAS, com sua capacidade de detectar novos padrões (novidades), é mais eficaz do que o CluStream Adaptado, que utiliza uma abordagem incremental. Devido às diferenças em seus processos de agrupamento e classificação, esses algoritmos podem ser comparados para determinar qual deles é mais adequado para classificar os *logs* de sessão de tráfego de rede. Além disso, foi realizada uma comparação com dois algoritmos supervisionados e incrementais, ambos de latência nula, nos quais os rótulos são disponibilizados de forma imediata. Os algoritmos utilizados nessa comparação foram o SVM incremental e o Adaptive Random Forest Classifier, ambos consolidados na literatura científica.

Esta metodologia foi utilizada para verificar a validade da hipótese de que métodos de mineração de FCDs que utilizam técnicas de agrupamento incremental para atualizar modelos de decisão à medida que os dados chegam do fluxo, serão capazes de classificar ações recomendadas para o tráfego de rede, mantendo a precisão e a eficiência mesmo em cenários onde os dados geralmente chegam em alta velocidade, são sequências infinitas geradas continuamente e podem ter alta variação no padrão de dados. A performance (precisão e eficiência) foi avaliada por meio de métricas como acurácia, F1-Score, entre outras.

6.2 Justificativa da escolha do MINAS e CluStream Adaptado

O MINAS e o CluStream Adaptado foram escolhidos para realizar a previsão da ação recomendada para o tráfego de rede usando os logs de sessões de tráfego capturados pelo firewall Palo Alto 3260. Esses dois algoritmos foram selecionados por serem algoritmos incrementais baseados em microgrupos e bem estabelecidos na área de mineração de fluxo de dados, representando abordagens distintas dentro desse domínio. Essas escolhas foram feitas por várias razões que os tornam adequados para esse contexto:

- **Custo computacional:** A abordagem incremental permite que o CluStream Adaptado e o MINAS atualizem os microgrupos de maneira eficiente, sem a necessidade de reprocessar todo o conjunto de dados, o que reduz o custo computacional;

- **Aprendizado Contínuo:** Diferente dos algoritmos tradicionais de AM que necessitam de treinamento em *batch*, o MINAS e o CluStream Adaptado ajustam seu modelo continuamente à medida que novos dados chegam. Isso permite uma resposta rápida a mudanças no tráfego de rede, adequando-se melhor aos dados que chegam em alta velocidade e de maneira contínua;

- **Grande Volume de Dados:** O MINAS e o CluStream Adaptado são projetados para lidar com grandes volumes de dados dinâmicos, mantendo a eficiência no agrupamento e na atualização contínua dos microgrupos conforme novos dados são recebidos;

- **Latência Extrema/Infinita:** Ambos os algoritmos são baseados em microgrupos e são utilizados para classificar fluxos de dados em evolução em cenários nos quais os rótulos verdadeiros são considerados sujeitos a um atraso infinito, ou seja, nunca são fornecidos. Dessa forma, o modelo não depende de rótulos verdadeiros para realizar atualizações durante a fase online;

- **Erros de Configuração em Regras de Firewall:** Ambos os algoritmos apresentam baixa dependência de rótulos corretos, o que significa que o desempenho não é significativamente impactado por erros nas regras do *firewall*. Isso ocorre porque sua base de funcionamento é centrada na formação de microgrupos, que são determinados por padrões intrínsecos dos dados e não diretamente pelos rótulos. Dessa forma, erros específicos nas regras não comprometem a identificação desses padrões. Além disso, com o desempenho satisfatório dos modelos, é possível realizar uma análise comparativa entre as divergências observadas nas ações recomendadas pelos algoritmos e aquelas tomadas pelas regras configuradas. Esse processo auxilia na identificação e correção de possíveis erros de configuração, garantindo maior eficácia na definição das regras de *firewall*.

- **Mecanismos de Esquecimento:** O MINAS e o CluStream Adaptado incluem mecanismos para descartar dados antigos que já não são relevantes, ajudando a manter o modelo atualizado sem ser sobrecarregado por dados obsoletos. Isso é fundamental para garantir que o modelo permaneça eficiente e preciso ao longo do tempo;

- **Capacidade de Detecção de Novidade:** O MINAS é projetado para identificar e adaptar-se a novos padrões de dados que não foram previamente observados. Em

um ambiente de rede, onde novas ameaças e padrões de tráfego podem surgir inesperadamente, essa capacidade pode ajudar especialmente em mudanças abruptas no tráfego de rede ou em cenários com classes altamente desbalanceadas;

Modelos Tradicionais de Aprendizado de Máquina, como Árvores de Decisão (AD), Rede Neural Artificial (RNA) ou Máquina de Vetores de Suporte (SVM), geralmente necessitam de re-treinamento completo com novos dados, o que pode ser demorado e não prático para FCDs. Além disso, esses modelos não possuem capacidades intrínsecas de DN ou de adaptação contínua à medida que novos dados chegam, tornando-os menos adequados para cenários dinâmicos.

Essas características os diferenciam de outros algoritmos propostos em experimentos anteriores para prever a ação recomendada para o tráfego de rede. Assim, esses dois algoritmos podem se mostrar adequados para ambientes de rede dinâmicos, onde é fundamental adaptar-se rapidamente a novas ameaças e mudanças no tráfego. Além disso, devido às diferenças em seus processos de agrupamento e classificação, esses algoritmos podem ser comparados para determinar qual deles é mais adequado para classificar os *logs* de sessão de tráfego de rede.

6.3 CluStream e Adaptação do CluStream

Nesta seção, será apresentado o funcionamento geral do algoritmo CluStream original, bem como sua adaptação para possibilitar a classificação das ações recomendadas para o tráfego de rede, utilizando os *logs* do firewall. Primeiro, é apresentado o algoritmo CluStream, incluindo suas principais fases e hiperparâmetros. Em seguida, é abordado como o CluStream foi adaptado para lidar com a classificação de tráfego de rede, destacando as modificações realizadas para atender às necessidades específicas dos experimentos.

6.3.1 CluStream

Dentre os diversos algoritmos de agrupamento, existem alguns algoritmos baseados em particionamento, como o CluStream, que separam as instâncias de dados em um número predefinido de grupos, com base na similaridade ou distância em relação aos centróides do grupo. Este tipo de algoritmo identifica exclusivamente grupos hiperesféricos e requer que o número de grupos seja especificado como hiperparâmetro. Algoritmos baseados em particionamento normalmente são fáceis de implementar (ZUBAROĞLU; ATALAY, 2021).

O CluStream (AGGARWAL et al., 2003) é um framework elaborado para trabalhar com o agrupamento em FCDs, fazendo uso da ideia de microgrupos. O agrupamento dos dados possui duas etapas, uma fase online e uma offline. Na fase online, é mantido periodicamente um resumo estatístico dos dados. Já na fase offline, um analista utiliza o resumo

criado anteriormente para compreender rapidamente os grupos presentes no FCDs. O resumo estatístico dos dados é armazenado em microgrupos, que podem ser interpretados como extensões temporais do CF-Vetor proposto no algoritmo BIRCH (ZHANG; RAMAKRISHNAN; LIVNY, 1996). Um hiperparâmetro q deve ser definido pelo usuário e, em cada momento, q microgrupos são alocados na memória.

Cada microgrupo armazena as informações do CF-Vector (N, LS e SS), $CF1^t$ que armazena a soma dos marcadores de tempo (*timestamps*) e a soma quadrada dos marcadores de tempo armazenada no $CF2^t$. O algoritmo K-Means é executado sobre um conjunto inicial de dados para obter os q microgrupos iniciais, sendo o tamanho desse conjunto inicial definido pelo hiperparâmetro *InitNumber*. Em seguida, cada novo exemplo é absorvido pelo microgrupo mais próximo, determinado com base na menor distância euclidiana entre o exemplo e os centróides dos microgrupos. Se essa distância estiver dentro de um limite predefinido, o novo objeto é incorporado ao microgrupo. O limite máximo de um microgrupo é determinado multiplicando o desvio padrão da distância dos exemplos do grupo ao seu centróide (também conhecido como raio) por um fator *fat* (fator de alargamento usado para determinar o raio de aceitação de novos exemplos em um microgrupo). Se nenhum dos microgrupos puder incorporar o novo dado, um novo microgrupo será criado. Para a criação de um novo microgrupo dois microgrupos devem ser unidos ou um microgrupo mais antigo deve ser removido, pois é necessário sempre manter q microgrupos na memória. O microgrupo mais antigo é excluído quando o seu marcador de tempo cai abaixo de um determinado limiar δ, que é um hiperparâmetro de entrada. Nesse ponto, ele é considerado um valor atípico (outlier) e deve ser eliminado. Se o microgrupo mais antigo não for removido, os dois microgrupos mais próximos são combinados, utilizando a propriedade da aditividade do CF-vetor (AGGARWAL et al., 2003).

É possível realizar macroagrupamentos dos microgrupos mantidos na fase online usando o algoritmo CluStream. Para isso, uma fase offline é executada, na qual os microgrupos são agrupados em k grupos. O algoritmo K-Means é utilizado para esse agrupamento, sendo necessário informar o hiperparâmetro de entrada k, que deve ser menor que o número de microgrupos (AGGARWAL et al., 2003).

6.3.2 Adaptação do CluStream

Foi implementada uma versão adaptada do *CluStream* original, voltada para a classificação de *logs* de sessões de tráfego de rede. O algoritmo opera em três fases principais: uma fase offline para a inicialização dos microgrupos rotulados, uma fase online para o processamento contínuo dos dados e uma fase de macroagrupamento para consolidar os microgrupos em macrogrupos. A seguir, serão detalhadas cada uma dessas fases e os hiperparâmetros utilizados, proporcionando uma visão de como o algoritmo é ajustado

para atender aos objetivos específicos da classificação de tráfego de rede.

6.3.2.1 Fase Offline

Na fase offline ou fase de treinamento (Figura 4), o algoritmo utiliza o *K-Means* em um conjunto inicial de dados rotulados para agrupar os exemplos em um número predefinido de microgrupos e construir o modelo para realizar as predições. Para isso, os seguintes passos são realizados:

1. **Carga dos dados:** Primeiramente, os dados são carregados de arquivos CSV que contêm *logs* de sessões de tráfego de rede. Um dos arquivos representa a fase offline, enquanto o outro corresponde à fase online, permitindo a distinção entre as etapas de treinamento e simulação do FCDs;

2. **Definição dos hiperparâmetros:** O número de microgrupos (`n_micro_clusters`), o limite máximo de um microgrupo que é determinado multiplicando o desvio padrão da distância dos exemplos do grupo ao seu centróide (também conhecido como raio) por um fator (`fat`), o limite de tempo para remoção de microgrupos (`delta`), e o número de macrogrupos (`n_macro_clusters`) são definidos;

3. **Aplicação do K-Means:** O algoritmo *K-Means* é utilizado para agrupar os dados offline em um número fixo de microgrupos, definido pelo hiperparâmetro `n_micro_clusters`. Esta aplicação resulta na formação dos microgrupos rotulados e na atribuição de cada exemplo de dados ao microgrupo mais próximo;

4. **Inicialização dos CF-Vectors:** Os vetores de características (*CF-Vectors*), que incluem o número de exemplos (`N`), soma linear (`LS`), soma quadrada (`SS`) e o tempo da adição do ultimo exemplo para cada microgrupo (*CF1t*) são inicializados;

5. **Cálculo dos CF-Vectors:** Para cada exemplo, os valores dos *CF-Vectors* são atualizados com base em qual microgrupo o exemplo foi atribuído;

6. **Atribuição de Rótulos:** Cada microgrupo é rotulado com base na classe predominante (*Action*) dos exemplos que o compõem. Embora possam existir diferentes rótulos dentro de um mesmo microgrupo, o rótulo final atribuído será aquele que ocorrer com maior frequência entre os exemplos. Os rótulos das classes são definidos da seguinte forma: (*0 - Allow, 1 - Deny, 2 - Drop, 3 - Reset-Both* e 4 - para novos microgrupos criados durante a fase online). Detalhes adicionais sobre a atribuição de rótulos são explicados nas fases online e de macroagrupamento do algoritmo.

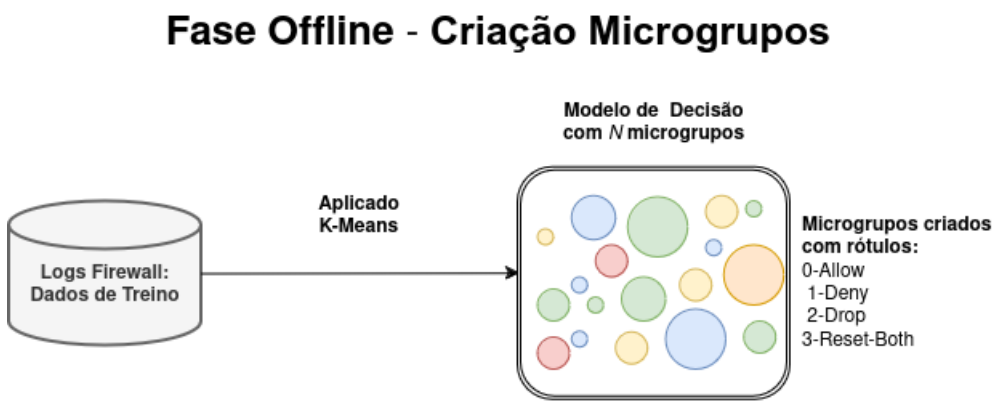


Figura 4 – Fase Offline do CluStream Adaptado.

Fonte: Elaborado pelo autor.

6.3.2.2 Fase Online

Na fase online (Figura 5), o algoritmo processa dados que chegam continuamente e ajusta os microgrupos conforme necessário. O processamento dos dados online envolve:

1. **Processamento de cada exemplo de dados:** Cada novo exemplo é processado individualmente, verificando qual microgrupo é o mais próximo (menor distância euclidiana do exemplo em relação ao centróide do microgrupo);

2. **Verificação do raio:** O algoritmo calcula o raio do microgrupo mais próximo para determinar se o novo exemplo pode ser incorporado a ele. O raio é determinado como o desvio padrão dos exemplos dentro do microgrupo, multiplicado pelo fator `fat`;

3. **Incorporação ao Microgrupo ou Criação de um Novo:** Se o exemplo estiver dentro do raio do microgrupo, ele é incorporado ao microgrupo existente. Caso contrário, um novo microgrupo é criado com esse novo exemplo e recebe o rótulo 4, indicando que foi criado na fase online e ainda não está associado a nenhuma das classes existentes. Como o número total de microgrupos criados na fase offline e armazenados na memória é definido pelo hiperparâmetro (`n_micro_clusters`) e deve ser constante, existem duas ações possíveis: se o microgrupo mais antigo não recebeu novos exemplos por um período prolongado, ele será removido. Caso contrário, se todos os microgrupos ainda estiverem dentro do limite de tempo, os dois microgrupos mais próximos serão combinados;

4. **Quantidade de microgrupos:** Quando um novo microgrupo precisa ser criado, o microgrupo mais antigo é removido ou dois microgrupos mais próximos são combinados, pois é necessário sempre manter os (`n_micro_clusters`) microgrupos na memória;

5. **Remoção de microgrupos por tempo:** Durante o processamento de novos dados online, se houver a necessidade de criar um novo microgrupo, o algoritmo verifica se algum microgrupo deve ser removido. O tempo atual é obtido e a idade de cada microgrupo é calculada subtraindo o tempo de última atualização (*CF1t*) do tempo atual. Se um microgrupo tiver uma idade muito alta e não recebe novos exemplos, ele é removido para liberar espaço para o novo microgrupo. Em outras palavras, se a idade de um microgrupo (calculada com base no tempo atual menos o tempo da última adicção de um exemplo ao microgrupo) exceder o limite (*delta*), o microgrupo é removido;

6. **Combinação de Microgrupos:** Quando há necessidade de redução de microgrupos e não é possível excluir um microgrupo antigo, os dois microgrupos mais próximos são fundidos. A distância entre os centróides dos microgrupos é calculada. Os dois microgrupos mais próximos são combinados, atualizando suas estatísticas (como somas e contagens) e recalculando o centróide do microgrupo combinado. Após a combinação, o algoritmo reatribui rótulos aos microgrupos combinados. Se os microgrupos combinados têm rótulos diferentes, o rótulo final é determinado com base no rótulo mais frequente entre os microgrupos combinados.

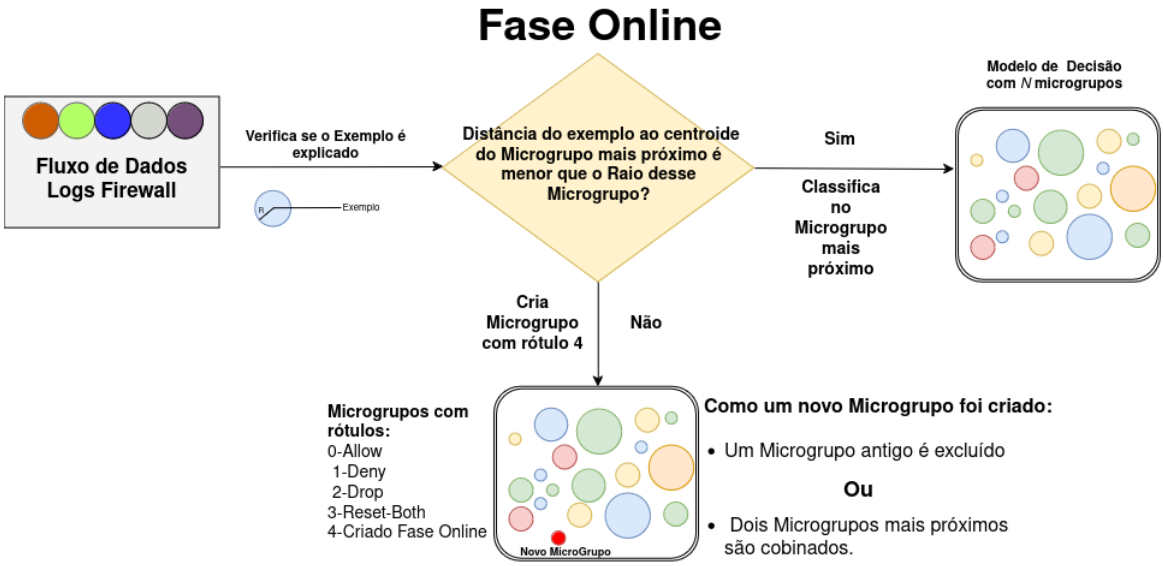


Figura 5 – Fase Online CluStream Adaptado.

Fonte: Elaborado pelo autor.

6.3.2.3 Fase de Macroagrupamento

Após o processamento online, o algoritmo realiza uma fase de macroagrupamento (Figura 6):

1. **Aplicação do K-Means:** Os microgrupos são agrupados em macrogrupos utilizando novamente o *K-Means*, com o número de macrogrupos definido por `n_macro_clusters`;

2. **Atribuição de rótulos aos macrogrupos:** Para cada macrogrupo, é atribuído um rótulo baseado na classe mais comum dos microgrupos que o compõem. Microgrupos com rótulo 4 (indicando que foram criados na fase online) são ignorados na contagem;

3. **Tratamento especial:** Se todos os microgrupos de um macrogrupo tiverem rótulo 4, o rótulo do macrogrupo é determinado com base no macrogrupo mais próximo que não tenha o rótulo 4.

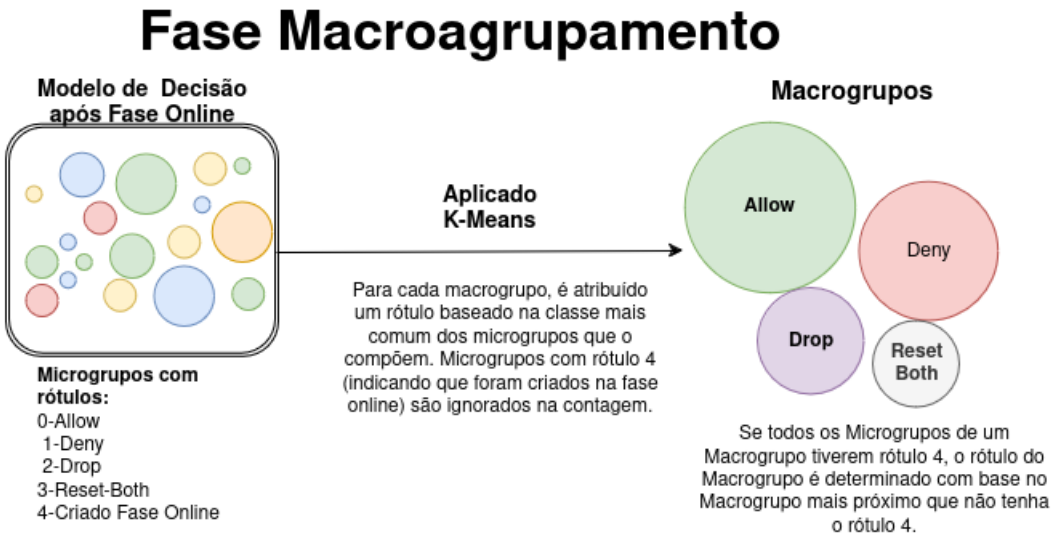


Figura 6 – Fase de Macroagrupamento CluStream Adaptado.

Fonte: Elaborado pelo autor.

6.3.2.4 Hiperparâmetros do Algoritmo

Os principais hiperparâmetros do algoritmo são:

- `n_micro_clusters`: Define o número de microgrupos a serem formados na fase offline e mantidos durante a fase online;
- `fat`: Fator de alargamento usado para determinar o raio de aceitação de novos exemplos em um microgrupo;
- `delta`: Limite de tempo utilizado para determinar quando um microgrupo deve ser removido com base na sua idade. Microgrupos são removidos somente quando é necessário criar novos microgrupos, pois o número total de microgrupos presentes na memória não deve ser alterado;

- **`n_macro_clusters`:** Define o número de macrogrupos a serem formados na fase de macroagrupamento. Os macrogrupos são grupos maiores que agrupam múltiplos microgrupos com o intuito de rotular uma das classes do atributo alvo Action: (*Allow, Deny, Drop e Reset-Both*).

6.3.2.5 Avaliação do Modelo

Após o macroagrupamento, o algoritmo avalia o modelo gerado, comparando as previsões do algoritmo na fase online com as classes reais *(Allow, Deny, Drop e Reset-Both).* Essa avaliação é feita utilizando a matriz de confusão e relatórios de classificação.

- **Matriz de Confusão:** Uma matriz que mostra o número de previsões corretas e incorretas para cada classe;

- **Relatório de Classificação:** Um relatório que inclui métricas para a avaliação do desempenho do modelo, incluindo precisão, recall, F1-score e outras métricas relevantes discutidas na metodologia empregada.

6.3.3 Principais diferenças - CluStream e CluStream Adaptado

O algoritmo CluStream original é um método de agrupamento incremental voltado para FCDs. Ele opera em duas fases: uma fase online, responsável por manter resumos estatísticos dos dados através de microgrupos, e uma fase offline, na qual esses microgrupos podem ser agrupados em macrogrupos usando algoritmos como o K-Means. Essa abordagem permite ao analista realizar análises retroativas e detectar padrões de comportamento sem a necessidade de armazenar os dados originais.

Entretanto, para possibilitar a classificação da ação recomendada em tráfego de rede (Allow, Deny, Drop e Reset-Both), foram realizadas adaptações específicas ao CluStream original. As principais diferenças e extensões realizadas são apresentadas a seguir:

- **Objetivo do algoritmo**: O CluStream original é utilizado para agrupamento não supervisionado, sem considerar rótulos de classe. Já a versão adaptada tem como foco a classificação utilizando dados rotulados na fase offline para prever a ação recomendada para cada sessão de tráfego de rede.

- **Rotulagem dos microgrupos na fase offline**: No CluStream original, os microgrupos não possuem rótulos. No adaptado, cada microgrupo é rotulado com a classe predominante dos exemplos que o compõem durante a fase offline, funcionando como base para a posterior classificação.

- **Criação de microgrupos com rótulo especial**: Na fase online do CluStream adaptado, quando um novo exemplo não se encaixa em nenhum microgrupo exis-

tente, um novo microgrupo é criado com rótulo 4, indicando que se trata de um dado de classe ainda desconhecida ou diferente do que foi aprendido na fase offline.

- **Uso da fase de macroagrupamento para classificação**: Apesar de ambos os algoritmos realizarem macroagrupamento utilizando K-Means sobre os microgrupos, no CluStream Adaptado essa fase é usada para gerar macrogrupos rotulados, os quais determinam se o tráfego de rede será permitido (Allow) ou negado (Deny, Drop e Reset-Both).

- **Classificação de novos exemplos**: O CluStream original não realiza nenhuma tarefa de classificação. No adaptado, novos exemplos são classificados com base no microgrupo mais próximo, considerando a menor distância ao centroide e o rótulo previamente atribuído a esse microgrupo na fase offline.

- **Tratamento de macrogrupos com rótulo desconhecido**: Se um macrogrupo for formado apenas por microgrupos com rótulo 4, o algoritmo atribui a ele o rótulo do macrogrupo mais próximo com classe conhecida, garantindo que todos os exemplos possam ser classificados.

Essas modificações transformam o CluStream original em um classificador incremental baseado em agrupamento rotulado. A estrutura de microgrupos é preservada, mas o modelo passa a incorporar a capacidade de tomar decisões com base nos rótulos aprendidos durante a fase offline. Na fase online, os exemplos que chegam são inicialmente atribuídos aos microgrupos mais próximos com base em similaridade, sem a necessidade de rótulos. Posteriormente, esses microgrupos são organizados em macrogrupos, os quais são rotulados com base nos microgrupos gerados durante a fase offline. Dessa forma, o modelo consegue rotular novos exemplos com base nesses macrogrupos. Além disso, mesmo os microgrupos criados dinamicamente na fase online são considerados na etapa de macroagrupamento, sendo incorporados a macrogrupos que representam uma das classes do problema, o que garante que o sistema continue classificando o tráfego de rede com base nos padrões previamente aprendidos e organizados ao longo do fluxo de dados.

A adaptação do CluStream possui uma estrutura em três fases: offline, online e macroagrupamento. Na fase offline, os microgrupos são criados e rotulados com base nos dados históricos, fornecendo um modelo inicial. Durante a fase online, o algoritmo atualiza continuamente os microgrupos, podendo gerar novos grupos com rótulo provisório “4” para representar padrões emergentes. Por fim, um macroagrupamento é realizado para consolidar os microgrupos em macrogrupos, permitindo a rotulação final das instâncias.

Para os experimentos apresentados neste trabalho, o macroagrupamento foi executado após a conclusão da fase online, com o objetivo de avaliar o comportamento do algoritmo na previsão da ação recomendada para o tráfego de rede e analisar as métricas de desempenho sobre todos os conjuntos de dados coletados (Experimentos 1, 2 e 3).

Essa abordagem não invalida o cenário em aplicações reais, pois a mesma estrutura pode ser ajustada para executar macroagrupamentos periódicos em mini-batches ao longo do fluxo. Dessa forma, o sistema mantém a capacidade de fornecer rótulos de maneira rápida e em tempo quase real, preservando as características de aprendizado incremental e adaptação contínua do algoritmo.

6.4 Algoritmo MINAS: Estrutura e Funcionamento

A aplicação de técnicas de DN em FCDs é desafiadora devido à constante evolução dos conceitos envolvidos (GAMA, 2010). Nesta seção, será apresentado o algoritmo MINAS (FARIA; GAMA; CARVALHO, 2013), um método de aprendizado multiclasse para FCDs que trata a DN como um problema de múltiplas classes, permitindo a identificação simultânea de várias classes novidade. Uma característica distintiva do MINAS é que ele não requer rótulos verdadeiros dos exemplos para atualização. Este algoritmo apresenta várias funcionalidades importantes: ele utiliza um modelo de decisão dinâmico que abrange o conhecimento atual do problema, lidando com múltiplas classes; emprega um conjunto representativo de exemplos que não são cobertos pelo modelo atual, facilitando a aprendizagem de novos conceitos ou a extensão dos existentes e integra continuamente o conhecimento adquirido, incorporando tanto as classes aprendidas na fase offline quanto os padrões de novidade detectados de maneira online. Além disso, o MINAS distingue entre exemplos isolados, que são considerados outliers ou ruídos, e padrões de novidade, que devem formar um grupo coeso e representativo (FARIA et al., 2016).

O algoritmo MINAS é dividido em duas fases principais: a fase offline e a fase online. A fase offline, realizada uma única vez, é uma etapa supervisionada onde o algoritmo recebe um conjunto de dados rotulados para desenvolver um modelo de decisão inicial. Cada classe no problema é representada por um conjunto de microgrupos, que são estruturas compactas de representação de dados. Esses microgrupos são formados utilizando algoritmos de agrupamento como o K-Means (LLOYD, 1982; MACQUEEN, 1967) ou o CluStream (AGGARWAL et al., 2003). O K-Means minimiza a distância quadrada dos exemplos até seus centróides, enquanto o CluStream, projetado para lidar com FCDs, trata outliers e evita grupos vazios. Na Figura 7 é representada a fase offline do algoritmo MINAS (FARIA et al., 2016).

Na fase online, o MINAS classifica novos exemplos não rotulados com base no modelo de decisão construído na fase offline. Exemplos não explicados pelo modelo são marcados como desconhecidos e armazenados em uma memória temporária. Esses exemplos são periodicamente agrupados em novos microgrupos, que são validados para verificar sua coesão e representatividade. Microgrupos válidos podem indicar novos padrões ou extensões dos conceitos já aprendidos e são incorporados ao modelo de decisão. Na Figura 8 é representada a fase online do algoritmo MINAS.

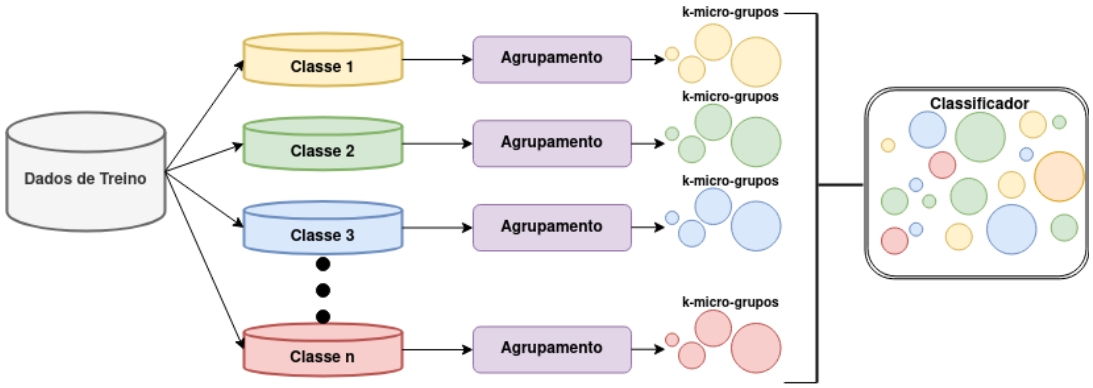


Figura 7 – Fase offline do Algoritmo MINAS.

Fonte: Adaptado de (FARIA et al., 2016)

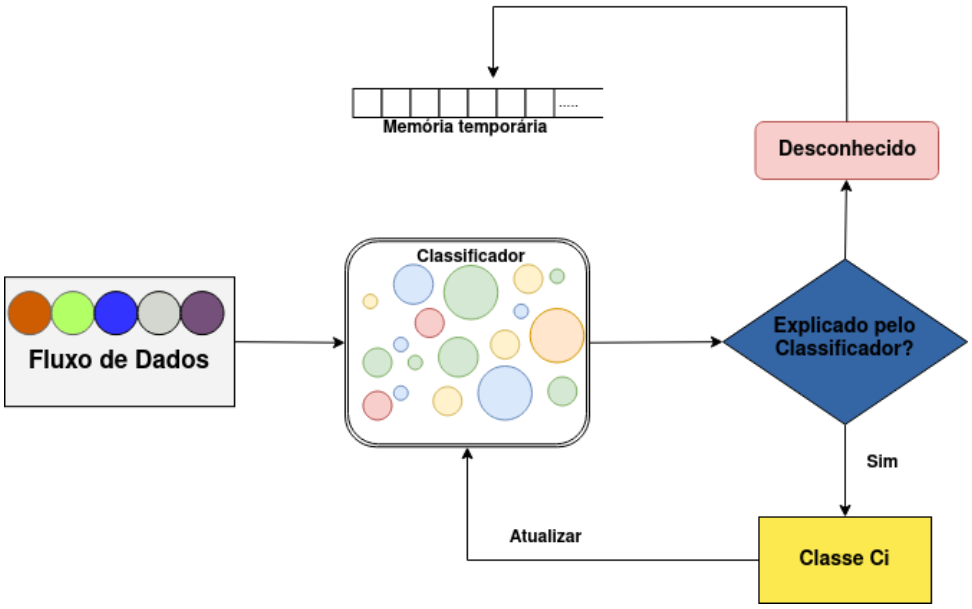


Figura 8 – Fase online do Algoritmo MINAS.

Fonte: Adaptado de (FARIA et al., 2016)

Além da atualização do modelo de decisão com novos microgrupos, o MINAS possui mecanismos para lidar com microgrupos obsoletos e limpeza da memória temporária. Exemplos desconhecidos que não formaram novos microgrupos são eliminados.

O modelo de decisão é gerado durante a fase offline, onde ele recebe um conjunto de dados rotulados e aplica um algoritmo de agrupamento (K-Means ou CluStream) a cada classe do problema, formando microgrupos que a representarão. Esses microgrupos são caracterizados por três componentes de sumarização: número de exemplos (N), soma linear (LS) e soma quadrada (SS), permitindo calcular o centróide e o raio do microgrupo. Além disso, cada microgrupo possui um marcador de tempo (t) que indica o momento do último exemplo adicionado ao microgrupo. O K-Means exige que o número de microgrupos seja definido previamente, enquanto o CluStream recomenda um

número significativamente maior de microgrupos do que o de grupos presentes nos dados. Embora um número excessivo de microgrupos possa resultar em ineficiências de busca e armazenamento, um número maior de microgrupos pode, por outro lado, proporcionar uma representação mais precisa dos dados, especialmente quando estes apresentam diferentes formatos (FARIA et al., 2016).

Na fase online, cada novo exemplo é classificado com base na distância ao centróide do microgrupo mais próximo presente no modelo de decisão. Se a distância for menor que o raio do microgrupo, o exemplo é classificado e o microgrupo é atualizado. Caso contrário, o exemplo é marcado como desconhecido e armazenado em uma memória temporária para análises futuras. Esses exemplos podem indicar ruídos, mudanças em conceitos ou novos padrões. Além disso, microgrupos do cenário atual são movidos para uma memória *sleep* se permanecerem sem receber exemplos por um determinado período de tempo, pois já não estão mais contribuindo para a classificação de novos exemplos. O tamanho da janela de dados é definido pelo hiperparâmetro *windowsize* (PAIVA, 2014).

O MINAS opera com um único modelo de decisão, composto por classes aprendidas durante a fase offline e padrões novidade ou extensões aprendidos online, todos representados por microgrupos. A classificação de novos exemplos considera tanto as classes iniciais quanto os padrões aprendidos online (FARIA et al., 2016).

6.4.1 Detecção de Extensões, Padrões Novidade, Recorrência e Esquecimento do Passado

Quando o número mínimo de exemplos desconhecidos na memória temporária é atingido, o algoritmo MINAS realiza um procedimento não supervisionado de DN. Esse procedimento aplica algoritmos de agrupamento, como CluStream ou K-Means, aos exemplos na memória temporária, gerando novos microgrupos que são avaliados para determinar sua validade (FARIA et al., 2016). Na Figura 9 é mostrado o funcionamento desse procedimento.

Para um microgrupo ser considerado válido, sua largura de silhueta deve ser maior que 0, conforme a Equação 5, onde b é a distância Euclidiana entre o centróide do novo microgrupo e o centróide do microgrupo mais próximo, e a é o desvio padrão das distâncias Euclidianas entre os elementos do novo microgrupo e seu centróide. Como resultado, a largura da silhueta apresenta um valor entre 0 e 1 (VENDRAMIN; CAMPELLO; HRUSCHKA, 2010).

$$\text{Silhueta} = \frac{b - a}{max(b, a)} \tag{5}$$

Além disso, o microgrupo deve ser representativo, contendo um número mínimo de exemplos. Para integrá-lo ao modelo de decisão, é necessário determinar se ele é uma extensão de um conceito já existente ou um Padrão Novidade. Essa determinação é feita

medindo a distância entre o centróide do novo microgrupo e o centróide do microgrupo mais próximo no modelo atual. Se essa distância for inferior a um limiar definido pelo usuário, o microgrupo é classificado como uma extensão do conceito existente. Caso contrário, é considerado um Padrão Novidade e recebe um novo rótulo sequencial (FARIA et al., 2016).

Microgrupos inválidos permanecem na memória temporária para futuras operações de agrupamento, mas devem ser removidos se não forem utilizados após um determinado período de tempo.

O MINAS distingue vários Padrões Novidade ao longo do tempo. Um Padrão Novidade pode consistir em um ou mais microgrupos e não está diretamente vinculado às classes do problema. Quando uma nova classe surge no FCDs, ela pode ser reconhecida por um ou mais Padrões Novidade, exigindo a avaliação de um especialista para determinar se representam uma nova classe ou se estão relacionados a uma mesma classe (FARIA et al., 2016).

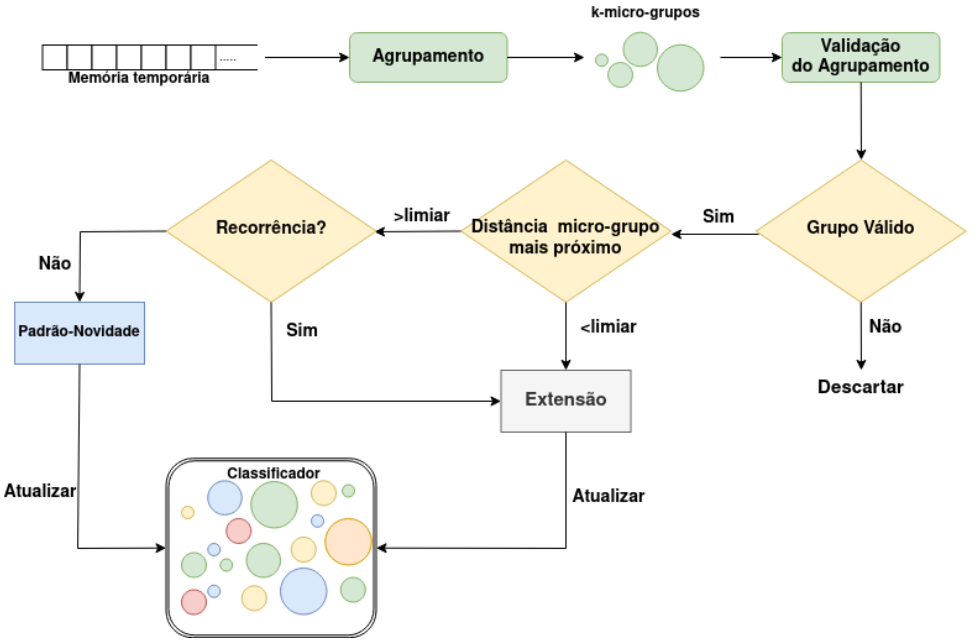


Figura 9 – Procedimento de DN do algoritmo MINAS.

Fonte: Adaptado de (FARIA et al., 2016)

O modelo de decisão é atualizado classificando exemplos com microgrupos existentes e incorporando novos microgrupos válidos, enquanto descarta aqueles antigos que não contribuem mais para a atividade recente do FCDs. Além da atualização do modelo de decisão por meio da inserção ou atualização de microgrupos, é essencial descartar microgrupos antigos que possam não contribuir mais para a atividade recente do FCDs. Dessa forma, cada microgrupo armazena uma componente que representa o marcador de tempo associado ao último exemplo classificado nesse microgrupo. São transferidos para

uma memória *sleep* os microgrupos que não recebem novos exemplos por um período de tempo. Assim é possível que o modelo esqueça microgrupos que se tornaram obsoletos.

Ao identificar um novo microgrupo válido, se o modelo de decisão atual não conseguir classifica-lo como uma extensão, o MINAS verificará na memória *sleep* se ele deve ser tratado como uma extensão ou um Padrão Novidade. Se o centróide do novo microgrupo estiver suficientemente próximo ao centróide de um microgrupo na memória *sleep*, ele é considerado uma extensão de um conceito previamente conhecido e o microgrupo adormecido é reintegrado ao modelo atual, indicando uma recorrência. Essa proximidade é determinada por um hiperparâmetro do algoritmo definido pelo usuário. Caso contrário, ele é classificado como um Padrão Novidade (FARIA et al., 2016). O hiperparâmetro (limiar) utilizado para verificar se o microgrupo válido é uma extensão de conceito, um Padrão de Novidade ou uma recorrência é o mesmo. O que diferencia cada caso é a distância do centróide do novo microgrupo em relação ao microgrupo mais próximo, seja no modelo de decisão ou na memória *sleep*.

Microgrupos que não alcançam critérios mínimos de coesão ou número de exemplos são excluídos. Exemplos que não são agrupados de forma válida permanecem na memória temporária para análises futuras, mas devem ser eliminados após um período determinado (PAIVA, 2014).

Ao incluir um novo exemplo na memória temporária, um marcador de tempo é registrado. Exemplos com marcadores antigos são removidos de acordo com a frequência determinada pelo hiperparâmetro *windowsize*, definido pelo usuário. Ao final de cada janela de dados, o MINAS elimina exemplos desatualizados que participaram de pelo menos um processo de agrupamento, mas cujos microgrupos não atenderam aos critérios de validação. Isso os torna suscetíveis a serem considerados ruídos ou outliers (PAIVA, 2014).

6.4.2 Adaptação do MINAS

Na fase offline, durante a criação do modelo inicial, cada classe do problema é representada por um conjunto de microgrupos. Assim, não é necessário adaptar o algoritmo, pois um conjunto de microgrupos será criado para representar cada classe do atributo alvo *action* (*Allow, Deny, Drop e Reset-Both*).

Na fase online, cada novo exemplo é classificado com base na distância ao centróide do microgrupo mais próximo presente no modelo de decisão gerado na fase offline, permitindo assim, determinar a qual classe ele pertence. Caso a distância do exemplo seja maior que o raio do microgrupo, o exemplo é marcado como desconhecido e armazenado em uma memória temporária para análises futuras.

Os microgrupos que se tornam coesos e representativos na memória temporária são integrados ao modelo. Se a distância entre um microgrupo coeso da memória temporária

e o microgrupo mais próximo no modelo de decisão for inferior a um limiar definido pelo usuário, o microgrupo da memória temporária é classificado como uma extensão de um conceito existente, e um novo microgrupo é adicionado para representar uma das classes do problema (*Allow, Deny, Drop ou Reset-Both*). Caso a distância seja superior ao limiar, isso indicaria o surgimento de uma nova classe.

Como as classes são fixas e novas classes não surgem durante o fluxo de dados, a estratégia é definir um limiar elevado. Isso garante que, quando um microgrupo é removido da memória temporária por ter se tornado coeso, ele seja integrado ao modelo como uma extensão de um conceito conhecido, em vez de ser tratado como um padrão novidade que sugeriria o surgimento de uma nova classe.

Devido às características do algoritmo, não é necessário alterar sua implementação. Apenas ajustes nos hiperparâmetros são necessários para adaptá-lo à criação de extensões de conceitos conhecidos, quando necessário, em vez de identificar padrões novidade que poderiam representar o surgimento de novas classes. Dessa forma, novos padrões e conceitos devem ser identificados, mas as ações a serem tomadas continuam sendo restritas a permitir ou negar o tráfego;

6.4.2.1 Avaliação do Modelo

Faria et al. (2015) e Faria et al. (2016) assumem que a matriz de confusão gerada pelo algoritmo MINAS não é quadrada, já que o número de colunas aumenta sempre que um novo Padrão Novidade é detectado. Cada linha da matriz representa uma classe do problema, enquanto cada coluna representa uma das classes previstas pelo algoritmo. Adicionalmente, a última coluna da matriz é dedicada aos exemplos classificados como desconhecidos. Estes são exemplos que permanecem na memória temporária e podem, no futuro, formar novos padrões que representem alguma das classes do problema, caso se agrupem de maneira coesa e representativa.

Como mencionado anteriormente, o objetivo do estudo é que o algoritmo MINAS classifique exemplos de *logs* de tráfego de rede em uma das classes existentes, sem identificar novos padrões novidade que representem novas classes do problema. Isso ocorre porque o número de classes é fixo, e o algoritmo deve atribuir cada exemplo a uma classe já definida, seja como uma extensão de conceito ou porque o exemplo se encaixa no padrão dos microgrupos que representam a classe. No entanto, em alguns casos, o número de exemplos classificados como desconhecidos pode ser maior. Isso pode acontecer quando o fluxo de dados está passando por uma mudança de conceito, o que pode temporariamente aumentar o número de exemplos desconhecidos até que esses dados se consolidem como um novo padrão dentro das classes existentes.

Durante o fluxo, chega um momento em que os exemplos desconhecidos precisam ser tratados (Fase de avaliação do desempenho do modelo). Se o tráfego de rede não é

reconhecido e não corresponde a nenhum padrão existente, é necessário que em algum momento ele seja bloqueado. Essa necessidade de bloquear dados desconhecidos surge da premissa de que, se o tráfego não é conhecido, ele precisa ser descartado para manter a segurança e a integridade da rede.

Para avaliar o desempenho do algoritmo, a matriz de confusão gerada inclui uma coluna adicional para exemplos desconhecidos (Unk). No entanto, para calcular métricas como precisão, recall e F1-score, é preciso que a matriz de confusão seja quadrada (ou seja, tenha o mesmo número de linhas e colunas). Para ajustar a matriz de confusão para a avaliação dessas métricas, os valores presentes na coluna "Unk" são realocados para a coluna correspondente à ação "Drop" classe (C2). Isso implica que todos os exemplos desconhecidos são tratados com a ação de bloqueio do tipo "C2-Drop" (descartados), resultando em uma matriz de confusão quadrada que permite a avaliação adequada das métricas de desempenho. Essa abordagem também antecipa o tratamento necessário para esses dados, bloqueando-os na avaliação, uma vez que o tráfego desconhecido precisa ser descartado. Na Tabela 4 é ilustrada a conversão da matriz de confusão gerada pelo MINAS em uma matriz quadrada.

Tabela 4 – Conversão de Exemplos Desconhecidos (Unk) para Drop (C2) na Matriz de Confusão do MINAS

	C 0	C 1	C 2	C 3	Unk
C 0	308351	153789	1000	1	15
C 1	1426	188043	16000	0	10
C 2	25	8867	54430	0	5
C 3	586	6	0	1	2

	C 0	C 1	C 2	C 3
C 0	308351	153789	1015	1
C 1	1426	188043	16010	0
C 2	25	8867	54435	0
C 3	586	6	2	1

Fonte: Elaborado pelo autor.

Com base nos resultados dessa matriz de confusão ajustada, é gerado um relatório que inclui métricas para a avaliação do desempenho do modelo, incluindo precisão, recall, F1-score e outras métricas relevantes discutidas na metodologia empregada.

É importante destacar que os dados desconhecidos não são imediatamente descartados como "Drop". Antes de tomar essa ação, esses exemplos continuam sendo monitorados, pois no futuro eles podem formar microgrupos coesos e representativos de um padrão que se estenda a um conceito já conhecido. O MINAS, por sua vez, é capaz de identificar e rotular novos exemplos que se ajustem a esse padrão emergente. Dessa forma, antes da ação "Drop" ser executada para dados desconhecidos (Fase de avaliação do desempenho do modelo), eles são mantidos por um período, permitindo a análise de novos exemplos e possibilitando o surgimento de extensões de conceitos.

A ação de bloqueio do tipo "Drop" foi escolhida em detrimento de "Deny" e "Reset-Both", pois estas são mais específicas e utilizadas em situações particulares de bloqueio. Já a ação "Drop" pode ser aplicada a qualquer exemplo, tornando-se a opção mais ade-

quada para lidar com exemplos marcados como desconhecidos. Por outro lado, a ação "Allow" deve ser evitada, pois permite o tráfego de rede, o que não é aceitável para tráfego desconhecido, especialmente por questões de segurança.

6.5 Considerações Finais

Neste capítulo, foram discutidos os métodos e os diversos aspectos do algoritmo MINAS, incluindo uma visão detalhada de suas fases offline e online. Também foram abordados os procedimentos de Detecção de Extensões, Detecção de Padrões Novidade e as abordagens para lidar com recorrências. Adicionalmente, foram apresentados os métodos e o funcionamento do algoritmo incremental para agrupamento em FCDs, conhecido como CluStream. Além da versão original do algoritmo, foi discutida a adaptação realizada para permitir sua aplicação na previsão da ação recomendada para o tráfego de rede em *firewalls*.

O próximo capítulo descreve os experimentos conduzidos com os algoritmos CluStream Adaptado e MINAS, voltados para FCDs, com o objetivo de prever as ações recomendadas para o tráfego de rede em *firewalls*. Além disso, são discutidas as comparações entre os experimentos deste estudo e os desafios que dificultam a comparação com outras pesquisas relacionadas.

Capítulo 7

Experimentos

No capítulo anterior, foram detalhados os métodos e algoritmos utilizados na pesquisa para classificar as ações recomendadas para o tráfego de rede em FCDs, empregando *logs* de *firewalls*. Este capítulo se concentra nos experimentos realizados para validar esses métodos. Serão apresentados os detalhes sobre a base de dados utilizada, a metodologia dos experimentos e a discussão dos resultados obtidos.

7.1 Base de Dados

Conforme citado nos capítulos anteriores, os dados experimentais foram obtidos de uma instituição de ensino com mais de 3500 servidores públicos federais e mais de 15.000 alunos matriculados em cursos de graduação, mestrado e doutorado. Esses *logs* foram capturados pelo *firewall* Palo Alto, modelo 3260. Cada *log* representa uma única instância de comunicação entre dois dispositivos na rede, denominada sessão de tráfego de rede. Nesse conjunto de dados, o atributo "action" será a classe a ser predita, podendo ser classificado em quatro rótulos distintos (Palo Alto Networks, 2024c):

- **Allow:** Ação que permite o tráfego de rede;
- **Drop:** Usada principalmente como uma forma furtiva de descartar o tráfego. O *firewall* simplesmente descartará quaisquer pacotes associados a uma conexão indesejada, não permitindo que o cliente ou servidor saiba que os pacotes estão sendo descartados;
- **Reset-both:** Envia uma redefinição de TCP para dispositivos do lado do cliente e do lado do servidor. Essa redefinição implica em encerrar abruptamente a sessão,

liberando imediatamente os recursos alocados para a conexão e apagando todas as outras informações relacionadas a ela;

- **Deny:** Bloqueia o tráfego de rede e impõe a ação de negação recomendada pela aplicação especificada na regra. A recomendação, por exemplo, pode ser bloquear o tráfego de rede e notificar o remetente que a sessão foi encerrada (enviando um pacote *TCP RESET* para que ele possa fechá-la localmente de forma adequada). Se não houver uma aplicação definida na regra ou uma recomendação específica da aplicação, os pacotes serão descartados silenciosamente.

É importante destacar que essas ações são tomadas com base em regras fixas, previamente configuradas no *firewall* implementado pela equipe de segurança da instituição de ensino. Tais regras só são alteradas quando há necessidade de ajustes nas políticas de segurança ou quando se identifica a necessidade de correções ou mudanças específicas. Vale ressaltar que essas regras não possuem caráter dinâmico.

Dessa forma, os algoritmos e modelos são treinados com os dados gerados pelos *logs* das ações realizadas com base nessas regras fixas do *firewall* e, posteriormente, passam a prever ações com base nessas informações. A partir disso, avalia-se a capacidade dos algoritmos e modelos em prever o tráfego de rede com precisão, utilizando os dados disponíveis.

Além disso, caso alguns modelos apresentem desempenho satisfatório nas métricas avaliadas para a previsão das ações corretas, é possível identificar discrepâncias entre as ações previstas pelos modelos e as implementadas pelo *firewall*, o que pode indicar possíveis erros na configuração das regras.

7.1.1 Conjunto de Dados

Os *logs* do *primeiro conjunto de dados* utilizados nos experimentos MINAS e CluStream Adaptado foram coletados ao longo de 17 dias consecutivos, em diferentes horários, com o objetivo de capturar a maior variação possível nos padrões de tráfego de rede. Em média, cada arquivo de *log* coletado continha aproximadamente 55 mil sessões de tráfego de rede, resultando em um conjunto de dados com um total de 1.002.160 exemplos. Dentre esses exemplos, 100.216 foram utilizados na fase offline para o treinamento do modelo, enquanto 901.944 foram empregados na fase online, simulando o cenário de FCDs. As quatro classes que compõem o problema de classificação são altamente desbalanceadas, conforme ilustrado na Tabela 5.

Um novo conjunto de dados foi criado 15 dias após a última coleta do *primeiro conjunto de dados*. Os *logs* foram capturados ao longo de 9 dias consecutivos, mais uma vez em diferentes horários, com o objetivo de registrar a maior variação possível nos padrões de tráfego de rede. Essa coleta resultou em um conjunto de dados contendo 515.575 exem-

Tabela 5 – Distribuição das Classes nas Fases Offline e Online - Primeiro conjunto de dados

Classe	**Fase Offline (10% do total)**	**Fase Online (90% do total)**
C0 - Allow	53,64% (53.753 exemplos)	53,57% (483.131 exemplos)
C1 - Deny	39,14% (39.223 exemplos)	39,28% (354.300 exemplos)
C2 - Drop	7,11% (7.122 exemplos)	7,04% (63.527 exemplos)
C3 - Reset-Both	0,12% (118 exemplos)	0,11% (986 exemplos)

Fonte: Elaborado pelo autor.

plos. O objetivo desse *segundo conjunto de dados* foi simular um novo fluxo (Fase Online 2) do FCDs. Os dados coletados foram utilizados no segundo e terceiro experimentos, aplicando os algoritmos MINAS e CluStream Adaptado. Assim como no primeiro conjunto, a distribuição das quatro classes do problema de classificação mostrou-se altamente desbalanceada, conforme ilustrado na Tabela 6.

Tabela 6 – Distribuição das Classes nas Fase Online 2 - Segundo conjunto de dados

Classe	**Fase Online 2**
C0 - Allow	54,60% (281.468 exemplos)
C1 - Deny	38,83% (200.213 exemplos)
C2 - Drop	6,31% (32.538 exemplos)
C3 - Reset-Both	0,26% (1.356 exemplos)

Fonte: Elaborado pelo autor.

Experimentos foram conduzidos para avaliar o desempenho do MINAS e do CluStream Adaptado em FCDs, utilizando diferentes combinações de conjuntos de dados.

- **Experimento 1:** O *primeiro conjunto de dados* foi utilizado para treinar e testar o modelo em um FCDs simulado. Para esse experimento, 10% dos dados foram alocados para a fase offline, destinada ao treinamento do modelo, enquanto 90% foram utilizados na fase online para simular o fluxo de dados. Esse experimento serve como uma linha de base para avaliar como o modelo lida com dados em fluxo contínuo e para medir seu desempenho inicial.

- **Experimento 2:** O objetivo é avaliar o desempenho do modelo quando exposto a um novo fluxo de dados coletado em um momento diferente daquele usado no Experimento 1. Ao introduzir um novo fluxo de dados (Segundo Conjunto de Dados), esse experimento investiga a capacidade do modelo de generalizar para dados com características potencialmente diferentes. Os mesmos 10% do *primeiro conjunto de dados* foram alocados para a fase offline, destinada ao treinamento do modelo.

- **Experimento 3:** O objetivo é analisar o desempenho do modelo em dois fluxos consecutivos (Primeiro e Segundo Conjunto de Dados), aproveitando o ajuste incremental do modelo a partir do primeiro fluxo para prever instâncias do segundo fluxo. Esse experimento testa a capacidade do modelo de se adaptar ao longo do tempo à medida que novos dados chegam, simulando um cenário mais realista e contínuo, no qual o modelo deve aprender e evoluir conforme mais dados se tornam disponíveis, além de esquecer aqueles que não contribuem mais para o processo de classificação.

7.1.2 Preparação dos Dados

O *primeiro conjunto de dados* coletado foi particionado em duas etapas: fase offline e fase online, sendo 10% destinado à fase offline e os 90% restantes à fase online. Como resultado, foram gerados dois arquivos finais, correspondentes às fases Offline e Online, respectivamente. Em seguida, foi submetido aos processos de codificação e normalização.

Para o *segundo conjunto de dados* coletado, o processo foi ligeiramente diferente, pois apenas uma Fase Online foi criada. Assim, foi gerado um único conjunto de dados, representando a Fase Online 2. Esse conjunto foi devidamente codificado e normalizado, representando um novo fluxo de dados para o FCDs, simulando outro cenário de tráfego de rede.

Realizou-se a codificação dos atributos categóricos, conforme listado a seguir:

- *Source address* (Endereço IP de origem)
- *Destination address* (Endereço IP de destino)
- *NAT Source IP* (IP de origem NAT)
- *NAT Destination IP* (IP de destino NAT)
- *Application* (Aplicação)
- *Source Zone* (Zona de origem)
- *Destination Zone* (Zona de destino)
- *IP Protocol* (Protocolo IP)
- *Category* (Categoria)
- *Source Country* (País de origem)
- *Destination Country* (País de destino)
- *Subcategory of app* (Subcategoria da aplicação)

- *Category of app* (Categoria da aplicação)
- *Technology of app* (Tecnologia da aplicação)
- *Container of app* (Container da aplicação)

O atributo alvo, *Action*, que indica a ação recomendada para o tráfego de rede ('Allow', 'Deny', 'Drop' e 'Reset-Both'), também foi codificado. Nesse processo, cada categoria de ação foi transformada em um valor numérico, conforme descrito a seguir:

- 0 - *Allow*
- 1 - *Deny*
- 2 - *Drop*
- 3 - *Reset-Both*

Após a codificação dos atributos categóricos, os dados foram normalizados com uma faixa de normalização de $(-1, 1)$ em mini-batches de 1000 instâncias, permitindo seu uso em modelos de aprendizado contínuo. A normalização é uma etapa que garante que todos os atributos tenham a mesma escala, o que pode melhorar o desempenho dos algoritmos de AM. A coluna *ID* e o atributo alvo *Action* foram excluídos do processo de normalização para preservar sua integridade sem alterar seus significados originais.

7.1.3 Atributos Utilizados

No *firewall* Palo Alto, modelo 3260, cada *log* representa uma única instância de comunicação entre dois dispositivos na rede, chamada de sessão de tráfego de rede. Cada sessão é identificada por uma combinação única de parâmetros, totalizando 113 atributos. No entanto, muitos desses atributos não contribuem de maneira significativa para o processo de classificação. É o caso de identificadores como IDs, números de série, nomes de host e endereços MAC (*Media Access Control*), que têm a função exclusiva de identificar dispositivos de rede, mas que, por si só, não carregam informações relevantes para a tarefa de classificação. Portanto, todos os atributos considerados irrelevantes foram removidos, resultando em um conjunto reduzido de 27 atributos, incluindo o atributo alvo *Action*. Estes atributos são descritos a seguir (Palo Alto Networks, 2024a):

1. **Source Address**: Endereço IP da máquina de origem que iniciou a comunicação na rede. Representa o dispositivo que está enviando dados;
2. **Destination Address**: Endereço IP do destino para o qual o tráfego está sendo enviado. Este é o dispositivo que recebe os dados;

3. **NAT Source IP**: Endereço IP da origem após a aplicação de *Network Address Translation* (NAT). O NAT pode ser usado para converter endereços IP privados em públicos;

4. **NAT Destination IP**: Endereço IP do destino após a aplicação de NAT. Pode ser usado para mapear o tráfego que está entrando para um endereço IP interno específico;

5. **Application**: Nome ou identificação da aplicação que gerou o tráfego. O firewall Palo Alto é capaz de identificar e classificar aplicações com base em padrões de tráfego, independentemente da porta;

6. **Source Zone**: Zona de segurança na qual o tráfego de origem está localizado. As zonas são usadas para segmentar a rede e aplicar políticas de segurança específicas;

7. **Destination Zone**: Zona de segurança onde o tráfego de destino está localizado. O firewall utiliza as zonas para determinar como o tráfego deve ser tratado com base nas políticas configuradas;

8. **Source Port**: Número da porta TCP/UDP usada pelo dispositivo de origem para enviar os dados. Indica o serviço ou aplicação de origem;

9. **Destination Port**: Número da porta TCP/UDP usada pelo dispositivo de destino para receber os dados. Determina o serviço ou aplicação no dispositivo de destino;

10. **NAT Source Port**: Porta de origem após a aplicação de NAT. Usada para identificar o tráfego após o processo de tradução de endereço de rede;

11. **NAT Destination Port**: Porta de destino após a aplicação de NAT. Usada para mapear o tráfego de entrada para a porta de um host interno;

12. **IP Protocol**: Protocolo de comunicação IP utilizado, como TCP, UDP, ICMP, etc;

13. **Bytes**: Total de bytes transmitidos durante a sessão. Este valor inclui todos os dados enviados e recebidos;

14. **Bytes Sent**: Total de bytes enviados da origem para o destino durante a sessão;

15. **Bytes Received**: Total de bytes recebidos pela origem do destino durante a sessão;

16. **Packets**: Total de pacotes de dados transmitidos durante a sessão. Um pacote é uma unidade de dados transmitida pela rede;

17. **Elapsed Time (sec)**: Duração da sessão em segundos. Refere-se ao tempo total em que a conexão esteve ativa;

18. **Category**: Categoria da aplicação ou do tráfego identificado, indica a categoria da URL associada à sessão de tráfego de rede, quando aplicável. A categoria pode ser usada para identificar e classificar o tipo de tráfego, como “Social Media”, “Business”, “Media”, etc;

19. **Source Country**: País de origem do tráfego, determinado com base no endereço IP de origem;

20. **Destination Country**: País de destino do tráfego, determinado com base no endereço IP de destino;

21. **Packets Sent**: Total de pacotes de dados enviados do dispositivo de origem para o dispositivo de destino durante a sessão;

22. **Packets Received**: Total de pacotes de dados recebidos pelo dispositivo de origem do dispositivo de destino durante a sessão;

23. **Category of App**: Categoria geral da aplicação identificada pelo firewall. Refere-se à classificação de aplicações com base em suas funções ou tipos, como business-systems, collaboration, media, general-internet, networking, e saas. Esta categorização ajuda a gerenciar e aplicar políticas de segurança;

24. **Subcategory of App**: Subcategoria da aplicação identificada, fornecendo uma classificação mais granular dentro da “Categoria da aplicação”;

25. **Technology of App**: Refere-se à tecnologia de aplicação especificada nas propriedades de configuração da aplicação. As categorias incluem: *Browser-Based, Client-Server, Network-Protocol e Peer-to-Peer*;

26. **Container of App**: Refere-se a um “container” ou grupo de aplicação, agrupando múltiplas aplicações em uma unidade (O aplicativo pai de um aplicativo). Esse atributo identifica a aplicação que hospeda ou gerencia outras aplicações dentro de sua infraestrutura, facilitando a visualização e o controle do tráfego associado a essas aplicações;

27. **Action**: Ação tomada pelo firewall em relação à sessão de tráfego (Allow, Deny, Drop e Reset-Both).

7.2 Combinações de Atributos

Durante os experimentos, foram testadas diversas combinações de atributos utilizando o algoritmo MINAS e o *primeiro conjunto de dados*, com o objetivo de avaliar o desempenho na previsão da ação recomendada para o tráfego de rede. No entanto, nem todas

as combinações geraram resultados satisfatórios. Ao todo, 29 combinações de atributos foram avaliadas, das quais 18 não alcançaram o desempenho esperado e, portanto, foram descartadas dos experimentos finais. Abaixo estão detalhadas as 11 combinações de atributos que apresentaram um melhor desempenho:

- **Dataset - Todos os Atributos Importantes**: Este conjunto de dados contém todos os 26 atributos considerados relevantes para os experimentos sem contar o atributo alvo *Action*:
  - Source address
  - Destination address
  - NAT Source IP
  - NAT Destination IP
  - Application
  - Source Zone
  - Destination Zone
  - Source Port
  - Destination Port
  - NAT Source Port
  - NAT Destination Port
  - IP Protocol
  - Bytes
  - Bytes Sent
  - Bytes Received
  - Packets
  - Elapsed Time (sec)
  - Category
  - Source Country
  - Destination Country
  - Packets Sent
  - Packets Received
  - Subcategory of app
  - Category of app
  - Technology of app

  - Container of app

- **Dataset - Todos os Atributos Importantes Sem IPs**: Este conjunto exclui os atributos relacionados a endereços IP, mas mantém outros atributos importantes:
  - Application
  - Source Zone
  - Destination Zone
  - Source Port
  - Destination Port
  - NAT Source Port
  - NAT Destination Port
  - IP Protocol
  - Bytes
  - Bytes Sent
  - Bytes Received
  - Packets
  - Elapsed Time (sec)
  - Category
  - Source Country
  - Destination Country
  - Packets Sent
  - Packets Received
  - Subcategory of app
  - Category of app
  - Technology of app
  - Container of app
- **Dataset - Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT**: Este conjunto de dados exclui os atributos relacionados a endereços IP, bem como aqueles associados ao processo de Tradução de Endereços de Rede (NAT - *Network Address Translation*):
  - Application
  - Source Zone

  - Destination Zone
  - Source Port
  - Destination Port
  - IP Protocol
  - Bytes
  - Bytes Sent
  - Bytes Received
  - Packets
  - Elapsed Time (sec)
  - Category
  - Source Country
  - Destination Country
  - Packets Sent
  - Packets Received
  - Subcategory of app
  - Category of app
  - Technology of app
  - Container of app

- **Dataset - Todos os Atributos Importantes Melhores Correlações**: Este conjunto de dados, que continha todos os 26 atributos considerados relevantes para os experimentos, excluindo o atributo alvo *Action*, passou por uma análise de correlação entre os atributos. A partir dessa análise, foram selecionados os atributos com as maiores correlações com o atributo alvo. O conjunto de dados resultante é composto pelos seguintes atributos:

  - Destination address
  - NAT Source IP
  - NAT Destination IP
  - Application
  - Source Zone
  - Destination Zone
  - Source Port
  - Destination Port

  - NAT Source Port
  - IP Protocol
  - Category
  - Destination Country
  - Subcategory of app
  - Category of app
  - Technology of app
  - Container of app

- **Dataset - Todos os Atributos Importantes Melhores Correlações Positivas**: Neste conjunto de dados, foram selecionados os atributos com as maiores correlações positivas em relação ao atributo alvo. Os atributos incluídos neste conjunto de dados são:
  - Application
  - Destination Port
  - IP Protocol
  - Subcategory of app
  - Category of app
  - Technology of app
  - Container of app

- **Dataset - Todos os Atributos Importantes Sem IPs e com Combinação**: Neste conjunto de dados, foi realizada uma seleção reduzida de atributos entre todos os atributos importantes, além de excluir aqueles que contêm IPs. Os atributos incluídos neste conjunto de dados são:
  - Application
  - Source Zone
  - Destination Zone
  - Source Port
  - Destination Port
  - NAT Source Port
  - NAT Destination Port
  - IP Protocol
  - Category

  - Source Country
  - Destination Country
  - Subcategory of app
  - Category of app
  - Technology of app
  - Container of app

- **Combinação-2**: Foram realizados testes com diferentes combinações de atributos para avaliar o desempenho na previsão da ação recomendada para o tráfego de rede. As combinações foram numeradas de forma crescente, gerando diversas possibilidades. No entanto, nem todas as combinações apresentaram resultados satisfatórios. Por exemplo, a Combinação-1 não foi incluída entre as melhores. Este conjunto de dados inclui os seguintes atributos:

  - Subcategory of app
  - Destination Port
  - IP Protocol
  - Application

- **Combinação-10**: Este conjunto de dados corresponde a uma das combinações geradas para avaliar o desempenho na previsão da ação recomendada para o tráfego de rede e é composto pelos seguintes atributos:

  - Container of app
  - Application
  - Source Country
  - Destination Zone
  - Elapsed Time (sec)

- **Combinação-12**: Este conjunto de dados refere-se a uma das combinações testadas para analisar o desempenho na previsão da ação recomendada para o tráfego de rede e é composto pelos seguintes atributos:

  - Destination Port
  - Packets
  - Application
  - Packets Received

- **Combinação-14**: Este conjunto de dados foi criado a partir de uma das combinações avaliadas para medir o desempenho na previsão da ação recomendada para o tráfego de rede, e inclui os seguintes atributos:
  - Source address
  - Source Country
  - Destination address
  - Source Zone
- **Combinação-17**: Este conjunto de dados faz parte de uma das combinações geradas durante a avaliação do desempenho na previsão da ação recomendada para o tráfego de rede, e é composto pelos seguintes atributos:
  - Subcategory of app
  - Destination Port
  - IP Protocol
  - Application

Na Tabela 7 é apresentado um resumo dos conjuntos de atributos, acompanhado de uma breve descrição e da lista completa de atributos utilizados.

Tabela 7 – Combinações de Atributos Utilizadas nos Experimentos

Conjunto	**Descrição**	**Atributos**
Todos os Atributos Importantes	Conjunto com todos os 26 atributos relevantes (exceto o alvo *Action*)	Source address, Destination address, NAT Source IP, NAT Destination IP, Application, Source Zone, Destination Zone, Source Port, Destination Port, NAT Source Port, NAT Destination Port, IP Protocol, Bytes, Bytes Sent, Bytes Received, Packets, Elapsed Time (sec), Category, Source Country, Destination Country, Packets Sent, Packets Received, Subcategory of app, Category of app, Technology of app, Container of app.

Continua na próxima página

Continuação da Tabela 7.

Conjunto	Descrição	Atributos
Todos os Atributos Importantes Sem IPs	Exclui atributos de endereços IP, mantendo os outros atributos importantes	Application, Source Zone, Destination Zone, Source Port, Destination Port, NAT Source Port, NAT Destination Port, IP Protocol, Bytes, Bytes Sent, Bytes Received, Packets, Elapsed Time (sec), Category, Source Country, Destination Country, Packets Sent, Packets Received, Subcategory of app, Category of app, Technology of app, Container of app.
Todos os Atributos Importantes Sem IPs e NAT	Exclui atributos de IP e do processo NAT	Application, Source Zone, Destination Zone, Source Port, Destination Port, IP Protocol, Bytes, Bytes Sent, Bytes Received, Packets, Elapsed Time (sec), Category, Source Country, Destination Country, Packets Sent, Packets Received, Subcategory of app, Category of app, Technology of app, Container of app.
Todos os Atributos Importantes Melhores Correlações	Atributos com maior correlação com o alvo *Action*	Destination address, NAT Source IP, NAT Destination IP, Application, Source Zone, Destination Zone, Source Port, Destination Port, NAT Source Port, IP Protocol, Category, Destination Country, Subcategory of app, Category of app, Technology of app, Container of app.
Todos os Atributos Importantes Melhores Correlações Positivas	Apenas atributos com correlação positiva com o alvo	Application, Destination Port, IP Protocol, Subcategory of app, Category of app, Technology of app, Container of app.
Todos os Atributos Importantes Sem IPs e com Combinação	Seleção reduzida, excluindo atributos de IP e mantendo combinação	Application, Source Zone, Destination Zone, Source Port, Destination Port, NAT Source Port, NAT Destination Port, IP Protocol, Category, Source Country, Destination Country, Subcategory of app, Category of app, Technology of app, Container of app.
Combinação-2	Conjunto criado para avaliar desempenho com atributos específicos	Subcategory of app, Destination Port, IP Protocol, Application.

Continua na próxima página

Continuação da Tabela 7.

Conjunto	Descrição	Atributos
Combinação-10	Conjunto criado para avaliar desempenho com atributos específicos	Container of app, Application, Source Country, Destination Zone, Elapsed Time (sec).
Combinação-12	Conjunto criado para avaliar desempenho com atributos específicos	Destination Port, Packets, Application, Packets Received.
Combinação-14	Conjunto criado para avaliar desempenho com atributos específicos	Source address, Source Country, Destination address, Source Zone.
Combinação-17	Conjunto criado para avaliar desempenho com atributos específicos	Subcategory of app, Destination Port, IP Protocol, Application.

Fonte: Elaborado pelo autor.

7.3 Experimentos - MINAS

Conforme explicado anteriormente, devido às características do algoritmo, não é necessário alterar sua implementação. Apenas ajustes nos hiperparâmetros são necessários para adaptá-lo à criação de extensões de conceitos conhecidos, quando necessário, em vez de identificar padrões novidade que poderiam representar o surgimento de novas classes.

7.3.1 Hiperparâmetros - MINAS

Nos experimentos, foi utilizado o algoritmo MINAS original (FARIA; GAMA; CARVALHO, 2013), implementado em Java. Diferentes configurações de hiperparâmetros influenciam o desempenho preditivo do MINAS. Foram realizados vários ajustes nos hiperparâmetros para cada combinação de atributos com o objetivo de otimizar o desempenho preditivo do modelo na classificação da ação recomendada para o tráfego de rede. Cada conjunto de atributos foi avaliado com diversas configurações de hiperparâmetros para determinar as melhores combinações que levaram aos melhores resultados analisados.

Na Tabela 8 é apresentado os hiperparâmetros do MINAS e suas respectivas descrições.

Tabela 8 – Hiperparâmetros do algoritmo MINAS e suas descrições.

Hiperparâmetro	**Descrição**
`algOff`	Algoritmo de agrupamento utilizado na fase offline: *kmeans* ou *clustream*. Em todos os experimentos realizados neste trabalho, adotou-se o algoritmo *clustream* devido aos seus resultados superiores.
`algOnl`	Algoritmo de agrupamento utilizado na fase online: *kmeans* ou *clustream*. Em todos os experimentos realizados neste trabalho, adotou-se o algoritmo *clustream* devido aos seus resultados superiores.
`threshold`	Limiar utilizado para considerar um microgrupo como novidade ou extensão de um conceito conhecido.
`thresholdForgetPast`	Tempo máximo que um microgrupo pode permanecer sem receber novos exemplos antes de ser movido para a memória sleep.
`nMicro`	Número de microgrogrupos gerados pelo algoritmo de agrupamento (*CluStream* ou *K-Means*).
`minExCluster`	O número mínimo de exemplos para um microgrupo ser considerado válido (representatividade).

Fonte: Elaborado pelo autor.

7.3.2 Experimento 1

Foi realizado um primeiro experimento (*Experimento* 1) utilizando diferentes combinações de atributos e ajustes nos hiperparâmetros. Para cada combinação de conjunto de atributos e hiperparâmetros, foi atribuído o nome de (modelo/matriz de confusão), com o intuito de identificar as configurações que gerassem os melhores resultados. No *Experimento* 1, foi utilizado o *primeiro conjunto de dados* mencionado neste capítulo na Seção 7.1.1 - Conjunto de Dados, onde o conjunto total foi dividido em 10% para a Fase Offline e 90% para a Fase Online.

Na Tabela 9, são apresentadas diferentes configurações de valores dos hiperparâmetros para diversos conjuntos de atributos utilizados no *Experimento* 1. O hiperparâmetro *Threshold* foi ajustado para evitar a criação de padrões novidade, aumentando seu valor para permitir um maior número de novidades através de extensões de conceitos já conhecidos. Alguns conjuntos de atributos possuem mais resultados exibidos na Tabela 9, pois apresentaram um desempenho superior com diferentes combinações de hiperparâmetros. Em contrapartida, os conjuntos de atributos que apresentaram menos resultados indicam que as variações nos hiperparâmetros não contribuíram para uma melhora no desempenho de predição.

Tabela 9 – Combinações de hiperparâmetros e conjuntos de atributos no Experimento 1.

Matriz de Confusão	**Conjunto de Atributos**	**Threshold**	**threshold Forget Past**	**n Micro**	**minEx Cluster**
1	Combinação-2	100.0	10000	10	20
2	Combinação-2	1000.0	10000	117	200
3	Todos os Atributos Importantes	6.0	100000	11	100
4	Todos os Atributos Importantes	6.0	100	11	100
5	Todos os Atributos Importantes	6.0	10000	11	100
6	Todos os Atributos Importantes	100.0	10000	11	400
7	Combinação-10	100.0	10000	100	200
8	Combinação-10	100.0	1000	30	200
9	Combinação-12	100.0	1000	15	500
10	Combinação-12	6.0	1000	17	500
11	Combinação-14	100.0	100000	8	580
12	Combinação-14	100.0	100000	15	580
13	Combinação-14	100.0	1000	15	1980
14	Combinação-14	100.0	100000	15	1980
15	Todos os Atributos Importantes Sem IPs	100.0	100000	116	580
16	Todos os Atributos Importantes Sem IPs	6.0	100000	11	100
17	Todos os Atributos Importantes Sem IPs	6.0	10000	11	100
18	Todos os Atributos Importantes Sem IPs	100.0	10000	11	400
19	Todos os Atributos Importantes Sem IPs	100.0	100000	23	800
20	Todos os Atributos Importantes Sem IPs	100.0	100000	4	800
21	Todos os Atributos Importantes Sem IPs	1000.0	100000	7	900
22	Todos os Atributos Importantes Sem IPs	100.0	10000	85	1000
23	Todos os Atributos Importantes Sem IPs	6.0	100000	3	1000
24	Combinação-17	100.0	10000	9	1000
25	Combinação-17	100.0	10000	9	800
26	Combinação-17	100.0	10000	9	100
27	Todos os Atributos Importantes Sem IPs e com Combinação	100.0	10000	3	1000
28	Todos os Atributos Importantes Sem IPs e com Combinação	6.0	1000	3	1000
29	Todos os Atributos Importantes Sem IPs e com Combinação	6.0	900	3	1000
30	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	6.0	900	117	1000
31	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	100.0	100000	43	1000
32	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	6.0	10000	3	1000
33	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	6.0	1000	3	1000
34	Todos os Atributos Importantes Melhores Correlações	100.0	100000	11	100
35	Todos os Atributos Importantes Melhores Correlações	100.0	100000	15	1000
36	Todos os Atributos Importantes Melhores Correlações Positivas	100.0	10000	3	1000
37	Todos os Atributos Importantes Melhores Correlações Positivas	100.0	100000	9	1000

Fonte: Elaborado pelo autor.

Os resultados do *Experimento 1* foram obtidos a partir da combinação de atributos e hiperparâmetros descritos na Tabela 9. Para cada conjunto de atributos e hiperparâmetros, a fase offline foi utilizada para o treinamento do modelo, enquanto a fase online foi empregada para simular um FCDs. Os resultados das métricas de desempenho para cada combinação estão apresentados na Tabela 10, que ordena as combinações em ordem decrescente com base no *F1-Score* mais alto.

Os resultados incluem métricas de desempenho como: a média de acerto das classes, calculada pela soma das porcentagens de acerto de cada classe dividida pelo total de classes; e as métricas clássicas: *acurácia, precisão, recall e F1-Score*. Além disso, é apresentado o percentual de acerto da classe C3 (*Reset-Both*). Esta última métrica é destacada devido à baixa frequência de exemplos de ações de "Reset-Both" no conjunto de dados, o que limita o número de instâncias disponíveis para o treinamento e a posterior classificação correta nesta classe.

O algoritmo MINAS possui uma coluna específica para exemplos desconhecidos, que representam padrões de tráfego de rede que o modelo ainda não conseguiu classificar até aquele momento do FCDs. À medida que o fluxo continua, é necessário tomar uma ação para esses exemplos desconhecidos, pois todo tráfego de rede precisa ser tratado de alguma forma. Como o tráfego desconhecido não corresponde a nenhum padrão conhecido, a ação mais apropriada é bloquear esse tráfego de rede descartando os pacotes. Para isso, o modelo adota a ação "C2-Drop" (descartar) para todos os exemplos desconhecidos no momento da avaliação de desempenho, garantindo que o tráfego que o modelo não sabe como tratar não seja permitido na rede.

Na avaliação de métricas como precisão, recall e F1-Score, é recomendado que a matriz de confusão seja quadrada (isto é, tenha o mesmo número de linhas e colunas). No entanto, a matriz de confusão gerada pelo MINAS possui uma coluna adicional para exemplos desconhecidos (Unk). Para ajustar essa matriz de confusão para a avaliação dessas métricas, os valores presentes na coluna "Unk" são realocados para a coluna correspondente à ação "Drop" (C2) no momento da avaliação de desempenho. Com isso, todos os exemplos desconhecidos são tratados como "Drop" (descartados), resultando em uma matriz de confusão quadrada que permite a avaliação adequada das métricas de desempenho. Essa abordagem também antecipa o tratamento necessário para esses dados, bloqueando-os na avaliação, uma vez que o tráfego desconhecido precisa ser descartado.

Para os 10 melhores resultados da Tabela 10, que obtiveram o maior desempenho em termos de (*F1-Score*) na classificação da ação recomendada para o tráfego de rede no *Experimento 1*, serão apresentadas tanto a matriz de confusão percentual quanto a matriz de confusão tradicional, conforme mostrado nas Tabelas 11 a 20. Essas matrizes oferecem uma visualização mais detalhada e um outro entendimento dos resultados obtidos, mostrando as diferentes ações previstas pelo modelo: C0 (Allow), C1 (Deny), C2 (Drop), C3 (Reset-Both) e Unk (Desconhecidos).

Tabela 10 – Resultados das métricas de desempenho no Experimento 1 em (%).

Matriz de Confusão	**Média Acerto Classes**	**Acurácia**	**Precisão**	**Recall**	**F1-Score**	**Acerto C3 RESET-BOTH**
7	52,49	78,09	89,01	78,16	81,46	0
4	52,98	79,70	83,27	79,70	80,99	0,2
8	48,06	76,26	82,38	76,68	79,12	0
30	51,31	74,75	84,55	74,93	77,94	0
33	76,96	75,57	80,12	75,57	77,26	95,64
23	76,51	75,57	79,80	75,57	77,17	95,44
10	50,75	75,58	80,43	76,58	76,89	0,2
32	76,35	75,22	79,61	75,23	76,88	95,44
29	68,78	74,62	80,15	74,62	76,73	64,60
28	76,13	73,78	80,11	73,78	76,21	95,64
27	75,10	72,34	78,96	72,35	74,84	95,44
22	54,55	69,56	81,87	69,57	72,77	0
2	55,03	69,06	92,55	69,07	71,93	0
15	53,40	67,98	81,16	68,05	71,85	0
31	69,51	64,83	83,23	64,85	70,89	76,47
18	62,84	66,17	78,67	66,17	69,53	73,63
16	67,18	64,92	78,53	64,92	68,55	93,31
17	62,02	64,41	78,11	64,41	67,97	73,63
24	54,47	61,46	81,61	61,47	67,97	14,20
6	64,88	63,98	75,16	64,00	67,50	72,21
21	72,12	63,35	75,84	63,35	67,37	97,06
25	55,70	60,40	81,85	60,42	67,32	21,20
5	61,66	62,68	72,90	62,98	66,00	68,15
3	68,24	62,56	73,06	62,56	65,96	94,73
9	50,67	61,07	75,93	61,09	65,11	0,10
26	60,71	57,08	81,13	57,08	64,27	47,57
34	69,95	55,96	73,73	55,96	60,84	87,32
13	51,02	56,59	68,19	56,60	60,73	37,73
35	68,83	53,96	72,61	53,96	59,09	85,60
37	59,52	54,42	73,98	54,42	58,76	47,36
36	71,04	51,24	81,66	51,24	58,41	95,44
14	60,18	50,71	68,20	50,73	56,67	86,00
12	59,35	46,92	67,50	46,93	53,54	90,26
1	55,57	47,40	70,51	47,40	52,85	45,03
19	63,47	45,75	69,24	45,76	51,64	84,38
20	64,13	46,55	70,22	46,55	49,20	100,00
11	53,99	40,23	60,50	40,24	46,25	88,84

Fonte: Elaborado pelo autor.

Tabela 11 – Matriz de Confusão 7 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	96.66%	0.55%	1.06%	0.00%	1.74%
C 1	0.43%	56.88%	42.46%	0.00%	0.23%
C 2	7.99%	34.55%	56.42%	0.00%	1.03%
C 3	87.93%	0.00%	0.00%	0.00%	12.07%

	C 0	C 1	C 2	C 3	Unk
C 0	466975	2637	5107	0	8412
C 1	1540	201512	150434	0	814
C 2	5080	21951	35844	0	652
C 3	867	0	0	0	119

Fonte: Elaborado pelo autor.

Tabela 12 – Matriz de Confusão 4 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	91.64%	6.94%	1.32%	0.02%	0.08%
C 1	7.74%	68.73%	23.48%	0.00%	0.06%
C 2	14.98%	33.68%	51.34%	0.00%	0.00%
C 3	94.02%	0.51%	0.00%	0.20%	5.27%

	C 0	C 1	C 2	C 3	Unk
C 0	442722	33538	6359	103	409
C 1	27406	243505	83174	0	215
C 2	9515	21394	32617	0	1
C 3	927	5	0	2	52

Fonte: Elaborado pelo autor.

Tabela 13 – Matriz de Confusão 8 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	88.54%	9.95%	1.24%	0.00%	0.28%
C 1	1.81%	68.07%	30.08%	0.00%	0.03%
C 2	7.76%	56.55%	35.61%	0.00%	0.07%
C 3	84.99%	4.16%	8.22%	0.00%	2.64%

	C 0	C 1	C 2	C 3	Unk
C 0	427744	48049	6002	0	1336
C 1	6417	241187	106583	0	113
C 2	4931	35926	22623	0	47
C 3	838	41	81	0	26

Fonte: Elaborado pelo autor.

Tabela 14 – Matriz de Confusão 30 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	89.84%	4.40%	1.09%	0.00%	4.67%
C 1	6.26%	57.37%	34.86%	0.00%	1.51%
C 2	16.37%	22.97%	58.04%	0.00%	2.63%
C 3	52.84%	0.00%	0.00%	0.00%	47.16%

	C 0	C 1	C 2	C 3	Unk
C 0	434028	21259	5264	0	22580
C 1	22166	203277	123512	0	5345
C 2	10398	14593	36868	0	1668
C 3	521	0	0	0	465

Fonte: Elaborado pelo autor.

Tabela 15 – Matriz de Confusão 23 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	82.40%	16.14%	0.22%	1.13%	0.12%
C 1	7.52%	69.16%	23.27%	0.01%	0.04%
C 2	7.16%	33.72%	59.04%	0.02%	0.06%
C 3	4.56%	0.00%	0.00%	95.44%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	398088	77971	1039	5474	559
C 1	26648	245038	82456	28	130
C 2	4547	21420	37508	12	40
C 3	45	0	0	941	0

Fonte: Elaborado pelo autor.

Tabela 16 – Matriz de Confusão 33 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	81.86%	16.58%	0.23%	1.13%	0.20%
C 1	6.76%	69.61%	23.60%	0.01%	0.03%
C 2	5.17%	34.08%	60.73%	0.02%	0.00%
C 3	4.36%	0.00%	0.00%	95.64%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	395474	80105	1107	5460	985
C 1	23937	246613	83615	28	107
C 2	3286	21648	38580	12	1
C 3	43	0	0	943	0

Fonte: Elaborado pelo autor.

Tabela 17 – Matriz de Confusão 10 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	75.11%	21.41%	3.19%	0.00%	0.29%
C 1	3.59%	81.73%	14.21%	0.00%	0.47%
C 2	0.26%	53.67%	45.96%	0.00%	0.11%
C 3	69.68%	0.51%	0.00%	0.20%	29.61%

	C 0	C 1	C 2	C 3	Unk
C 0	362888	103421	15434	1	1387
C 1	12730	289567	50350	0	1653
C 2	162	34096	29198	0	71
C 3	687	5	0	2	292

Fonte: Elaborado pelo autor.

Tabela 18 – Matriz de Confusão 32 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	81.75%	16.60%	0.21%	1.13%	0.31%
C 1	7.46%	69.16%	23.27%	0.01%	0.11%
C 2	7.07%	33.72%	59.03%	0.02%	0.16%
C 3	4.56%	0.00%	0.00%	95.44%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	394947	80194	1038	5456	1496
C 1	26439	245026	82429	23	383
C 2	4493	21420	37499	12	103
C 3	45	0	0	941	0

Fonte: Elaborado pelo autor.

Tabela 19 – Matriz de Confusão 29 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	80.12%	16.22%	0.23%	3.07%	0.36%
C 1	6.72%	69.61%	23.63%	0.02%	0.02%
C 2	5.10%	34.09%	60.80%	0.01%	0.00%
C 3	35.40%	0.00%	0.00%	64.60%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	387096	78373	1109	14812	1741
C 1	23805	246641	83713	69	72
C 2	3240	21654	38627	5	1
C 3	349	0	0	637	0

Fonte: Elaborado pelo autor.

Tabela 20 – Matriz de Confusão 28 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	78.51%	16.22%	0.23%	4.62%	0.42%
C 1	6.72%	69.61%	23.62%	0.03%	0.02%
C 2	5.14%	34.08%	60.76%	0.01%	0.00%
C 3	4.36%	0.00%	0.00%	95.64%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	379307	78370	1108	22298	2048
C 1	23799	246618	83690	110	83
C 2	3267	21648	38602	9	1
C 3	43	0	0	943	0

Fonte: Elaborado pelo autor.

Além da apresentação das matrizes de confusão dos 10 melhores resultados da Tabela 10 — selecionados com base no maior desempenho em termos de (*F1-Score*) na classificação da ação recomendada para o tráfego de rede no *Experimento 1* —, na Tabela 21 são apresentados os resultados dos Erros de Classificação Críticos para esses modelos. Essa métrica é especialmente relevante, pois valores menores indicam menor confusão dos modelos na distinção entre ações de permissão (C0 - Allow) e bloqueio (C1 - Deny, C2 - Drop e C3 - Reset-Both) de tráfego de rede, refletindo maior precisão na classificação.

Tabela 21 – Resultados da métrica de desempenho "Erros de Classificação Críticos" no Experimento 1

Matriz de Confusão	**Erro Permitir X Bloquear (%)**
7	2,61
4	8,67
8	7,49
30	9,11
33	12,74
23	12,89
10	14,84
32	13,21
29	13,68
28	14,52

Fonte: Elaborado pelo autor.

7.3.2.1 Conclusões e Discussões dos Resultados do Experimento 1

Os resultados do *Experimento 1* são apresentados na Tabela 10, com diferentes combinações de atributos e configurações de hiperparâmetros utilizados no algoritmo MINAS. O objetivo principal deste experimento foi identificar as configurações que geraram os melhores resultados na classificação da ação recomendada para o tráfego de rede, capturado por *logs* de sessão do *firewall* Palo Alto.

Desempenho Geral e Critérios de Avaliação

Os resultados foram avaliados com base em métricas clássicas de desempenho, incluindo acurácia, precisão, recall, e F1-Score, além da média de acerto das classes e do percentual de acerto da classe "C3 - Reset-Both". Devido ao desbalanceamento dos dados e à baixa frequência da classe "C3 - Reset-Both", o F1-Score se torna uma métrica mais representativa, pois combina a precisão e o recall, oferecendo uma medida mais equilibrada do desempenho preditivo.

Outro fator importante considerado na análise é a capacidade dos modelos de se adaptarem a mudanças abruptas no fluxo de tráfego, típicas de cenários de FCDs. Nessas condições, o modelo deve ser robusto o suficiente para capturar variações inesperadas nos

padrões de tráfego de rede, especialmente para a classe rara "C3 - Reset-Both". Portanto, foi dada especial atenção aos modelos que evitaram 0% de acerto para a classe "C3 - Reset-Both" e mantiveram um bom equilíbrio entre todas as classes.

Erros de Classificação Críticos

Uma consideração importante na seleção dos modelos mais apropriados foi a análise dos erros de classificação críticos:

- **Falsos negativos para "Allow (C0)"**: Casos em que tráfego permitido (C0) é incorretamente classificado como uma ação de bloqueio ("C1 - Deny", "C2 - Drop" ou "C3 - Reset-Both"). Esses erros podem resultar na interrupção de tráfego legítimo e são indesejáveis em um ambiente de rede;
- **Falsos positivos para "Allow (C0)"**: Casos em que tráfego que deveria ser bloqueado (C1, C2, C3) é erroneamente classificado como permitido (C0). Esses erros são especialmente perigosos, pois podem representar falhas de segurança que podem permitir o tráfego potencialmente perigoso.

Análise dos Melhores Modelos

Para a seleção dos melhores modelos, foi adotada uma abordagem equilibrada de análise das métricas de desempenho. Em vez de priorizar modelos que apresentaram resultados excelentes em uma métrica isolada, mas fracos em outras, buscou-se identificar aqueles com desempenho consistente em múltiplas métricas. Essa escolha visa garantir modelos mais robustos e confiáveis no contexto de FCDs.

Com base nos critérios mencionados, os três modelos mais adequados, considerando a adaptabilidade a mudanças de fluxo, a minimização dos erros críticos de classificação e o desempenho geral em termos de F1-Score e acertos por classes, são apresentados a seguir:

1. **Matriz de Confusão 33 - Tabela 10** (F1-Score: 77,26%, Acerto "C3 - Reset-Both": 95,64%): Este modelo destacou-se pelo bom F1-Score em comparação com os demais e pela elevada precisão na classe "C3 - Reset-Both", atingindo 95,64% de acerto, o que é fundamental para capturar mudanças de padrões de tráfego em fluxos contínuos de dados. Além disso, apresentou a melhor taxa média de acerto nas classes, com 76,96%. Os erros de classificação críticos foram minimizados: "C0 - Allow" foi erroneamente classificado como "C1 - Deny" em 16,58% dos casos, como "C2 - Drop" em apenas 0,23%, e como "C3 - Reset-Both" em 1,13%. Por outro lado, as classes de bloqueio (C1, C2, C3) foram classificadas como "C0 - Allow"

com uma taxa de erro baixa (entre 4,36% e 6,76%). Esses resultados indicam um bom equilíbrio entre precisão e adaptabilidade;

2. **Matriz de Confusão 23 - Tabela 10** (F1-Score: 77,17%, Acerto "C3 - Reset-Both": 95,44%): Semelhante ao modelo anterior, este modelo destacou-se pelo bom F1-Score em comparação com os demais e pela elevada precisão na classe "C3 - Reset-Both" (95,44%). Ele também apresentou a segunda melhor taxa média de acerto nas classes, com 76,51%. O erro de classificação de "C0 - Allow" como "C1 - Deny" foi de 16,14%, "C0 - Allow" como "C2 - Drop" foi de 0,22%, e "C0 - Allow" como "C3 - Reset-Both" foi de 1,13%. Enquanto isso, o erro de classificação de "C1 - Deny" como "C0 - Allow" foi de 7,52%, "C2 - Drop" como "C0 - Allow" foi de 7,16%, e "C3 - Reset-Both" como "C0 - Allow" foi de 4,56%. A pequena variação nos erros em comparação com a Matriz de Confusão 33 torna este modelo igualmente superior aos outros modelos testados até o momento, quando analisadas todas as métricas;

3. **Matriz de Confusão 32 - Tabela 10** (F1-Score: 76,88%, Acerto "C3 - Reset-Both": 95,44%): Este modelo apresenta um desempenho equilibrado, com um acerto de 95,44% para a classe "C3 - Reset-Both", o que é relevante para capturar padrões de tráfego menos frequentes em um cenário de FCDs. Além disso, teve um F1-Score muito próximo aos modelos anteriores e a terceira melhor taxa média de acerto nas classes, com 76,35%. Os erros de classificação de "C0 - Allow" como "C1 - Deny", "C2 - Drop" e "C3 - Reset-Both" são 16,60%, 0,21% e 1,13%, respectivamente, que estão bem próximos quando comparados as Matrizes de Confusão 23 e 33. Adicionalmente, os erros de "C1 - Deny" como "C0 - Allow" foram de 7,46%, "C2 - Drop" como "C0 - Allow" foram de 7,07%, e "C3 - Reset-Both" como "C0 - Allow" foram de 4,56%. Isso demonstra que o modelo é adequado, ao avaliar as métricas de forma geral e ao compará-las com as dos demais modelos.

Conclusão

Os três modelos selecionados - Matriz de Confusão 23, 32 e 33 - demonstraram um desempenho robusto na previsão das ações corretas para os exemplos do fluxo, um equilíbrio no acerto entre as classes e na minimização de erros de classificação críticos, quando comparados aos demais. Em particular, as Matrizes de Confusão 23, 32 e 33 se destacam pela boa precisão na classe rara "C3 - Reset-Both" e oferecem uma opção viável com um bom desempenho geral, seguida pelas Matrizes 29 e 28. Esses modelos demonstraram ser os mais equilibrados para cenários de FCDs entre todos os avaliados até o momento. Se considerássemos apenas o fluxo de dados atual e o melhor desempenho de classificação — com foco em métricas como Acurácia, Precisão e F1-Score — sem levar em conta o

equilíbrio entre as classes, ou a capacidade de projeção para classificações futuras, onde o tráfego pode mudar de maneira abrupta, as melhores opções para a previsão da ação recomendada no tráfego de rede seriam as Matrizes de Confusão 7, 4 e 8. Esses modelos apresentaram F1-Scores de 81,46%, 80,99% e 79,12% respectivamente, destacando-se por sua taxa de acerto elevada.

7.3.3 Experimento 2

O *Experimento 2* tem como objetivo avaliar o desempenho do algoritmo MINAS em um fluxo de dados coletado 15 dias após o *primeiro conjunto*. A proposta é analisar como o algoritmo se comporta ao processar um novo fluxo de dados, coletado em um período diferente do utilizado e testado no *Experimento 1*, o que pode refletir variações no comportamento ou nas características dos dados ao longo do tempo. Ao introduzir um novo fluxo de dados, esse experimento investiga a capacidade do modelo de generalizar para dados com características potencialmente diferentes.

No *Experimento 2*, foram utilizadas as mesmas combinações de atributos e configurações de hiperparâmetros que resultaram nos 10 melhores desempenhos do *Experimento 1*, conforme apresentados na Tabela 10. Essa tabela organiza as combinações de atributos em ordem decrescente de acordo com o maior *F1-Score* obtido. No entanto, no *Experimento 2* foi utilizado um *segundo conjunto de dados*, mencionado na subseção 7.1.1 - Conjunto de Dados, com o intuito de simular um novo fluxo (Fase Online 2) do FCDs. A fase offline empregada para gerar o modelo permaneceu a mesma do *primeiro conjunto de dados*, correspondente a 10% do total desse conjunto de dados.

Na Tabela 22 são apresentadas as combinações de atributos e as configurações de hiperparâmetros que foram aplicadas no *Experimento 2*. A numeração das matrizes de confusão foi mantida a mesma dos 10 melhores resultados do *Experimento 1*, a fim de facilitar a comparação entre os resultados obtidos. Novamente, a matriz de confusão foi ajustada para a avaliação das métricas de precisão, recall e F1-Score, realocando os valores da coluna “Unk” para a coluna correspondente à ação “Drop” (C2). Dessa forma, todos os exemplos classificados como desconhecidos foram tratados como “Drop” (descartados), resultando em uma matriz de confusão quadrada, que permite uma avaliação mais adequada das métricas de desempenho e elimina o tráfego de rede para o qual o modelo não possui uma ação definida, ou seja, tráfego desconhecido.

Tabela 22 – Configurações de hiperparâmetros para diferentes conjuntos de atributos no Experimento 2.

Matriz de Confusão	**Conjunto de Atributos**	**Threshold**	**threshold Forget Past**	**n Micro**	**minEx Cluster**
4	Todos os Atributos Importantes	6.0	100	11	100

Continuação da Tabela 22

Matriz de Confusão	**Conjunto de Atributos**	**Threshold**	**threshold Forget Past**	**n Micro**	**minEx Cluster**
7	Combinação-10	100.0	10000	100	200
8	Combinação-10	100.0	1000	30	200
10	Combinação-12	6.0	1000	17	500
23	Todos os Atributos Importantes Sem IPs	6.0	100000	3	1000
28	Todos os Atributos Importantes Sem IPs e com Combinação	6.0	1000	3	1000
29	Todos os Atributos Importantes Sem IPs e com Combinação	6.0	900	3	1000
30	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	6.0	900	117	1000
32	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	6.0	10000	3	1000
33	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	6.0	1000	3	1000

Fonte: Elaborado pelo autor.

Os resultados do *Experimento 2* foram obtidos com base nas combinações de atributos e configurações de hiperparâmetros detalhadas na Tabela 22. Para cada conjunto de atributos e hiperparâmetros, a fase offline (utilizando 10% do primeiro conjunto de dados, o mesmo do *Experimento 1*) foi usada para o treinamento do modelo, enquanto a fase online (segundo conjunto de dados) foi aplicada para simular um novo fluxo do FCDs. Os resultados das métricas de desempenho para cada combinação estão apresentados na Tabela 23, que mostra, para cada métrica, o resultado obtido no *Experimento 1* seguido do resultado correspondente no *Experimento 2*. As linhas destacadas em verde indicam que o modelo do *Experimento 2* apresentou desempenho superior em comparação aos modelos do *Experimento 1* nas métricas de acurácia, precisão, recall e F1-score.

Tabela 23 – Resultados das métricas de desempenho no Experimento 1/Experimento 2 em (%).

Matriz de Confusão	**Média Acerto Classes**	**Acurácia**	**Precisão**	**Recall**	**F1-Score**	**Acerto C3 RESET-BOTH**
7	52,49/24,49	78,09/53,43	89,01/65,51	78,16/53,71	81,46/39	0/0
8	48,06/24,90	76,26/54,38	82,38/29,82	76,68/54,39	79,12/38,47	0/0
10	50,75/27,67	75,58/48,59	80,43/52,33	76,58/48,59	76,89/46,01	0,2/0
23	76,51/73,82	75,57/70,45	79,8/73,73	75,57/70,47	77,17/71,66	95,44/97,12
28	76,13/62,71	73,78/75,43	80,11/81,40	73,78/76,97	76,21/78,47	95,64/41,08
29	68,78/63,41	74,62/76,97	80,15/81,40	74,62/76,97	76,73/78,47	64,6/41,08
30	51,31/45,75	74,75/51,65	84,55/82,03	74,93/51,74	77,94/60,19	0/0
32	76,35/73,09	75,22/67,36	79,61/73,83	75,23/67,37	76,88/69,59	95,44/97,12
33	76,96/64,63	75,57/79,71	80,12/82,06	75,57/79,71	77,26/80,23	95,64/41,37

Fonte: Elaborado pelo autor.

Foi observado que o conjunto de atributos "Todos os Atributos Importantes - Matriz de Confusão 4" da Tabela 22 gerou muitas classes novidade, mesmo com o hiperparâmetro *Threshold* configurado para um valor alto. O modelo, no entanto, deveria ter identificado extensões de um conceito conhecido, dado que o número de classes é fixo — como ocorreu com os quatro resultados obtidos com esse conjunto de atributos no *Experimento 1* (Matrizes de Confusão: 3, 4, 5 e 6 da Tabela 10).

Essa discrepância pode ser atribuída a diferenças na natureza dos fluxos de dados entre os dois experimentos. No *Experimento 1*, o fluxo de dados pode ter apresentado características mais homogêneas e mais próximas do conjunto inicial de treinamento, o que permitiu ao modelo identificar extensões de conceitos conhecidos. Já no *Experimento 2*, o fluxo de dados coletado 15 dias após o primeiro pode ter apresentado uma variabilidade ou mudança nos padrões de tráfego muito maior, com esse conjunto de atributos em específico. Isso pode ter dificultado a identificação de extensões de conceitos conhecidos e levado à geração de classes novidade.

Devido a esse comportamento inesperado, este conjunto de atributos foi excluído dos resultados apresentados no *Experimento 2*. Para os demais 9 modelos, serão apresentadas tanto a matriz de confusão percentual quanto a tradicional, conforme mostrado nas Tabelas 24 a 32. Essas matrizes oferecem uma visualização mais detalhada e um outro entendimento dos resultados obtidos.

Tabela 24 – Experimento 2 - Matriz de Confusão 7 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	97.70%	0.02%	0.00%	0.00%	2.28%
C 1	96.15%	0.24%	0.00%	0.00%	3.61%
C 2	95.53%	0.00%	0.00%	0.00%	4.47%
C 3	95.35%	0.00%	0.00%	0.00%	4.65%

	C 0	C 1	C 2	C 3	Unk
C 0	275008	54	0	0	6406
C 1	192511	475	0	0	7227
C 2	31083	0	0	0	1455
C 3	1293	0	0	0	63

Fonte: Elaborado pelo autor.

Tabela 25 – Experimento 2 - Matriz de Confusão 8 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	99.62%	0.01%	0.00%	0.00%	0.38%
C 1	99.96%	0.00%	0.00%	0.00%	0.42%
C 2	99.96%	0.00%	0.00%	0.00%	0.37%
C 3	98.67%	0.00%	0.00%	0.00%	1.33%

	C 0	C 1	C 2	C 3	Unk
C 0	280394	3	0	0	1071
C 1	200129	0	0	0	84
C 2	32526	0	0	0	12
C 3	1338	0	0	0	18

Fonte: Elaborado pelo autor.

Além da apresentação das 9 matrizes de confusão, na Tabela 33 são apresentados os resultados dos Erros de Classificação Críticos para esses modelos. Essa métrica é

Tabela 26 – Experimento 2 - Matriz de Confusão 10 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	35.53%	64.04%	0.00%	0.00%	0.43%
C 1	24.70%	75.18%	0.00%	0.00%	0.12%
C 2	1.74%	98.26%	0.00%	0.00%	0.00%
C 3	46.24%	41.74%	0.00%	0.00%	12.02%

	C 0	C 1	C 2	C 3	Unk
C 0	100002	180249	10	4	1203
C 1	49456	150513	0	0	244
C 2	567	31971	0	0	0
C 3	627	566	0	0	163

Fonte: Elaborado pelo autor.

Tabela 27 – Experimento 2 - Matriz de Confusão 23 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	76.18%	21.10%	0.17%	1.62%	0.93%
C 1	16.85%	64.30%	18.72%	0.02%	0.12%
C 2	13.10%	28.90%	57.67%	0.01%	0.31%
C 3	2.88%	0.00%	0.00%	97.12%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	214423	59392	477	4559	2617
C 1	33742	128729	37480	32	230
C 2	4263	9405	18766	3	101
C 3	39	0	0	1317	0

Fonte: Elaborado pelo autor.

Tabela 28 – Experimento 2 - Matriz de Confusão 28 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	84.70%	6.54%	5.59%	3.14%	0.03%
C 1	15.68%	65.18%	19.12%	0.01%	0.01%
C 2	10.54%	29.58%	59.87%	0.00%	0.01%
C 3	58.92%	0.00%	0.00%	41.08%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	238395	18421	15736	8830	86
C 1	31390	130493	38285	17	28
C 2	3430	9626	19479	0	3
C 3	799	0	0	557	0

Fonte: Elaborado pelo autor.

Tabela 29 – Experimento 2 - Matriz de Confusão 29 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	87.48%	6.82%	2.51%	3.13%	0.06%
C 1	15.72%	65.21%	19.02%	0.01%	0.04%
C 2	10.55%	29.58%	59.85%	0.00%	0.02%
C 3	58.92%	0.00%	0.00%	41.08%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	246234	19190	7052	8819	173
C 1	31478	130553	38084	15	83
C 2	3433	9626	19473	0	6
C 3	799	0	0	557	0

Fonte: Elaborado pelo autor.

Tabela 30 – Experimento 2 - Matriz de Confusão 30 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	56.67%	10.07%	22.61%	0.00%	10.65%
C 1	1.07%	39.19%	58.11%	0.00%	1.63%
C 2	1.10%	10.35%	87.14%	0.00%	1.42%
C 3	27.14%	0.15%	0.37%	0.00%	72.35%

	C 0	C 1	C 2	C 3	Unk
C 0	159501	28352	63629	0	29986
C 1	2142	78461	116349	0	3261
C 2	358	3367	28352	0	461
C 3	368	2	5	0	981

Fonte: Elaborado pelo autor.

Tabela 31 – Experimento 2 - Matriz de Confusão 32 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	70.18%	21.45%	6.09%	1.61%	0.67%
C 1	14.13%	64.26%	21.52%	0.02%	0.08%
C 2	10.05%	28.90%	60.80%	0.01%	0.24%
C 3	2.58%	0.00%	0.29%	97.12%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	197530	60375	17151	4525	1887
C 1	28283	128660	43082	31	157
C 2	3271	9404	19782	3	78
C 3	35	0	4	1317	0

Fonte: Elaborado pelo autor.

Tabela 32 – Experimento 2 - Matriz de Confusão 33 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	92.58%	6.54%	0.19%	0.66%	0.02%
C 1	15.91%	65.17%	18.91%	0.01%	0.01%
C 2	10.99%	29.58%	59.43%	0.00%	0.00%
C 3	58.63%	0.00%	0.00%	41.37%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	260592	18419	536	1870	51
C 1	31859	130469	37857	12	16
C 2	3575	9626	19337	0	0
C 3	795	0	0	561	0

Fonte: Elaborado pelo autor.

especialmente relevante, pois valores menores indicam menor confusão dos modelos na distinção entre ações de permissão (C0 - Allow) e bloqueio (C1 - Deny, C2 - Drop e C3 - Reset-Both) de tráfego de rede, refletindo maior precisão na classificação.

7.3.3.1 Conclusões e Discussões dos Resultados do Experimento 2

Os resultados do *Experimento 2* são apresentados na Tabela 23, com diferentes combinações de atributos e configurações de hiperparâmetros utilizados no algoritmo MINAS. O objetivo principal deste experimento foi avaliar as mesmas configurações que geraram os melhores resultados no *Experimento 1*, mas utilizando um novo fluxo de dados capturado por *logs* de sessão do *firewall* Palo Alto, simulando um novo cenário do FCDs.

Desempenho Geral e Critérios de Avaliação

Os resultados foram avaliados com base nas mesmas métricas de desempenho utilizadas no *Experimento 1*, incluindo acurácia, precisão, recall, e F1-Score, além da média de

Tabela 33 – Resultados da métrica de desempenho “Erros de Classificação Críticos” no Experimento 2

Matriz de Confusão	Erro Permitir X Bloquear (%)
7	44,87
8	45,59
10	45,02
23	20,38
28	15,30
29	13,76
30	24,21
32	22,40
33	11,07

Fonte: Elaborado pelo autor.

acerto das classes e do percentual de acerto da classe “C3 - Reset-Both”. Como no experimento anterior, devido ao desbalanceamento dos dados e à baixa frequência da classe “C3 - Reset-Both”, o F1-Score é a métrica mais representativa, pois combina a precisão e o recall, oferecendo uma medida mais equilibrada do desempenho preditivo.

Outro fator importante considerado na análise é a capacidade dos modelos de se adaptarem a mudanças abruptas no fluxo de tráfego, típicas de cenários de FCDs. Nessas condições, o modelo deve ser robusto o suficiente para capturar variações inesperadas nos padrões de tráfego de rede, especialmente para a classe rara “C3 - Reset-Both”. Portanto, foi dada especial atenção aos modelos que evitaram 0% de acerto para a classe “C3 - Reset-Both” e mantiveram um bom equilíbrio entre todas as classes.

Erros de Classificação Críticos

Uma consideração importante na seleção dos modelos mais apropriados foi a análise dos erros de classificação críticos:

- **Falsos negativos para “Allow (C0)”**: Casos em que tráfego permitido (C0) é incorretamente classificado como uma ação de bloqueio (“C1 - Deny”, “C2 - Drop” ou “C3 - Reset-Both”). Esses erros podem resultar na interrupção de tráfego legítimo e são indesejáveis em um ambiente de rede;

- **Falsos positivos para “Allow (C0)”**: Casos em que tráfego que deveria ser bloqueado (C1, C2, C3) é erroneamente classificado como permitido (C0). Esses erros são especialmente perigosos, pois podem representar falhas de segurança que podem permitir o tráfego potencialmente perigoso.

Análise dos Melhores Modelos

Para a seleção dos melhores modelos, foi adotada uma abordagem equilibrada de análise das métricas de desempenho. Em vez de priorizar modelos que apresentaram resultados excelentes em uma métrica isolada, mas fracos em outras, buscou-se identificar aqueles com desempenho consistente em múltiplas métricas. Essa escolha visa garantir modelos mais robustos e confiáveis no contexto de FCDs.

Com base nos critérios mencionados, os três modelos mais adequados do *Experimento 2* considerando a adaptabilidade a mudanças de fluxo, a minimização dos erros críticos de classificação e o desempenho geral em termos de F1-Score e acertos por classes, são apresentados a seguir:

1. **Matriz de Confusão 23 - Tabela 23** (F1-Score: 71,66%, Acerto C3 - Reset-Both: 97,12%): Este modelo apresentou um bom desempenho para a classe "C3 - Reset-Both" (97,12%) e um F1-Score considerado satisfatório, sendo o quarto melhor. Além disso, destacou-se com a melhor taxa média de acerto de 73,82% nas classes, demonstrando um desempenho equilibrado. Os erros de classificação críticos foram: "C0 - Allow" foi erroneamente classificado como "C1 - Deny" em 21,10% dos casos, como "C2 - Drop" em apenas 0,17%, e como "C3 - Reset-Both" em 1,62%. As classes de bloqueio (C1, C2, C3) foram classificadas como "C0 - Allow" com uma taxa de erro (entre 2,88% e 16,85%);

2. **Matriz de Confusão 32 - Tabela 23** (F1-Score: 69,59%, Acerto C3 - Reset-Both: 97,12%): Este modelo apresentou um desempenho equilibrado, com um acerto de 97,12% para a classe "C3 - Reset-Both", o que é relevante para capturar padrões de tráfego menos frequentes em um cenário de FCDs. Além disso, teve um F1-Score muito próximo ao modelo anterior e a segunda melhor taxa média de acerto nas classes, com 73,09%. Os erros de classificação de "C0 - Allow" como "C1 - Deny", "C2 - Drop" e "C3 - Reset-Both" foram 21,45%, 6,09% e 1,61%, respectivamente, que são aceitáveis quando comparado a Matriz de Confusão 23 . Adicionalmente, os erros de "C1 - Deny" como "C0 - Allow" foram de 14,13%, "C2 - Drop" como "C0 - Allow" foram de 10,05%, e "C3 - Reset-Both" como "C0 - Allow" foram de 2,58%;

3. **Matriz de Confusão 33 - Tabela 23** (F1-Score: 80,23%, Acerto C3 - Reset-Both: 41,37%): Este modelo destacou-se por reduzir a confusão entre as ações de permitir e bloquear o tráfego de rede. Apresentou o melhor desempenho geral em termos de acurácia (79,81%) e F1-Score (80,23%) em comparação com os outros modelos, o que sugere um bom desempenho preditivo para as demais classes, mesmo em um novo fluxo de dados. Com uma taxa média de acerto nas classes de 64,83%, foi o

terceiro melhor modelo nesse aspecto. No entanto, o acerto da classe "C3 - Reset-Both" foi relativamente baixo (41,37%), indicando uma dificuldade em capturar mudanças de padrões de tráfego que representem a classe rara "C3 - Reset-Both" neste novo cenário do FCDs. Os erros de classificação críticos foram : "C0 - Allow" foi erroneamente classificado como "C1 - Deny" em 6,54% dos casos, como "C2 - Drop" em apenas 0,19%, e como "C3 - Reset-Both" em 0,66%. As classes de bloqueio (C1, C2, C3) foram classificadas como "C0 - Allow" com uma taxa de erro entre 10,99% e 58,63%.

Comparação de Desempenho Entre os Experimentos 1 e 2

Como foi observado na Tabela 23, ao comparar os resultados do *Experimento 2* com o *Experimento 1*, observa-se que alguns modelos apresentaram um desempenho superior em termos de métricas como acurácia, precisão, recall e F1-Score. Esses modelos são destacados a seguir:

- **Matriz de Confusão 28 (Experimento 2 - F1-Score: 78,47%, Acerto C3 - Reset-Both: 41,08%)**: Este modelo apresentou um aumento no F1-Score (78,47%) em comparação ao Experimento 1 (76,21%). A acurácia e precisão também melhoraram no Experimento 2. Apesar do aumento no F1-Score, o acerto da classe "C3 - Reset-Both" diminuiu de (95,64% no Experimento 1 para 41,08% no Experimento 2);

- **Matriz de Confusão 29 (Experimento 2 - F1-Score: 78,47%, Acerto C3 - Reset-Both: 41,08%)**: Similar à Matriz de Confusão 28, este modelo apresentou um F1-Score superior no Experimento 2 (78,47%) comparado ao Experimento 1 (76,73%). No entanto, houve uma queda acentuada no acerto da classe "C3 - Reset-Both" de 64,6% (Experimento 1) para 41,08% (Experimento 2);

- **Matriz de Confusão 33 (Experimento 2 - F1-Score: 80,23%, Acerto C3 - Reset-Both: 41,37%)**: A Matriz de Confusão 33 apresentou o maior F1-Score entre os modelos do Experimento 2 (80,23%), superando seu desempenho no Experimento 1 (77,26%). Apesar do alto F1-Score geral, o acerto da classe "C3 - Reset-Both" caiu significativamente de 95,64% para 41,37%.

Conclusão

Os três modelos selecionados - Matriz de Confusão 23, 32 e 33 - demonstraram um desempenho robusto na previsão das ações corretas para os exemplos do fluxo, um equilíbrio no acerto entre as classes e na minimização de erros de classificação críticos, seguidos de perto pelas Matrizes 29 e 28. Em particular, as Matrizes de Confusão 23 e 32 se destacam

pela alta precisão na classe rara "C3 - Reset-Both", enquanto a Matriz de Confusão 33 oferece uma opção viável com um bom desempenho geral. Esses modelos são, portanto, considerados os mais equilibrados para cenários de FCDs, entre todos os modelos testados no *Experimento 2*.

Se considerássemos apenas o fluxo de dados atual e o melhor desempenho de classificação — com foco em métricas como Acurácia, Precisão e F1-Score — sem levar em conta o equilíbrio entre as classes, ou a capacidade de projeção para classificações futuras, onde o tráfego pode mudar de maneira abrupta, as melhores opções para a previsão da ação recomendada no tráfego de rede seriam as Matrizes de Confusão 33, 29 e 28. Esses modelos apresentaram F1-Scores de 80,23%, 78,47% e 78,47% respectivamente, destacando-se por sua taxa de acerto elevada.

7.3.4 Experimento 3

Considerando que o algoritmo MINAS é incremental e se adapta conforme o fluxo de dados ocorre, o objetivo desse experimento foi avaliar o comportamento do algoritmo ao longo do tempo, observando como ele se ajusta ao *primeiro conjunto de dados* e, posteriormente, como ele reage ao *segundo conjunto*. Diferentemente do *Experimento 2*, onde o modelo foi treinado apenas com 10% do primeiro conjunto de dados, sem se adaptar ao restante do fluxo desse conjunto, neste caso, analisa-se a capacidade de adaptação contínua do algoritmo ao longo do fluxo completo (primeiro e segundo conjuntos de dados).

No *Experimento 3*, foram empregadas as mesmas combinações de atributos e configurações de hiperparâmetros que resultaram nos 10 melhores desempenhos do *Experimento 1* e que também foram utilizadas no *Experimento 2*, conforme apresentado na Tabela 10. Esta tabela organiza os modelos em ordem decrescente de acordo com o maior *F1-Score* obtido. No entanto, no *Experimento 3*, utilizou-se a combinação do *primeiro* e *segundo conjunto de dados*, conforme descrito na subseção 7.1.1 - Conjunto de Dados. O objetivo foi simular o fluxo inicial (Fase Online 1 - utilizado no Experimento 1) juntamente com o novo fluxo (Fase Online 2 - utilizado no Experimento 2). A fase offline empregada para gerar o modelo permaneceu a mesma do *primeiro conjunto de dados*, correspondente a 10% do total desse conjunto de dados. Os outros 90% do *primeiro conjunto de dados* e o *segundo conjunto de dados* foram unificados para formar uma nova fase online.

Na Tabela 34 são apresentadas as combinações de atributos e as configurações de hiperparâmetros utilizadas no *Experimento 3*. A numeração das matrizes de confusão foi mantida conforme os 10 melhores resultados dos *Experimento 1* e *Experimento 2*, para facilitar a comparação entre os resultados obtidos.

Assim como nos experimentos anteriores, a matriz de confusão foi ajustada para a avaliação das métricas de precisão, recall e F1-Score, com a realocação dos valores da

coluna “Unk” para a coluna correspondente à ação “Drop” (C2). Dessa forma, todos os exemplos classificados como desconhecidos foram tratados como “Drop” (descartados), resultando em uma matriz de confusão quadrada. Este ajuste permite uma avaliação mais precisa das métricas de desempenho e elimina o tráfego de rede para o qual o modelo não possui uma ação definida, ou seja, o tráfego desconhecido. É importante ressaltar que, na avaliação, nenhum dos modelos selecionados nos experimentos como os mais equilibrados para FCDs apresentou uma alta taxa de desconhecidos, com valores variando por classe de 0% a um máximo de 0,93%.

Tabela 34 – Configurações de hiperparâmetros para diferentes conjuntos de atributos no Experimento 3.

Matriz de Confusão	**Conjunto de Atributos**	**Threshold**	**threshold Forget Past**	**n Micro**	**minEx Cluster**
4	Todos os Atributos Importantes	6.0	100	11	100
7	Combinação-10	100.0	10000	100	200
8	Combinação-10	100.0	1000	30	200
10	Combinação-12	6.0	1000	17	500
23	Todos os Atributos Importantes Sem IPs	6.0	100000	3	1000
28	Todos os Atributos Importantes Sem IPs e com Combinação	6.0	1000	3	1000
29	Todos os Atributos Importantes Sem IPs e com Combinação	6.0	900	3	1000
30	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	6.0	900	117	1000
32	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	6.0	10000	3	1000
33	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	6.0	1000	3	1000

Fonte: Elaborado pelo autor.

Os resultados do *Experimento 3* foram obtidos com base nas combinações de atributos e configurações de hiperparâmetros detalhadas na Tabela 34. Para cada conjunto de atributos e hiperparâmetros, a fase offline utilizou 10% do *primeiro conjunto de dados* para o treinamento do modelo. A fase online foi realizada com a unificação de 90% do *primeiro conjunto de dados* com o *segundo conjunto de dados*, simulando assim um fluxo do FCDs. Esta nova fase online integrou os dados utilizados nos *Experimentos 1* e *2*.

Os resultados das métricas de desempenho para cada combinação estão apresentados na Tabela 35. Esta tabela exibe, para cada métrica, o resultado obtido no *Experimento 1* seguido pelo resultado correspondente no *Experimento 3*. As linhas destacadas em verde indicam onde os modelos do *Experimento 3* apresentaram desempenho superior em comparação aos modelos do *Experimento 1* nas métricas de acurácia, precisão, recall e F1-score.

Assim como observado no *Experimento 2*, o conjunto de atributos “Todos os Atributos Importantes - Matriz de Confusão 4” da Tabela 34 gerou um número significativo de

Tabela 35 – Resultados das métricas de desempenho no Experimento 1/Experimento 3 em (%).

Matriz de Confusão	Média Acerto Classes	Acurácia	Precisão	Recall	F1-Score	Acerto C3 RESET-BOTH
7	52,49/53,02	78,09/78,78	89,01/89,42	78,16/78,85	81,46/82,05	0/0
8	48,06/48,84	76,26/69,50	82,38/81,27	76,68/69,50	79,12/73,84	0/0
10	50,75/48,36	75,58/72,39	80,43/78,70	76,58/72,39	76,89/73,63	0,2/0,38
23	76,51/75,68	75,57/73,82	79,8/77,63	75,57/73,82	77,17/75,25	95,44/96,37
28	76,13/68,83	73,78/75,46	80,11/80,41	73,78/75,46	76,21/77,43	95,64/64,05
29	68,78/60,16	74,62/76,57	80,15/80,47	74,62/76,57	76,73/78,08	64,6/27,20
30	51,31/48,97	74,75/68,24	84,55/84,49	74,93/68,48	77,94/74,00	0/0
32	76,35/75,56	75,22/73,58	79,61/77,39	75,23/73,58	76,88/75,01	95,44/96,37
33	76,96/69,59	75,57/77,16	80,12/80,48	75,57/77,16	77,26/78,41	95,64/64,09

Fonte: Elaborado pelo autor.

classes novidade, mesmo com o hiperparâmetro *Threshold* configurado para um valor alto.

Essa discrepância pode ser atribuída a diferenças na natureza dos fluxos de dados entre os dois conjuntos de dados. No *primeiro conjunto* de dados, o tráfego de rede pode ter apresentado características mais homogêneas e mais próximas do conjunto inicial de treinamento, o que permitiu ao modelo identificar extensões de conceitos conhecidos. Já no *segundo conjunto* de dados, os *logs* foram coletados 15 dias após o primeiro, o que pode ter levado a uma variabilidade ou mudança nos padrões de tráfego muito maior, com esse conjunto de atributos em específico. Isso pode ter dificultado a identificação de extensões de conceitos conhecidos e levado à geração de classes novidade.

O modelo, no entanto, deveria ter identificado extensões de um conceito conhecido, dado que o número de classes é fixo, conforme demonstrado pelos quatro resultados obtidos com esse conjunto de atributos no *Experimento 1* (Matrizes de Confusão: 3, 4, 5 e 6 na Tabela 10). Devido a esse comportamento inesperado, esse conjunto de atributos foi excluído dos resultados apresentados no *Experimento 3*.

Para os demais nove experimentos, serão apresentadas tanto a matriz de confusão percentual quanto a tradicional, conforme mostrado nas Tabelas 36 a 44. Essas matrizes oferecem uma visualização mais detalhada e outra compreensão dos resultados obtidos.

Além da apresentação das 9 matrizes de confusão, na Tabela 45 são apresentados os resultados dos Erros de Classificação Críticos para esses modelos. Essa métrica é especialmente relevante, pois valores menores indicam menor confusão dos modelos na distinção entre ações de permissão (C0 - Allow) e bloqueio (C1 - Deny, C2 - Drop e C3 - Reset-Both) de tráfego de rede, refletindo maior precisão na classificação.

Tabela 36 – Experimento 3 - Matriz de Confusão 7 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	97.04%	0.61%	1.13%	0.00%	1.22%
C 1	0.43%	57.63%	40.99%	0.00%	0.96%
C 2	8.15%	33.35%	57.44%	0.00%	1.06%
C 3	69.34%	0.00%	0.00%	0.00%	30.66%

	C 0	C 1	C 2	C 3	Unk
C 0	741960	4648	8671	0	9320
C 1	2363	319544	227306	0	5300
C 2	7832	32036	55181	0	1016
C 3	1624	0	0	0	718

Fonte: Elaborado pelo autor.

Tabela 37 – Experimento 3 - Matriz de Confusão 8 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	85.33%	9.76%	4.78%	0.00%	0.14%
C 1	1.66%	51.21%	47.13%	0.00%	0.01%
C 2	6.44%	42.70%	50.83%	0.00%	0.02%
C 3	92.53%	3.03%	4.36%	0.00%	0.09%

	C 0	C 1	C 2	C 3	Unk
C 0	652423	74587	36526	0	1063
C 1	9181	283954	261329	0	49
C 2	6190	41020	48834	0	21
C 3	2167	71	102	0	2

Fonte: Elaborado pelo autor.

Tabela 38 – Experimento 3 - Matriz de Confusão 10 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	67.79%	27.43%	2.50%	1.73%	0.54%
C 1	3.73%	84.50%	11.56%	0.01%	0.21%
C 2	0.25%	58.96%	40.78%	0.00%	0.01%
C 3	61.53%	29.50%	0.00%	0.38%	8.58%

	C 0	C 1	C 2	C 3	Unk
C 0	518355	209732	19153	13221	4138
C 1	20667	468577	64079	41	1149
C 2	239	56636	39179	3	8
C 3	1441	691	0	9	201

Fonte: Elaborado pelo autor.

Tabela 39 – Experimento 3 - Matriz de Confusão 23 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	80.24%	18.00%	0.22%	1.31%	0.23%
C 1	10.81%	67.51%	21.64%	0.01%	0.03%
C 2	9.15%	32.16%	58.60%	0.02%	0.08%
C 3	3.63%	0.00%	0.00%	96.37%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	613515	137648	1680	10032	1724
C 1	59943	374337	120001	60	172
C 2	8786	30891	56294	15	79
C 3	85	0	0	2257	0

Fonte: Elaborado pelo autor.

Tabela 40 – Experimento 3 - Matriz de Confusão 28 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	82.78%	12.67%	0.24%	4.07%	0.24%
C 1	9.99%	68.02%	21.96%	0.02%	0.01%
C 2	6.98%	32.56%	60.45%	0.01%	0.01%
C 3	35.95%	0.00%	0.00%	64.05%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	632963	96873	1811	31128	1824
C 1	55390	377161	121753	127	82
C 2	6704	31274	58071	9	7
C 3	842	0	0	1500	0

Fonte: Elaborado pelo autor.

Tabela 41 – Experimento 3 - Matriz de Confusão 29 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	84.94%	12.67%	0.24%	1.94%	0.21%
C 1	9.99%	68.02%	21.96%	0.01%	0.01%
C 2	6.95%	32.56%	60.47%	0.01%	0.01%
C 3	72.80%	0.00%	0.00%	27.20%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	649477	96876	1812	14812	1622
C 1	55409	377184	121770	69	81
C 2	6680	31280	58095	5	5
C 3	1705	0	0	637	0

Fonte: Elaborado pelo autor.

Tabela 42 – Experimento 3 - Matriz de Confusão 30 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	77.91%	5.35%	5.96%	0.00%	10.78%
C 1	4.19%	56.35%	37.22%	0.00%	2.24%
C 2	11.07%	23.77%	61.62%	0.00%	3.54%
C 3	36.59%	0.04%	0.90%	0.00%	62.47%

	C 0	C 1	C 2	C 3	Unk
C 0	595662	40904	45589	0	82444
C 1	23238	312476	206403	0	12396
C 2	10636	22834	59191	0	3404
C 3	857	1	21	0	1463

Fonte: Elaborado pelo autor.

Tabela 43 – Experimento 3 - Matriz de Confusão 32 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	79.83%	18.47%	0.22%	1.31%	0.18%
C 1	10.88%	67.47%	21.62%	0.01%	0.02%
C 2	9.19%	32.15%	58.58%	0.02%	0.07%
C 3	3.63%	0.00%	0.00%	96.37%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	610345	141241	1679	9981	1353
C 1	60321	374121	119897	49	125
C 2	8824	30887	56275	15	64
C 3	85	0	0	2257	0

Fonte: Elaborado pelo autor.

Tabela 44 – Experimento 3 - Matriz de Confusão 33 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3	Unk
C 0	85.95%	12.88%	0.21%	0.94%	0.01%
C 1	10.07%	68.01%	21.91%	0.01%	0.01%
C 2	7.12%	32.56%	60.31%	0.01%	0.00%
C 3	35.91%	0.00%	0.00%	64.09%	0.00%

	C 0	C 1	C 2	C 3	Unk
C 0	657141	98518	1643	7218	79
C 1	55839	377132	121473	39	30
C 2	6836	31274	57940	12	3
C 3	841	0	0	1501	0

Fonte: Elaborado pelo autor.

Tabela 45 – Resultados da métrica de desempenho “Erros de Classificação Críticos” no Experimento 3

Matriz de Confusão	**Erro Permitir X Bloquear (%)**
7	2,43
8	9,17
10	18,98
23	15,54
28	13,75
29	12,64
30	14,39
32	15,79
33	12,08

Fonte: Elaborado pelo autor.

7.3.4.1 Conclusões e Discussões dos Resultados do Experimento 3

Os resultados do *Experimento 3* foram apresentados na Tabela 35, que exibe as métricas de desempenho para diferentes combinações de atributos e configurações de hiperparâmetros utilizadas no algoritmo MINAS. O objetivo principal deste experimento foi avaliar as mesmas configurações que geraram os melhores resultados nos *Experimentos 1* e *2*, mas utilizando uma fase online unificada, composta por 90% do *primeiro conjunto de dados* e o *segundo conjunto de dados*, simulando um novo fluxo.

Este novo cenário teve como objetivo simular um ambiente de FCDs, proporcionando uma análise mais abrangente da eficácia das configurações testadas. A proposta foi observar o comportamento dos modelos diante de fluxos de dados distintos e verificar se as métricas do primeiro fluxo melhoram ou pioram em relação ao segundo. Como o algoritmo é incremental, o modelo se ajusta continuamente à medida que o fluxo de dados avança, permitindo analisar como ele se adapta ao longo do tempo e como isso influencia os resultados obtidos. Dessa forma, é possível avaliar o desempenho dos modelos em três cenários distintos: (1) aplicação do primeiro conjunto de dados, utilizando 10% para o treinamento inicial e 90% para simulação do fluxo (Experimento 1); (2) aplicação direta do segundo conjunto de dados após uma fase de treinamento com 10% do primeiro conjunto (Experimento 2); e (3) atualização incremental do modelo ao longo do primeiro fluxo, utilizando o modelo já adaptado ao primeiro fluxo para realizar previsões do segundo (Experimento 3). O estudo permite analisar como o modelo, após o treinamento inicial, responde à introdução contínua de novos exemplos do primeiro fluxo e como ele se adapta e evolui ao lidar com o segundo fluxo de dados.

Desempenho Geral e Critérios de Avaliação

Os resultados foram avaliados com base nas mesmas métricas de desempenho dos experimentos anteriores: acurácia, precisão, *recall* e *F1-Score*, além da média de acerto das classes e do percentual de acerto da classe "C3 - Reset-Both". Como observado nos experimentos anteriores, devido ao desbalanceamento dos dados e à baixa frequência da classe "C3 - Reset-Both", o *F1-Score* continua sendo a métrica mais representativa, pois combina a precisão e o *recall*, oferecendo uma medida mais equilibrada do desempenho preditivo.

Erros de Classificação Críticos

A seleção dos modelos mais apropriados no *Experimento 3* também levou em consideração a análise dos erros de classificação críticos:

- **Falsos Negativos para "Allow (C0)":** Erros em que o tráfego permitido (C0) é incorretamente classificado como uma ação de bloqueio ("C1 - Deny", "C2 - Drop" ou "C3 - Reset-Both"). Esses erros podem resultar na interrupção de tráfego legítimo e são indesejáveis em um ambiente de rede;
- **Falsos Positivos para "Allow (C0)":** Casos em que o tráfego que deveria ser bloqueado (C1, C2, C3) é erroneamente classificado como permitido (C0). Esses erros são especialmente perigosos, pois podem representar falhas de segurança que permitem o tráfego potencialmente perigoso.

Análise dos Melhores Modelos

Para a seleção dos melhores modelos, foi adotada uma abordagem equilibrada de análise das métricas de desempenho. Em vez de priorizar modelos que apresentaram resultados excelentes em uma métrica isolada, mas fracos em outras, buscou-se identificar aqueles com desempenho consistente em múltiplas métricas. Essa escolha visa garantir modelos mais robustos e confiáveis no contexto de FCDs.

Com base nos critérios mencionados, os três modelos mais adequados do *Experimento 3*, considerando a adaptabilidade a mudanças de fluxo, a minimização dos erros críticos de classificação e o desempenho geral em termos de F1-Score e acertos por classes, são apresentados a seguir:

1. **Matriz de Confusão 23 - Tabela 35 (F1-Score: 75,25%, Acerto C3 - Reset-Both: 96,37%):** Este modelo apresentou um bom desempenho para a classe "C3 - Reset-Both". Embora o F1-Score geral (75,25%) tenha sido ligeiramente inferior ao do modelo anterior (Experimento 1, com 77,17%), ele ainda é considerado

satisfatório comparado aos outros modelos do experimento. Ele também apresentou a melhor taxa média de acerto nas classes, com 75,68%. Os erros de classificação críticos foram: "C0 - Allow" foi erroneamente classificado como "C1 - Deny" em 18,00% dos casos, como "C2 - Drop" em apenas 0,22%, e como "C3 - Reset-Both" em 1,31%. As classes de bloqueio (C1, C2, C3) foram classificadas como "C0 - Allow" com uma taxa de erro (entre 3,63% e 10,81%);

2. **Matriz de Confusão 33 - Tabela 35 (F1-Score: 78,41%, Acerto C3 - Reset-Both: 64,09%):** Este modelo foi capaz de manter um bom desempenho geral enquanto proporcionava uma taxa de erro relativamente baixa para a classe "C3 - Reset-Both". O ligeiro aumento no *F1-Score* (78,41%) em comparação com o *Experimento 1* (77,26%) indica que o modelo conseguiu se adaptar melhor às novas condições de dados unificados. Além de alcançar o segundo melhor desempenho na métrica F1-Score, o modelo registrou a terceira melhor taxa média de acerto nas classes, com 69,59%. Os erros de classificação de "C0 - Allow" como "C1 - Deny", "C2 - Drop" e "C3 - Reset-Both" foram 12,88%, 0,21% e 0,94%, respectivamente. Os erros de "C1 - Deny" como "C0 - Allow" foram de 10,07%, "C2 - Drop" como "C0 - Allow" foram de 7,12%, e "C3 - Reset-Both" como "C0 - Allow" foram de 35,91%;

3. **Matriz de Confusão 32 - Tabela 35 (F1-Score: 75,01%, Acerto C3 - Reset-Both: 96,37%):** Este modelo apresentou um bom desempenho para a classe "C3 - Reset-Both", o que é relevante para capturar padrões de tráfego menos frequentes em um cenário de FCDs. Além disso, registrou a segunda melhor taxa média de acerto nas classes, com 75,56% e apresentou um bom desempenho nas métricas de acurácia e F1-Score em comparação com os outros modelos do experimento. Os erros de classificação de "C0 - Allow" como "C1 - Deny", "C2 - Drop" e "C3 - Reset-Both" foram 18,47%, 0,22% e 1,31%, respectivamente. Adicionalmente, os erros de "C1 - Deny" como "C0 - Allow" foram de 10,88%, "C2 - Drop" como "C0 - Allow" foram de 9,19%, e "C3 - Reset-Both" como "C0 - Allow" foram de 3,63%.

Comparação de Desempenho Entre os Experimentos 1 e 3

Como observado na Tabela 35, ao comparar os resultados do *Experimento 3* com o *Experimento 1*, observa-se que alguns modelos apresentaram um desempenho superior em termos de métricas como acurácia, precisão, recall e F1-Score. Esses modelos são destacados a seguir:

- **Matriz de Confusão 7 (Experimento 3 - F1-Score: 82,05%, Acerto C3 - Reset-Both: 0%)**: Este modelo apresentou um leve aumento no F1-Score (82,5%) em comparação com o Experimento 1 (81,46%). A acurácia e precisão também

melhoraram no Experimento 3. Apesar do aumento no F1-Score (o valor mais alto obtido no Experimento 3), o acerto da classe "C3 - Reset-Both" permaneceu em 0%);

- **Matriz de Confusão 28 (Experimento 3 - F1-Score: 77,43%, Acerto C3 - Reset-Both: 64,05%)**: Similar à Matriz de Confusão 7, este modelo apresentou um F1-Score superior no Experimento 3 (77,43%) comparado ao Experimento 1 (76,21%). No entanto, houve uma queda acentuada no acerto da classe "C3 - Reset-Both" de 95,64% (Experimento 1) para 64,05% (Experimento 3);

- **Matriz de Confusão 33 (Experimento 3 - F1-Score: 78,41%, Acerto C3 - Reset-Both: 27,20%)**: A Matriz de Confusão 33 apresentou um F1-Score de (78,41%) no Experimento 3, superando seu desempenho no Experimento 1 (77,26%). Apesar da leve melhora no F1-Score geral, o acerto da classe "C3 - Reset-Both" caiu significativamente de 95,64% para 64,09%.

Conclusão

Os três modelos selecionados - Matriz de Confusão 23, 33 e 32 - demonstraram um desempenho robusto na previsão das ações corretas para os exemplos do fluxo, um equilíbrio no acerto entre as classes e na minimização de erros de classificação críticos, seguidos de perto pelas Matrizes 28 e 29. Esses modelos são, portanto, considerados os mais equilibrados para cenários de FCDs, entre todos os modelos testados no *Experimento 3*.

Se considerássemos apenas o fluxo de dados atual e o melhor desempenho de classificação — com foco em métricas como Acurácia, Precisão e F1-Score — sem levar em conta o equilíbrio entre as classes, ou a capacidade de projeção para classificações futuras, onde o tráfego pode mudar de maneira abrupta, as melhores opções para a previsão da ação recomendada no tráfego de rede seriam as Matrizes de Confusão 7, 33 e 29. Esses modelos apresentaram F1-Scores de 82,05%, 78,41% e 78,08% respectivamente, destacando-se por sua taxa de acerto elevada.

Nos experimentos, inicialmente foram filtrados os melhores modelos com base no F1-Score geral considerando o resultado geral do modelo e Acurácia. Em seguida, dentre esses modelos (10 melhores conjuntos de atributos e hiperparâmetros), foram selecionados os que apresentaram o melhor equilíbrio entre precisão e robustez, considerando não apenas a acurácia geral, mas também o acerto equilibrado entre as classes e a minimização de erros de classificação críticos. Por exemplo, a classe C3, que possui poucos exemplos, pode experimentar um aumento exponencial abrupto no número de ocorrências. Dado que as características do fluxo de dados não são constantes e o perfil das classes existentes muda ao longo do tempo, é fundamental alcançar um equilíbrio entre o desempenho geral do modelo e sua capacidade de lidar de forma eficaz com cada classe individualmente. Além

disso, é preciso que o modelo consiga distinguir o que deve ser bloqueado (classes C1, C2 e C3) e o que deve ser permitido (classe C0). Todos esses fatores foram considerados na seleção dos modelos.

É importante destacar que nenhum modelo do *Experimento 1* que obteve 0% de acerto na classe C3 foi capaz de melhorar seu desempenho nos *Experimentos 2* e *3* (outros fluxos). No entanto, alguns modelos com excelente desempenho na classe C3 do *Experimento 1* tiveram uma queda abrupta em outros fluxos.

Ao analisar os experimentos realizados com diferentes combinações de atributos, é possível observar o seguinte:

- **Combinação-10:** Demonstrou facilidade de acerto nas classes C0 e C1, mas apresentou grande dificuldade com a classe C3 (0% de acerto). Além disso, obteve resultados insatisfatórios no *Experimento 2*;
- **Combinação-12:** Obteve resultados satisfatórios nas classes C0 e C1, mas apresentou desempenho abaixo de 50% nas classes C2 e C3 no *Experimento 3*. Apesar disso, teve um bom desempenho geral nos *Experimento 1* e *3*. No entanto, teve um desempenho insatisfatório no *Experimento 2*;
- **Todos os Atributos Importantes:** Apresentou bom desempenho nas classes C0, C1, C2 e C3 no *Experimento 1*. Porém, nos *Experimento 2* e *3*, observou-se um grande número de classes novidade, tornando-o menos adequado para cenários de FCDs;
- **Todos os Atributos Importantes Sem Atributos Relacionados a Endereços IP e NAT:** Demonstrou um desempenho mais equilibrado em todas as classes, inclusive alcançando uma alta taxa de acerto na classe C3. Também apresentou resultados favoráveis nos *Experimentos 2* e *3*;
- **Todos os Atributos Importantes Sem IPs:** Também mostrou resultados equilibrados em todas as classes, com uma alta taxa de acerto na classe C3 e desempenho consistente nos Experimentos;
- **Todos os Atributos Importantes Sem IPs e com Combinação:** Esta abordagem combinada manteve um desempenho equilibrado nas C0, C1 e C2, inclusive obtendo alta taxa de acerto na classe C3 no *Experimento 1*. Além disso, apresentou bons resultados em todos os Experimentos.

Comparação de Desempenho Entre os Experimentos 2 e 3

Na Tabela 46 é apresentada a comparação de desempenho entre os Experimentos 2 e 3 para cada métrica. Os valores são exibidos na ordem: primeiro o resultado do

Experimento 2, seguido pelo correspondente do *Experimento 3*. As linhas destacadas em verde indicam os casos em que os modelos do *Experimento 3* superaram os do *Experimento 2* nas métricas de acurácia, precisão, recall e F1-score.

Tabela 46 – Resultados das métricas de desempenho no Experimento 2/Experimento 3 em (%).

Matriz de Confusão	**Média Acerto Classes**	**Acurácia**	**Precisão**	**Recall**	**F1-Score**	**Acerto C3 RESET-BOTH**
7	24,49/53,02	53,43/78,78	65,51/89,42	53,71/78,85	39,00/82,05	0/0
8	24,90/48,84	54,38/69,50	29,82/81,27	54,39/69,50	38,47/73,84	0/0
10	27,67/48,36	48,59/72,39	52,33/78,70	48,59/72,39	46,01/73,63	0/0,38
23	73,82/75,68	70,45/73,82	73,73/77,63	70,47/73,82	71,66/75,25	97,12/96,37
28	62,71/68,83	75,43/75,46	81,40/80,41	76,97/75,46	78,47/77,43	41,08/64,05
29	63,41/60,16	76,97/76,57	81,40/80,47	76,97/76,57	78,47/78,08	41,08/27,20
30	45,75/48,97	51,65/68,24	82,03/84,49	51,74/68,48	60,19/74,00	0/0
32	73,09/75,56	67,36/73,58	73,83/77,39	67,37/73,58	69,59/75,01	97,12/96,37
33	64,63/69,59	79,71/77,16	82,06/80,48	79,71/77,16	80,23/78,41	41,37/64,09

Fonte: Elaborado pelo autor.

Observa-se que a maioria dos modelos obteve um desempenho superior no Experimento 3 em relação ao Experimento 2. Nesse experimento, os modelos foram inicialmente treinados com o *primeiro conjunto de dados* e, posteriormente, atualizados incrementalmente com os dados remanescentes desse mesmo conjunto, seguidos pelo *segundo conjunto de dados*, coletado 15 dias depois. Em contraste, no Experimento 2, os modelos foram treinados apenas com o *primeiro conjunto de dados* e testados diretamente no segundo.

Mesmo nos casos em que os modelos apresentaram um desempenho superior no Experimento 2, as métricas de acurácia, precisão, recall e F1-score mostraram valores muito próximos aos do Experimento 3. No entanto, o inverso não se aplica, pois, quando o Experimento 3 obteve melhores resultados, a diferença foi mais significativa em vários modelos.

Quanto à taxa de acerto da classe *Reset-Both*, apenas a matriz de confusão 29 apresentou um desempenho significativamente superior no Experimento 2 em comparação ao Experimento 3. Por outro lado, as matrizes de confusão 28 e 33 tiveram um aumento considerável na taxa de acerto dessa classe no Experimento 3 em relação ao Experimento 2.

Apesar de todos os experimentos realizados, é necessário conduzir estudos futuros para analisar mais profundamente o comportamento desses modelos à medida que o fluxo de dados evolui com novos exemplos. Isso permitirá uma maior compreensão de sua adaptabilidade e eficácia em cenários com maior representatividade e variabilidade dos dados ao longo do tempo.

7.4 Experimentos - Clustream Adaptado

Nos experimentos com o algoritmo CluStream Adaptado, a seleção dos melhores conjuntos de atributos (Tabela 47) foi baseada nos 10 modelos considerados os mais adequados para FCDs, os quais foram gerados pelo MINAS. Esses modelos, identificados com base no F1-Score do Experimento 1, foram reaplicados nos Experimentos 2 e 3. Posteriormente, novos testes foram realizados com o CluStream Adaptado utilizando os 6 conjuntos de atributos mais promissores que geraram os 10 modelos selecionados. Para isso, foram definidas seis combinações distintas de hiperparâmetros para cada um desses conjuntos. A escolha de utilizar os melhores conjuntos de atributos avaliados pelo MINAS visa permitir uma comparação mais precisa entre os modelos, analisando como esses conjuntos se comportam em algoritmos com abordagens distintas de mineração de FCDs, mas mantendo a mesma base de dados e os mesmos conjuntos de atributos.

Tabela 47 – 6 Melhores conjuntos de Atributos - Experimentos MINAS.

Conjunto de Atributos
Combinação-10
Todos os Atributos Importantes
Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT
Todos os Atributos Importantes Sem IPs e com Combinação
Todos os Atributos Importantes Sem IPs
Combinação-12

Fonte: Elaborado pelo autor.

7.4.1 Hiperparâmetros - Clustream Adaptado

Conforme explicado anteriormente, devido às particularidades do algoritmo CluStream, foi necessário adaptar sua implementação para realizar a previsão da ação recomendada para o tráfego de rede. Nos experimentos, foi empregada a versão modificada do algoritmo, conforme detalhado no Capítulo 6, Seção 6.3.2 - Adaptação do CluStream. Diferentes configurações de hiperparâmetros influenciam diretamente o desempenho preditivo do algoritmo. Foram testadas diversas combinações de hiperparâmetros para cada um dos seis melhores conjuntos de atributos, culminando em seis configurações padrão definidas para a classificação da ação recomendada para o tráfego de rede. Na Tabela 48 são apresentados os hiperparâmetros do CluStream Adaptado e suas respectivas descrições.

7.4.2 Experimento 1

Foi conduzido um primeiro experimento (*Experimento 1*) utilizando as seis melhores combinações de atributos, com ajustes nos hiperparâmetros, com o objetivo de identificar

Tabela 48 – Hiperparâmetros do algoritmo Clustream Adaptado e suas descrições.

Hiperparâmetro	**Descrição**
`n_micro_clusters`	Define o número de microgrupos a serem formados na fase offline e mantidos durante a fase online.
`fat`	Fator de alargamento usado para determinar o raio de aceitação de novos exemplos em um microgrupo.
`delta`	Limite de tempo utilizado para determinar quando um microgrupo deve ser removido com base na sua idade. Microgrupos são removidos somente quando é necessário criar novos microgrupos, pois o número total de microgrupos presentes na memória não deve ser alterado.
`n_macro_clusters`	Define o número de macrogrupos a serem formados na fase de macroagrupamento. Os macrogrupos são grupos maiores que agrupam múltiplos microgrupos com o intuito de rotular uma das classes do atributo alvo *Action* (Allow, Deny, Drop e Reset-Both).

Fonte: Elaborado pelo autor.

as configurações que proporcionassem os melhores resultados. No *Experimento 1*, foi utilizado o *primeiro conjunto de dados*, conforme descrito neste capítulo na Seção 7.1.1 - Conjunto de Dados, em que o total de dados foi dividido em 10% para a Fase Offline e 90% para a Fase Online.

Na Tabela 49, são apresentadas as seis diferentes configurações de hiperparâmetros testadas para cada um dos seis melhores conjuntos de atributos empregados no *Experimento 1*.

Os resultados do *Experimento 1* foram obtidos a partir da combinação de atributos e hiperparâmetros descritos na Tabela 49. Para cada conjunto de atributos e hiperparâmetros, a fase offline foi utilizada para o treinamento do modelo, enquanto a fase online foi empregada para simular um FCDs. Os resultados das métricas de desempenho para cada combinação estão apresentados na Tabela 50, que ordena as combinações em ordem decrescente com base no *F1-Score* mais alto.

Os resultados incluem diversas métricas de desempenho, tais como a média de acerto das classes, que é calculada pela soma das porcentagens de acerto de cada classe dividida pelo número total de classes. Também são apresentadas as métricas clássicas de avaliação: *acurácia*, *precisão*, *recall* e *F1-Score*. Além dessas, o percentual de acerto das classes C2 (*Drop*) e C3 (*Reset-Both*) é destacado. Estas duas últimas métricas são especialmente enfatizadas devido às dificuldades que alguns modelos enfrentaram ao prever corretamente as classes "Drop" e "Reset-Both". Vale ressaltar que a baixa frequência de exemplos da Classe C3 (*Reset-Both*) no conjunto de dados torna a tarefa de treinamento e classificação correta mais complexa. Os experimentos realizados com o algoritmo MINAS também demonstraram dificuldades na predição da classe C3 em alguns modelos.

Devido à grande dificuldade dos modelos em realizar predições corretas para as classes

Tabela 49 – Configurações de hiperparâmetros para diferentes conjuntos de atributos no Experimento 1 - CluStream Adaptado.

Matriz de Confusão	**Conjunto de Atributos**	**n_micro _clusters**	**fat**	**delta**	**n_macro _clusters**
1	Combinação-10	100	2.0	1000.0	10
2	Combinação-10	200	2.0	1000.0	20
3	Combinação-10	150	1.0	9000.0	15
4	Combinação-10	250	1.5	1000.0	25
5	Combinação-10	500	1.5	1000.0	140
6	Combinação-10	150	2.0	1000.0	100
7	Todos os Atributos Importantes	100	2.0	1000.0	10
8	Todos os Atributos Importantes	200	2.0	1000.0	20
9	Todos os Atributos Importantes	150	1.0	9000.0	15
10	Todos os Atributos Importantes	250	1.5	1000.0	25
11	Todos os Atributos Importantes	500	1.5	1000.0	140
12	Todos os Atributos Importantes	150	2.0	1000.0	100
13	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	100	2.0	1000.0	10
14	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	200	2.0	1000.0	20
15	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	150	1.0	9000.0	15
16	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	250	1.5	1000.0	25
17	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	500	1.5	1000.0	140
18	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	150	2.0	1000.0	100
19	Todos os Atributos Importantes Sem IPs e com Combinação	100	2.0	1000.0	10
20	Todos os Atributos Importantes Sem IPs e com Combinação	200	2.0	1000.0	20
21	Todos os Atributos Importantes Sem IPs e com Combinação	150	1.0	9000.0	15
22	Todos os Atributos Importantes Sem IPs e com Combinação	250	1.5	1000.0	25
23	Todos os Atributos Importantes Sem IPs e com Combinação	500	1.5	1000.0	140
24	Todos os Atributos Importantes Sem IPs e com Combinação	150	2.0	1000.0	100
25	Todos os Atributos Importantes Sem IPs	100	2.0	1000.0	10
26	Todos os Atributos Importantes Sem IPs	200	2.0	1000.0	20
27	Todos os Atributos Importantes Sem IPs	150	1.0	9000.0	15
28	Todos os Atributos Importantes Sem IPs	250	1.5	1000.0	25
29	Todos os Atributos Importantes Sem IPs	500	1.5	1000.0	140
30	Todos os Atributos Importantes Sem IPs	150	2.0	1000.0	100
31	Combinação-12	100	2.0	1000.0	10
32	Combinação-12	200	2.0	1000.0	20
33	Combinação-12	150	1.0	9000.0	15
34	Combinação-12	250	1.5	1000.0	25
35	Combinação-12	500	1.5	1000.0	140
36	Combinação-12	150	2.0	1000.0	100

Fonte: Elaborado pelo autor.

Tabela 50 – Resultados das métricas de desempenho no Experimento 1 em (%) - CluStream Adaptado.

Matriz de Confusão	**Média Acerto Classes**	**Acurácia**	**Precisão**	**Recall**	**F1-Score**	**Acerto C2 Drop**	**Acerto C3 RESET-BOTH**
5	51,35	92,10	90,96	92,10	89,67	8,01	0
32	56,86	90,33	85,03	90,33	87,27	0	32,35
34	56,81	90,27	84,99	90,27	87,21	0	32,25
33	53,22	90,18	84,92	90,18	87,12	0	18,05
23	59,46	89,41	86,34	89,42	86,68	2,61	43
18	62,56	87,75	85,92	87,75	86,37	14,77	48,86
17	59,46	89,49	83,56	89,49	86,33	0	45,33
29	58,65	87,15	86,11	87,15	85,79	14,30	33,77
2	47,59	88,17	82,54	88,17	85,05	0	0
6	52,20	85,95	86,38	85,95	84,43	13,73	10,35
11	58,37	87,19	81,54	87,19	84,13	0	45,64
30	58,53	86,64	81,05	86,64	83,60	0	47,36
35	55,90	84,74	78,98	84,74	81,50	0	43,61
20	46,66	83,96	80,33	83,96	81,24	0	0
24	57,24	83,85	80,25	83,85	81,13	0	46,45
14	45,59	83,73	80,41	83,73	81,05	0	0
10	45,51	83,67	80,09	83,67	80,96	0	0
7	45,39	83,27	80,25	83,27	80,63	0	0
8	45,37	83,25	80,15	83,25	80,60	0	0
22	45,27	83,25	79,60	83,25	80,55	0	0
26	45,21	83,16	79,46	83,16	80,46	0	0
16	45,19	83,12	79,44	83,12	80,42	0	0
19	45,15	83,08	79,34	83,08	80,37	0	0
27	45,13	83,04	79,30	83,04	80,34	0	0
21	62,75	77,44	83,57	77,44	79,89	51,19	42,80
28	44,81	82,54	78,61	82,54	79,84	0	0
12	47,87	78,24	77,46	78,24	76,95	11,46	0
4	42,94	80,08	76,00	80,08	76,62	0	0
13	41,68	78,32	72,67	78,32	75,31	0	0
25	41,65	78,26	72,60	78,26	75,25	0	0
31	43,31	75,03	74,53	75,03	71,13	0,00	18,56
9	37,80	71,09	65,87	71,09	68,22	0	0
36	45,38	63,84	59,77	63,84	61,59	0	43,61
1	31,79	63,08	58,63	63,08	57,82	0	0
3	31,40	63,60	66,35	63,60	55,96	0	0
15	27,88	56,91	51,20	56,91	49,25	0	0

Fonte: Elaborado pelo autor.

C2 (Drop) e C3 (Reset-Both), foram selecionados os 11 melhores resultados da Tabela 50, com base no maior desempenho em termos de F1-Score na classificação da ação recomendada para o tráfego de rede no *Experimento 1*. Além disso, foi escolhido o modelo que obteve o vigésimo quinto melhor F1-Score (Matriz 21), que, embora não estivesse entre os melhores resultados gerais, apresentou uma taxa de acerto superior para as classes C2 e C3 em comparação aos outros modelos do experimento. Para esses 12 modelos, serão apresentadas tanto a matriz de confusão percentual quanto a matriz de confusão tradicional, conforme mostrado nas Tabelas 51 a 62. Essas matrizes oferecem uma visualização mais detalhada e um outro entendimento dos resultados obtidos, mostrando as diferentes ações previstas pelo modelo: C0 (Allow), C1 (Deny), C2 (Dro) e C3 (Reset-Both).

Tabela 51 – Matriz de Confusão 5 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	97.96%	1.54%	0.51%	0.00%
C 1	0.54%	99.44%	0.03%	0.00%
C 2	3.91%	88.08%	8.01%	0.00%
C 3	100.00%	0.00%	0.00%	0.00%

	C 0	C 1	C 2	C 3
C 0	473261	7429	2441	0
C 1	1903	352303	94	0
C 2	2485	55955	5087	0
C 3	986	0	0	0

Fonte: Elaborado pelo autor.

Tabela 52 – Matriz de Confusão 32 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95,65%	4,34%	0,00%	0,02%
C 1	0,54%	99,45%	0,00%	0,00%
C 2	0,04%	99,96%	0,00%	0,00%
C 3	67,65%	0,00%	0,00%	32,35%

	C 0	C 1	C 2	C 3
C 0	462092	20946	0	93
C 1	1926	352359	0	15
C 2	24	63503	0	0
C 3	667	0	0	319

Fonte: Elaborado pelo autor.

Tabela 53 – Matriz de Confusão 34 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95,52%	4,46%	0,00%	0,02%
C 1	0,55%	99,45%	0,00%	0,00%
C 2	0,04%	99,96%	0,00%	0,00%
C 3	67,75%	0,00%	0,00%	32,25%

	C 0	C 1	C 2	C 3
C 0	461494	21544	0	93
C 1	1950	352336	0	14
C 2	24	63503	0	0
C 3	668	0	0	318

Fonte: Elaborado pelo autor.

Além da apresentação das matrizes de confusão dos 11 melhores resultados da Tabela 50 - selecionados com base no maior desempenho em termos de (*F1-Score*) na classificação da ação recomendada para o tráfego de rede no *Experimento 1* — e do modelo que alcançou o vigésimo quinto melhor *F1-Score* (Matriz 21), incluído por apresentar uma

Tabela 54 – Matriz de Confusão 33 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95,39%	4,60%	0,00%	0,01%
C 1	0,54%	99,45%	0,00%	0,00%
C 2	0,04%	99,96%	0,00%	0,00%
C 3	81,95%	0,00%	0,00%	18,05%

	C 0	C 1	C 2	C 3
C 0	460850	22239	0	42
C 1	1921	352365	0	14
C 2	24	63503	0	0
C 3	808	0	0	178

Fonte: Elaborado pelo autor.

Tabela 55 – Matriz de Confusão 23 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95.72%	4.20%	0.08%	0.00%
C 1	2.70%	96.50%	0.80%	0.00%
C 2	12.93%	84.46%	2.61%	0.00%
C 3	56.99%	0.00%	0.00%	43.00%

	C 0	C 1	C 2	C 3
C 0	462450	20294	368	19
C 1	9566	341915	2819	0
C 2	8215	53652	1660	0
C 3	562	0	0	424

Fonte: Elaborado pelo autor.

Tabela 56 – Matriz de Confusão 18 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	93.52%	4.88%	1.59%	0.33%
C 1	4.50%	93.08%	2.43%	0.00%
C 2	12.46%	72.78%	14.77%	0.00%
C 3	53.14%	0.00%	0.00%	46.86%

	C 0	C 1	C 2	C 3
C 0	451848	23598	7669	16
C 1	15931	329775	8594	0
C 2	7913	46232	9382	0
C 3	524	0	0	462

Fonte: Elaborado pelo autor.

Tabela 57 – Matriz de Confusão 17 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	96.77%	3.22%	0.00%	0.25%
C 1	4.27%	95.73%	0.00%	0.00%
C 2	13.19%	86.81%	0.00%	0.00%
C 3	54.67%	0.00%	0.00%	45.33%

	C 0	C 1	C 2	C 3
C 0	467543	15576	0	12
C 1	15113	339187	0	0
C 2	8381	55146	0	0
C 3	539	0	0	447

Fonte: Elaborado pelo autor.

Tabela 58 – Matriz de Confusão 29 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	89.87%	8.33%	1.80%	0.21%
C 1	1.77%	96.65%	1.57%	0.00%
C 2	7.89%	77.81%	14.30%	0.00%
C 3	63.99%	2.23%	0.00%	33.77%

	C 0	C 1	C 2	C 3
C 0	434183	40238	8700	10
C 1	6278	342442	5580	0
C 2	5013	49431	9083	0
C 3	631	22	0	333

Fonte: Elaborado pelo autor.

Tabela 59 – Matriz de Confusão 2 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	93.68%	6.32%	0.00%	0.00%
C 1	3.29%	96.71%	0.00%	0.00%
C 2	16.57%	83.43%	0.00%	0.00%
C 3	100.00%	0.00%	0.00%	0.00%

	C 0	C 1	C 2	C 3
C 0	452584	30547	0	0
C 1	11674	342626	0	0
C 2	10524	53003	0	0
C 3	986	0	0	0

Fonte: Elaborado pelo autor.

Tabela 60 – Matriz de Confusão 6 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	86.91%	12.14%	0.93%	0.02%
C 1	1.75%	97.80%	0.46%	0.00%
C 2	2.39%	83.89%	13.73%	0.00%
C 3	89.66%	0.00%	0.00%	10.35%

	C 0	C 1	C 2	C 3
C 0	419894	58634	4508	95
C 1	6191	346493	1616	0
C 2	1516	53291	8720	0
C 3	884	0	0	102

Fonte: Elaborado pelo autor.

Tabela 61 – Matriz de Confusão 11 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	93.46%	6.54%	0.00%	0.39%
C 1	5.61%	94.39%	0.00%	0.00%
C 2	19.17%	80.83%	0.00%	0.00%
C 3	53.96%	0.41%	0.00%	45.64%

	C 0	C 1	C 2	C 3
C 0	451512	31600	0	19
C 1	19874	334426	0	0
C 2	12179	51348	0	0
C 3	532	4	0	450

Fonte: Elaborado pelo autor.

Tabela 62 – Matriz de Confusão 21 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	84.78%	9.17%	5.97%	0.08%
C 1	4.04%	72.23%	22.98%	0.75%
C 2	4.79%	44.01%	51.19%	0.00%
C 3	57.20%	0.00%	0.00%	42.80%

	C 0	C 1	C 2	C 3
C 0	409592	44319	28852	368
C 1	14300	255912	81415	2673
C 2	3043	27959	32522	3
C 3	564	0	0	422

Fonte: Elaborado pelo autor.

taxa de acerto superior para as classes C2 e C3 em comparação aos demais modelos do experimento, na Tabela 63 são apresentados os resultados dos Erros de Classificação Críticos para esses 12 modelos. Essa métrica é especialmente relevante, pois valores menores indicam menor confusão dos modelos na distinção entre ações de permissão (C0 - Allow) e bloqueio (C1 - Deny, C2 - Drop e C3 - Reset-Both) de tráfego de rede, refletindo maior precisão na classificação.

Tabela 63 – Resultados da métrica de desempenho "Erros de Classificação Críticos" no Experimento 1 - CluStream Adaptado

Matriz de Confusão	**Erro Permitir X Bloquear (%)**
5	1,69
32	2,62
34	2,69
33	2,78
23	4,33
18	6,17
17	4,39
29	6,75
2	5,96
6	7,96
11	7,12
21	10,14

Fonte: Elaborado pelo autor.

7.4.2.1 Conclusões e Discussões dos Resultados do Experimento 1

Os resultados do *Experimento 1* são apresentados na Tabela 50, que mostra as diferentes combinações de atributos e configurações de hiperparâmetros utilizadas no algoritmo CluStream Adaptado. O objetivo principal deste experimento foi identificar as configurações que proporcionaram os melhores resultados na classificação da ação recomendada para o tráfego de rede, com base nos *logs* gerados pelo *firewall* Palo Alto. É importante destacar que o tempo de execução dos experimentos utilizando o algoritmo MINAS é significativamente menor em comparação com o tempo de execução do CluStream adaptado. Além disso, observa-se que o aumento no número de microgrupos especificado pelo hiperparâmetro *n_micro_clusters* (que define o número de microgrupos a serem formados na fase offline e mantidos durante a fase online) resulta em um aumento acentuado no tempo de execução dos experimentos.

Desempenho Geral e Critérios de Avaliação

Os resultados foram avaliados com base em métricas clássicas de desempenho, incluindo acurácia, precisão, recall, e F1-Score, além da média de acerto das classes e do

percentual de acerto das classes “C2 - Drop” e “C3 - Reset-Both”. Devido ao desbalanceamento dos dados e à baixa frequência da classe “C3 - Reset-Both”, o F1-Score se torna uma métrica mais representativa, pois combina a precisão e o recall, oferecendo uma medida mais equilibrada do desempenho preditivo.

Outro fator a ser analisado é a capacidade dos modelos de se adaptarem a mudanças abruptas no fluxo de tráfego, que são comuns em cenários de FCDs. Em tais condições, o modelo precisa ser suficientemente robusto para capturar variações inesperadas nos padrões de tráfego de rede, especialmente para a classe rara “C3 - Reset-Both”. Por isso, foi dada especial atenção aos modelos que evitaram alcançar 0% de acerto na classe “C3 - Reset-Both” e que mantiveram um equilíbrio satisfatório entre todas as classes. Além disso, considerando que o algoritmo enfrentou dificuldades em classificar corretamente os exemplos da classe “C2 - Drop”, também foi feita uma análise desse aspecto.

Erros de Classificação Críticos

Uma consideração importante na seleção dos modelos mais apropriados foi a análise dos erros de classificação críticos:

- **Falsos negativos para “Allow (C0)”**: Casos em que o tráfego permitido (C0 - Allow) é incorretamente classificado como uma ação de bloqueio (“C1 - Deny”, “C2 - Drop” ou “C3 - Reset-Both”). Esses erros podem resultar na interrupção de tráfego legítimo e são indesejáveis em um ambiente de rede;
- **Falsos positivos para “Allow (C0)”**: Casos em que o tráfego que deveria ser bloqueado (“C1 - Deny”, “C2 - Drop”, “C3 - Reset-Both”) é erroneamente classificado como permitido (C0 - Allow). Esses erros são especialmente perigosos, pois podem representar falhas de segurança que podem permitir o tráfego potencialmente perigoso.

Análise dos Melhores Modelos

Para a seleção dos melhores modelos, foi adotada uma abordagem equilibrada de análise das métricas de desempenho. Em vez de priorizar modelos que apresentaram resultados excelentes em uma métrica isolada, mas fracos em outras, buscou-se identificar aqueles com desempenho consistente em múltiplas métricas. Essa escolha visa garantir modelos mais robustos e confiáveis no contexto de FCDs.

Os experimentos realizados evidenciaram que os modelos testados com o algoritmo CluStream Adaptado enfrentam grandes dificuldades na predição correta das classes C2 (Drop) e C3 (Reset-Both). Apenas a *Matriz 21* apresentou uma taxa de acerto superior a 50% na classe C2 (51,19%) e 42,80% na classe C3, enquanto nenhum modelo conseguiu ultrapassar essa marca (50%) na classe C3.

Além disso, os resultados mostram que os modelos frequentemente confundem exemplos da classe C2 (Drop) com a classe C1 (Deny). Embora essa confusão represente um erro, ela é menos grave, pois são ações diferentes para bloquear um tráfego de rede. Por outro lado, a classe C3 (Reset-Both), que possui um número limitado de exemplos para treinamento e avaliação, é frequentemente confundida com a classe C0 (Allow). Essa confusão é mais preocupante, pois permite o tráfego que deveria ser bloqueado, aumentando o risco de falhas de segurança.

Com base nos critérios mencionados, os três modelos mais adequados, considerando a adaptabilidade a mudanças de fluxo, a minimização dos erros críticos de classificação e o desempenho geral em termos de F1-Score e acertos por classes, são apresentados a seguir:

1. **Matriz de Confusão 18 - Tabela 50** (F1-Score: 86,37%, Acerto C2 - Drop: 14,77%, Acerto C3 - Reset-Both: 48,06%): Este modelo destacou-se pelo seu alto valor de F1-Score e acurácia. Além disso, apresentou uma boa taxa média de acerto de 62,56% nas classes, superando a maioria dos modelos comparados. Os erros de classificação críticos foram minimizados: "C0 - Allow" foi erroneamente classificado como "C1 - Deny" em 4,88% dos casos, como "C2 - Drop" em apenas 1,59%, e como "C3 - Reset-Both" em 0,25%. Por outro lado, as classes de bloqueio ("C1 - Deny", "C2 - Drop", "C3 - Reset-Both") foram classificadas como "C0 - Allow" com uma taxa de erro entre 4,50% e 53,14%. Esses resultados indicam um bom desempenho em relação aos outros modelos testados;

2. **Matriz de Confusão 23 - Tabela 50** (F1-Score: 86,68%, Acerto C2 - Drop: 2,61%, Acerto C3 - Reset-Both: 43,00%): Este modelo também se destacou pelo seu alto valor de F1-Score e acurácia. Com uma taxa média de acerto de 59,46% nas classes, ele ocupa a terceira posição entre entre os modelos testados. O erro de classificação de "C0 - Allow" para "C1 - Deny" foi de 4,20%, "C0 - Allow" para "C2 - Drop" foi de 0,08%, e "C0 - Allow" para "C3 - Reset-Both" foi de 0%. Enquanto isso, o erro de classificação de "C1 - Deny" como "C0 - Allow" foi de 2,70%, "C2 - Drop" como "C0 - Allow" foi de 12,93%, e "C3 - Reset-Both" como "C0 - Allow" foi de 56,99%. Comparado aos outros modelos, este apresenta um bom desempenho na predição geral e na média de acerto nas classes, além de confundir pouco as ações de permissão e bloqueio;

3. **Matriz de Confusão 21 - Tabela 50** (F1-Score: 79,89%, Acerto C2 - Drop: 51,19%, Acerto C3 - Reset-Both: 42,80%): Embora este modelo tenha apresentado resultados um pouco inferiores nas métricas de F1-Score e acurácia, ele se destacou com a melhor taxa média de acerto de 62,75% nas classes, demonstrando um desempenho equilibrado. Os erros de classificação de "C0 - Allow" como "C1 - Deny",

"C2 - Drop"e "C3 - Reset-Both" são 9,17%, 5,97% e 0,08%, respectivamente. Adicionalmente, os erros de "C1 - Deny" como "C0 - Allow" foram de 4,04%, "C2 - Drop" como "C0 - Allow" foram de 4,79%, e "C3 - Reset-Both" como "C0 - Allow" foram de 57,20%.

Conclusão

Os três modelos selecionados - Matrizes de Confusão 18, 23 e 21, seguidas pela Matriz 29 - demonstraram um desempenho robusto na previsão das ações corretas para os exemplos do fluxo, um equilíbrio no acerto entre as classes e na minimização de erros de classificação críticos, sendo os mais equilibrados para cenários de FCDs entre todos os avaliados até o momento. Se considerássemos apenas o fluxo de dados atual e o melhor desempenho de classificação — com foco em métricas como Acurácia, Precisão e F1-Score — sem levar em conta o equilíbrio entre as classes, ou a capacidade de projeção para classificações futuras, onde o tráfego pode mudar de maneira abrupta, as melhores opções para a previsão da ação recomendada no tráfego de rede seriam as Matrizes de Confusão 5, 32 e 34. Esses modelos apresentaram F1-Scores de 89,67%, 87,27% e 87,21% respectivamente, destacando-se por sua taxa de acerto elevada.

7.4.3 Experimento 2

O *Experimento 2* tem como objetivo avaliar o desempenho do algoritmo CluStream Adaptado em um fluxo de dados coletado 15 dias após o *primeiro conjunto*. A proposta é analisar como o algoritmo se comporta ao processar um novo fluxo de dados, coletado em um período diferente do utilizado e testado no *Experimento 1*, o que pode refletir variações no comportamento ou nas características dos dados ao longo do tempo. Ao introduzir um novo fluxo de dados, esse experimento investiga a capacidade do modelo de generalizar para dados com características potencialmente diferentes.

No *Experimento 2*, foram utilizadas as mesmas combinações de atributos e configurações de hiperparâmetros que resultaram nos 11 melhores desempenhos do *Experimento 1*, além do modelo da *Matriz 21*, que obteve a melhor taxa média de acerto nas classes, conforme apresentado na Tabela 50. Essa tabela organiza as combinações de atributos em ordem decrescente de acordo com o maior *F1-Score* obtido. No entanto, no *Experimento 2*, foi utilizado um *segundo conjunto de dados*, mencionado na subseção 7.1.1 - Conjunto de Dados, com o intuito de simular um novo fluxo (Fase Online 2) do FCDs. A fase offline empregada para gerar o modelo permaneceu a mesma do *primeiro conjunto de dados*, correspondente a 10% do total desse conjunto de dados.

Na Tabela 64 são apresentadas as combinações de atributos e as configurações de hiperparâmetros que foram aplicadas no *Experimento 2*. A numeração das matrizes de

confusão foi mantida a mesma dos 12 melhores resultados do *Experimento 1*, a fim de facilitar a comparação entre os resultados obtidos.

Tabela 64 – Configurações de hiperparâmetros para diferentes conjuntos de atributos no Experimento 2 - CluStream Adaptado.

Matriz de Confusão	**Conjunto de Atributos**	**n_micro _clusters**	**fat**	**delta**	**n_macro _clusters**
2	Combinação-10	200	2.0	1000.0	20
5	Combinação-10	500	1.5	1000.0	140
6	Combinação-10	150	2.0	1000.0	100
11	Todos os Atributos Importantes	500	1.5	1000.0	140
17	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	500	1.5	1000.0	140
18	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	150	2.0	1000.0	100
21	Todos os Atributos Importantes Sem IPs e com Combinação	150	1.0	9000.0	15
23	Todos os Atributos Importantes Sem IPs e com Combinação	500	1.5	1000.0	140
29	Todos os Atributos Importantes Sem IPs	500	1.5	1000.0	140
32	Combinação-12	200	2.0	1000.0	20
33	Combinação-12	150	1.0	9000.0	15
34	Combinação-12	250	1.5	1000.0	25

Fonte: Elaborado pelo autor.

Os resultados do *Experimento 2* foram obtidos com base nas combinações de atributos e configurações de hiperparâmetros detalhadas na Tabela 64. Para cada conjunto de atributos e hiperparâmetros, a fase offline (utilizando 10% do primeiro conjunto de dados, o mesmo do *Experimento 1*) foi usada para o treinamento do modelo, enquanto a fase online (segundo conjunto de dados) foi aplicada para simular um novo fluxo do FCDs. Os resultados das métricas de desempenho para cada combinação estão apresentados na Tabela 65, que mostra, para cada métrica, o resultado obtido no *Experimento 1* seguido do resultado correspondente no *Experimento 2*. As linhas destacadas em verde indicam onde o modelo do *Experimento 2* apresentou desempenho superior em comparação aos modelos do *Experimento 1* nas métricas de acurácia, precisão, recall e F1-score.

Para os 12 modelos do *Experimento 2*, serão apresentadas tanto a matriz de confusão percentual quanto a tradicional, conforme mostrado nas Tabelas 66 a 77. Essas matrizes oferecem uma visualização mais detalhada e um outro entendimento dos resultados obtidos. Assim como no *Experimento 1*, evidencia-se a significativa dificuldade dos modelos em prever corretamente os exemplos da classe “C2 (Drop)”.

Além da apresentação das 12 matrizes de confusão, na Tabela 78 são apresentados os resultados dos Erros de Classificação Críticos para esses modelos. Essa métrica é especialmente relevante, pois valores menores indicam menor confusão dos modelos na

Tabela 65 – Resultados das métricas de desempenho no Experimento 1/Experimento 2 em (%) - CluStream Adaptado.

Matriz de Confusão	**Média Acerto Classes**	**Acurácia**	**Precisão**	**Recall**	**F1-Score**	**Acerto C2 Drop**	**Acerto C3 RESET-BOTH**
5	51,35/50,93	92,1/92,36	90,96/91,46	92,1/92,36	89,67/89,97	8,01/6,60	0/0
32	56,86/61,98	90,33/89,31	85,03/84,10	90,33/89,31	87,27/86,51	0/0	32,35/57,08
34	56,81/63,20	90,27/90,69	84,99/85,91	90,27/90,69	87,21/87,94	0/0	32,25/58,41
33	53,22/57,77	90,18/90,72	84,92/86	90,18/90,72	87,12/87,95	0/0	18,05/36,36
23	59,46/56,95	89,41/89,84	86,34/88,12	89,42/89,84	86,68/88,17	2,61/12,24	43/25,22
18	62,56/70,61	87,75/89,73	85,92/87,70	87,75/89,73	86,37/88,22	14,77/13,59	48,86/79,94
17	59,46/67,03	89,49/88,70	83,56/83,88	89,49/88,70	86,33/85,88	0/0	45,33/78,83
29	58,65/61,80	87,15/88,01	86,11/85,32	87,15/88,01	85,79/85,32	14,3/0	33,77/58,85
2	47,59/28,46	88,17/59,96	82,54/64,30	88,17/59,96	85,05/49,45	0/0	0/0
6	52,20/75,73	85,95/91,63	86,38/90,03	85,95/91,63	84,43/89,94	13,73/12,31	10,35/96,90
11	58,37/62,59	87,19/80,62	81,54/78,06	87,19/80,62	84,13/78,44	0/0,87	45,64/76,70
21	62,75/30,97	77,44/62,90	83,57/60,62	77,44/62,90	79,89/56,88	51,19/0	42,8/0

Fonte: Elaborado pelo autor.

Tabela 66 – Experimento 2 - Matriz de Confusão 5 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	97.71%	2.02%	0.27%	0.00%
C 1	0.58%	99.40%	0.03%	0.00%
C 2	5.03%	88.36%	6.60%	0.00%
C 3	100.00%	0.00%	0.00%	0.00%

	C 0	C 1	C 2	C 3
C 0	275022	5683	763	0
C 1	1157	199004	52	0
C 2	1638	28752	2148	0
C 3	1356	0	0	0

Fonte: Elaborado pelo autor.

Tabela 67 – Experimento 2 - Matriz de Confusão 32 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95.56%	4.43%	0.00%	0.01%
C 1	4.73%	95.27%	0.00%	0.00%
C 2	11.82%	88.18%	0.00%	0.00%
C 3	42.92%	0.00%	0.00%	57.08%

	C 0	C 1	C 2	C 3
C 0	268964	12463	0	41
C 1	9475	190738	0	0
C 2	3845	28693	0	0
C 3	582	0	0	774

Fonte: Elaborado pelo autor.

Tabela 68 – Experimento 2 - Matriz de Confusão 34 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95.48%	4.50%	0.00%	0.01%
C 1	1.09%	98.91%	0.00%	0.00%
C 2	0.06%	99.94%	0.00%	0.00%
C 3	41.59%	0.00%	0.00%	58.41%

	C 0	C 1	C 2	C 3
C 0	268755	12675	0	38
C 1	2176	198037	0	0
C 2	20	32518	0	0
C 3	564	0	0	792

Fonte: Elaborado pelo autor.

Tabela 69 – Experimento 2 - Matriz de Confusão 33 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95.30%	4.69%	0.00%	0.01%
C 1	0.60%	99.40%	0.00%	0.00%
C 2	0.02%	99.98%	0.00%	0.00%
C 3	63.64%	0.00%	0.00%	36.36%

	C 0	C 1	C 2	C 3
C 0	268233	13204	0	31
C 1	1200	199013	0	0
C 2	5	32533	0	0
C 3	863	0	0	493

Fonte: Elaborado pelo autor.

Tabela 70 – Experimento 2 - Matriz de Confusão 23 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95.80%	4.07%	0.13%	0.00%
C 1	3.42%	94.52%	2.07%	0.00%
C 2	14.31%	73.45%	12.24%	0.00%
C 3	74.78%	0.00%	0.00%	25.22%

	C 0	C 1	C 2	C 3
C 0	269655	11448	365	0
C 1	6842	189232	4139	0
C 2	4655	23900	3983	0
C 3	1014	0	0	342

Fonte: Elaborado pelo autor.

Tabela 71 – Experimento 2 - Matriz de Confusão 18 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	97.11%	2.38%	0.50%	0.01%
C 1	5.76%	91.79%	2.46%	0.00%
C 2	14.99%	71.41%	13.59%	0.00%
C 3	20.06%	0.00%	0.00%	79.94%

	C 0	C 1	C 2	C 3
C 0	273328	6697	1407	36
C 1	11527	183768	4918	0
C 2	4879	23237	4422	0
C 3	272	0	0	1084

Fonte: Elaborado pelo autor.

Tabela 72 – Experimento 2 - Matriz de Confusão 17 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95.06%	4.94%	0.00%	0.00%
C 1	5.76%	94.24%	0.00%	0.00%
C 2	24.62%	75.38%	0.00%	0.00%
C 3	21.17%	0.00%	0.00%	78.83%

	C 0	C 1	C 2	C 3
C 0	267556	13901	0	11
C 1	11539	188674	0	0
C 2	8012	24526	0	0
C 3	287	0	0	1069

Fonte: Elaborado pelo autor.

Tabela 73 – Experimento 2 - Matriz de Confusão 29 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	92.67%	7.33%	0.00%	0.00%
C 1	4.04%	95.96%	0.00%	0.00%
C 2	10.15%	89.85%	0.00%	0.00%
C 3	41.22%	0.22%	0.00%	58.55%

	C 0	C 1	C 2	C 3
C 0	260824	20636	0	8
C 1	8084	192129	0	0
C 2	3302	29236	0	0
C 3	559	3	0	794

Fonte: Elaborado pelo autor.

Tabela 74 – Experimento 2 - Matriz de Confusão 2 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	99.99%	0.01%	0.00%	0.00%
C 1	86.16%	13.84%	0.00%	0.00%
C 2	83.35%	16.65%	0.00%	0.00%
C 3	100.00%	0.00%	0.00%	0.00%

	C 0	C 1	C 2	C 3
C 0	281438	30	0	0
C 1	172502	27711	0	0
C 2	27119	5419	0	0
C 3	1356	0	0	0

Fonte: Elaborado pelo autor.

Tabela 75 – Experimento 2 - Matriz de Confusão 6 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	97.56%	1.80%	0.43%	0.21%
C 1	2.58%	96.14%	1.28%	0.00%
C 2	3.16%	84.52%	12.31%	0.00%
C 3	3.10%	0.00%	0.00%	96.90%

	C 0	C 1	C 2	C 3
C 0	274605	5076	1197	590
C 1	5166	192488	2557	2
C 2	1029	27502	4007	0
C 3	42	0	0	1314

Fonte: Elaborado pelo autor.

Tabela 76 – Experimento 2 - Matriz de Confusão 11 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	84.14%	15.82%	0.04%	0.00%
C 1	10.72%	88.66%	0.62%	0.00%
C 2	15.16%	83.97%	0.87%	0.00%
C 3	23.23%	0.07%	0.00%	76.70%

	C 0	C 1	C 2	C 3
C 0	236815	44542	100	11
C 1	21461	177518	1234	0
C 2	4934	27322	282	0
C 3	315	1	0	1040

Fonte: Elaborado pelo autor.

Tabela 77 – Experimento 2 - Matriz de Confusão 21 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	93.86%	6.14%	0.00%	0.00%
C 1	69.97%	30.03%	0.00%	0.00%
C 2	72.25%	27.75%	0.00%	0.00%
C 3	99.56%	0.44%	0.00%	0.00%

	C 0	C 1	C 2	C 3
C 0	264189	17279	0	0
C 1	140094	60119	0	0
C 2	23508	9030	0	0
C 3	1350	6	0	0

Fonte: Elaborado pelo autor.

distinção entre ações de permissão (C0 - Allow) e bloqueio (C1 - Deny, C2 - Drop e C3 - Reset-Both) de tráfego de rede, refletindo maior precisão na classificação.

Tabela 78 – Resultados da métrica de desempenho "Erros de Classificação Críticos" no Experimento 2 - CluStream Adaptado

Matriz de Confusão	**Erro Permitir X Bloquear (%)**
5	2,06
32	5,12
34	3,00
33	2,97
23	4,72
18	4,81
17	6,55
29	6,32
2	38,99
6	2,54
11	13,84
21	35,35

Fonte: Elaborado pelo autor.

7.4.3.1 Conclusões e Discussões dos Resultados do Experimento 2

Os resultados do *Experimento 2* são apresentados na Tabela 65, com diferentes combinações de atributos e configurações de hiperparâmetros utilizados no algoritmo CluStream Adaptado. O objetivo principal deste experimento foi avaliar as mesmas configurações que geraram os melhores resultados no *Experimento 1*, mas utilizando um novo fluxo de dados capturado através do *firewall* Palo Alto, simulando um novo cenário do FCDs.

Desempenho Geral e Critérios de Avaliação

Os resultados foram avaliados com base nas mesmas métricas de desempenho utilizadas no *Experimento 1*, incluindo acurácia, precisão, recall, e F1-Score, além da média de acerto das classes e do percentual de acerto das classes "C2 - Drop" e "C3 - Reset-Both". Devido ao desbalanceamento dos dados e à baixa frequência da classe "C3 - Reset-Both",

o F1-Score se torna uma métrica mais representativa, pois combina a precisão e o recall, oferecendo uma medida mais equilibrada do desempenho preditivo.

Outro fator a ser analisado é a capacidade dos modelos de se adaptarem a mudanças abruptas no fluxo de tráfego, que são comuns em cenários de FCDs. Em tais condições, o modelo precisa ser suficientemente robusto para capturar variações inesperadas nos padrões de tráfego de rede, especialmente para a classe rara "C3 - Reset-Both". Por isso, foi dada especial atenção aos modelos que evitaram alcançar 0% de acerto na classe "C3 - Reset-Both" e que mantiveram um equilíbrio satisfatório entre todas as classes. Além disso, considerando que o algoritmo enfrentou dificuldades em classificar corretamente os exemplos da classe "C2 - Drop", também foi feita uma análise desse aspecto.

Erros de Classificação Críticos

Uma consideração importante na seleção dos modelos mais apropriados foi a análise dos erros de classificação críticos:

- **Falsos negativos para "Allow (C0)"**: Casos em que tráfego permitido ("C0 - Allow") é incorretamente classificado como uma ação de bloqueio ("C1 - Deny", "C2 - Drop" ou "C3 - Reset-Both"). Esses erros podem resultar na interrupção de tráfego legítimo e são indesejáveis em um ambiente de rede;
- **Falsos positivos para "Allow (C0)"**: Casos em que tráfego que deveria ser bloqueado ("C1 - Deny", "C2 - Drop", "C3 - Reset-Both") é erroneamente classificado como permitido ("C0 - Allow"). Esses erros são especialmente perigosos, pois podem representar falhas de segurança que podem permitir o tráfego potencialmente perigoso.

Análise dos Melhores Modelos

Para a seleção dos melhores modelos, foi adotada uma abordagem equilibrada de análise das métricas de desempenho. Em vez de priorizar modelos que apresentaram resultados excelentes em uma métrica isolada, mas fracos em outras, buscou-se identificar aqueles com desempenho consistente em múltiplas métricas. Essa escolha visa garantir modelos mais robustos e confiáveis no contexto de FCDs.

Os experimentos realizados até o momento indicam que os modelos testados com o algoritmo CluStream Adaptado enfrentam desafios significativos na predição correta das classes C2 (Drop) e C3 (Reset-Both). Entretanto, a maioria dos modelos apresentou uma melhoria notável no desempenho da Classe C3 no *Experimento 2* em comparação ao *Experimento 1*. A análise da *Matriz 21* revelou, no *Experimento 1*, taxas de acerto superiores a 50% para a classe C2 (51,19%) e 42,80% para a classe C3. Contudo, no

Experimento 2, a *Matriz 21* demonstrou um desempenho insatisfatório, com ausência de predições corretas para as classes C2 (0%) e C3 (0%). Isso sugere que o modelo não conseguiu se adaptar ao novo fluxo de dados implementado na fase online.

Além disso, os resultados mostram que os modelos frequentemente confundem exemplos da classe C2 (Drop) com a classe C1 (Deny). Embora essa confusão represente um erro, ela é menos grave, pois são ações diferentes para bloquear um tráfego de rede. Por outro lado, a classe C3 (Reset-Both), que possui um número limitado de exemplos para treinamento e avaliação, é frequentemente confundida com a classe C0 (Allow). Essa confusão é mais preocupante, pois permite o tráfego que deveria ser bloqueado, aumentando o risco de falhas de segurança.

Com base nos critérios mencionados, os três modelos mais adequados, considerando a adaptabilidade a mudanças de fluxo, a minimização dos erros críticos de classificação e o desempenho geral em termos de F1-Score e acertos por classes, são apresentados a seguir:

1. **Matriz de Confusão 6 - Tabela 65** (F1-Score: 89,94%, Acerto C2 - Drop: 12,31%, Acerto C3 - Reset-Both: 96,90%): Este modelo destacou-se pelo seu alto valor de F1-Score, acurácia e taxa de acerto classe C3. Além disso, apresentou uma taxa média de acerto de 75,73% nas classes, a maior entre os modelos comparados. Os erros de classificação críticos foram minimizados: "C0 - Allow" foi erroneamente classificado como "C1 - Deny" em 1,80% dos casos, como "C2 - Drop" em apenas 0,43%, e como "C3 - Reset-Both" em 0,21%. Por outro lado, as classes de bloqueio ("C1 - Deny", "C2 - Drop", "C3 - Reset-Both") foram classificadas como "C0 - Allow" com uma taxa de erro baixa (entre 2,58% e 3,16%). Esses resultados indicam um bom desempenho em relação aos outros modelos testados;

2. **Matriz de Confusão 18 - Tabela 65** (F1-Score: 88,22%, Acerto C2 - Drop: 13,59%, Acerto C3 - Reset-Both: 79,94%): Este modelo também se destacou pelo seu alto valor de F1-Score, acurácia e taxa de acerto na classe C3. Com uma taxa média de acerto de 70,61% nas classes, ele ocupa a segunda posição entre os modelos testados. O erro de classificação de "C0 - Allow" para "C1 - Deny" foi de 2,38%, "C0 - Allow" para "C2 - Drop" foi de 0,50%, e "C0 - Allow" para "C3 - Reset-Both" foi de 0,01%. Enquanto isso, o erro de classificação de "C1 - Deny" como "C0 - Allow" foi de 5,76%, "C2 - Drop" como "C0 - Allow" foi de 14,99%, e "C3 - Reset-Both" como "C0 - Allow" foi de 20,06%. Comparado aos outros modelos, este apresenta um bom desempenho na predição geral e na média de acerto nas classes, além de confundir pouco as ações de permissão e bloqueio;

3. **Matriz de Confusão 17 - Tabela 65** (F1-Score: 85,88%, Acerto C2 - Drop: 0%, Acerto C3 - Reset-Both: 78,83%): Embora o modelo não tenha taxa de acerto na

classe C2, ele se destacou com um bom F1-Score e a terceira melhor taxa média de acerto de 67,03% nas classes, demonstrando um desempenho equilibrado. Além disso, o modelo faz pouca confusão entre o tráfego que deve ser permitido e bloqueado. Os erros de classificação de "C0 - Allow" como "C1 - Deny", "C2 - Drop" e "C3 - Reset-Both" são 4,94%, 0% e 0%, respectivamente. Adicionalmente, os erros de "C1 - Deny" como "C0 - Allow" foram de 5,76%, "C2 - Drop" como "C0 - Allow" foram de 14,99%, e "C3 - Reset-Both" como "C0 - Allow" foram de 20,06%.

Comparação de Desempenho Entre os Experimentos 1 e 2

Ao comparar os resultados do *Experimento 2* com o *Experimento 1*, utilizando a Tabela 65, observa-se que alguns modelos apresentaram um desempenho superior em termos de métricas como acurácia, precisão, recall e F1-Score. Esses modelos são destacados a seguir:

- **Matriz de Confusão 5 (Experimento 2 - F1-Score: 89,97%, Acerto C2 - Drop: 6,60%, Acerto C3 - Reset-Both: 0%)**: Este modelo apresentou um pequeno aumento no F1-Score (89,97%) em comparação com o Experimento 1 (89,67%). A acurácia e precisão também melhoraram no Experimento 2. Apesar do aumento no F1-Score, o acerto da classe "C2 - Drop" diminuiu de (8,01% no Experimento 1 para 6,60% no Experimento 2);

- **Matriz de Confusão 34 (Experimento 2 - F1-Score: 87,94%, Acerto C2 - Drop: 0%, Acerto C3 - Reset-Both: 58,41%)**: Similar à Matriz de Confusão 34, este modelo apresentou um F1-Score ligeiramente superior no Experimento 2 (87,94%) em comparação ao Experimento 1 (87,21%). No entanto, houve um aumento no acerto da classe "C3 - Reset-Both" de 32,25% (Experimento 1) para 58,41% (Experimento 2);

- **Matriz de Confusão 33 (Experimento 2 - F1-Score: 87,95%, Acerto C2 - Drop: 0%, Acerto C3 - Reset-Both: 36,36%)**: Este modelo também apresentou um F1-Score ligeiramente superior no Experimento 2 (87,95%) comparado ao Experimento 1 (87,12%). Similar à Matriz 34, houve um aumento no acerto da classe "C3 - Reset-Both" de 18,05% (Experimento 1) para 36,36% (Experimento 2);

- **Matriz de Confusão 23 (Experimento 2 - F1-Score: 88,17%, Acerto C2 - Drop: 12,24%, Acerto C3 - Reset-Both: 25,22%)**: Assim como os modelos anteriores, este modelo também apresentou um F1-Score ligeiramente superior no Experimento 2 (88,17%) em comparação ao Experimento 1 (86,68%). Houve um aumento no acerto da classe "C2 - Drop" de 2,61% (Experimento 1) para 12,24%

(Experimento 2). No entanto, a classe "C3 - Reset-Both" apresentou uma queda no acerto, de 43% (Experimento 1) para 25,22% (Experimento 2);

- **Matriz de Confusão 18 (Experimento 2 - F1-Score: 88,22%, Acerto C2 - Drop: 13,59%, Acerto C3 - Reset-Both: 79,94%)**: Este modelo demonstrou um F1-Score ligeiramente superior no Experimento 2, alcançando 88,22% em comparação aos 86,37% do Experimento 1. O acerto da classe "C2 - Drop" sofreu uma leve diminuição, passando de 14,77% (Experimento 1) para 13,59% (Experimento 2). Em contrapartida, a classe "C3 - Reset-Both" apresentou uma melhoria significativa no acerto, aumentando de 48,86% (Experimento 1) para 79,94% (Experimento 2);

- **Matriz de Confusão 6 (Experimento 2 - F1-Score: 89,94%, Acerto C2 - Drop: 12,31%, Acerto C3 - Reset-Both: 96,90%)**: Este modelo demonstrou um aumento considerável no F1-Score do Experimento 2, alcançando 89,94% em comparação aos 84,43% do Experimento 1. O acerto da classe "C2 - Drop" apresentou uma leve diminuição, passando de 13,73% (Experimento 1) para 12,31% (Experimento 2). No entanto, a classe "C3 - Reset-Both" apresentou uma melhoria significativa no acerto, aumentando de 10,35% (Experimento 1) para 96,90% (Experimento 2).

Conclusão

Os três modelos selecionados - Matriz de Confusão 6, 18 e 17, seguidas pela Matriz 23 - demonstraram um desempenho robusto na previsão das ações corretas para os exemplos do fluxo, um equilíbrio no acerto entre as classes e na minimização de erros de classificação críticos, sendo os mais equilibrados para cenários de FCDs entre todos os avaliados até o momento. Se considerássemos apenas o fluxo de dados atual e o melhor desempenho de classificação - com foco em métricas como Acurácia, Precisão e F1-Score - sem levar em conta o equilíbrio entre as classes, ou a capacidade de projeção para classificações futuras, onde o tráfego pode mudar de maneira abrupta, as melhores opções para a previsão da ação recomendada no tráfego de rede seriam as Matrizes de Confusão 5, 6 e 18. Esses modelos apresentaram F1-Scores de 89,97%, 89,94% e 88,22% respectivamente, destacando-se por sua taxa de acerto elevada.

7.4.4 Experimento 3

Considerando que o algoritmo CluStream Adaptado é incremental e se adapta conforme o fluxo de dados ocorre, o objetivo desse experimento foi avaliar o comportamento do algoritmo ao longo do tempo, observando como ele se ajusta ao *primeiro conjunto de*

dados e, posteriormente, como ele reage ao *segundo conjunto*. Diferentemente do *Experimento 2*, onde o modelo foi treinado apenas com 10% do primeiro conjunto de dados, sem se adaptar ao restante do fluxo, neste caso analisa-se a capacidade de adaptação contínua do algoritmo ao longo do fluxo completo.

No *Experimento 3*, foram empregadas as mesmas combinações de atributos e configurações de hiperparâmetros que resultaram nos 11 melhores desempenhos do *Experimento 1*, além do modelo da *Matriz 21*, que obteve a melhor taxa média de acerto nas classes, conforme apresentado na Tabela 50. Esta tabela organiza os modelos em ordem decrescente de acordo com o maior *F1-Score* obtido. No entanto, no *Experimento 3*, utilizou-se a combinação do *primeiro* e *segundo conjunto de dados*, conforme descrito na subseção 7.1.1 - Conjunto de Dados. O objetivo foi simular o fluxo inicial (Fase Online 1 - utilizado no Experimento 1) juntamente com o novo fluxo (Fase Online 2 - utilizado no Experimento 2). A fase offline empregada para gerar o modelo permaneceu a mesma do *primeiro conjunto de dados*, correspondente a 10% do total desse conjunto de dados. Os outros 90% do *primeiro conjunto de dados* e o *segundo conjunto de dados* foram unificados para formar uma nova fase online.

Na Tabela 79 são apresentadas as combinações de atributos e as configurações de hiperparâmetros utilizadas no *Experimento 3*. A numeração das matrizes de confusão foi mantida conforme os 12 melhores resultados dos *Experimento 1* e *Experimento 2*, para facilitar a comparação entre os resultados obtidos.

Tabela 79 – Configurações de hiperparâmetros para diferentes conjuntos de atributos no Experimento 3 - CluStream Adaptado.

Matriz de Confusão	**Conjunto de Atributos**	**n_micro _clusters**	**fat**	**delta**	**n_macro _clusters**
2	Combinação-10	200	2.0	1000.0	20
5	Combinação-10	500	1.5	1000.0	140
6	Combinação-10	150	2.0	1000.0	100
11	Todos os Atributos Importantes	500	1.5	1000.0	140
17	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	500	1.5	1000.0	140
18	Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT	150	2.0	1000.0	100
21	Todos os Atributos Importantes Sem IPs e com Combinação	150	1.0	9000.0	15
23	Todos os Atributos Importantes Sem IPs e com Combinação	500	1.5	1000.0	140
29	Todos os Atributos Importantes Sem IPs	500	1.5	1000.0	140
32	Combinação-12	200	2.0	1000.0	20
33	Combinação-12	150	1.0	9000.0	15
34	Combinação-12	250	1.5	1000.0	25

Fonte: Elaborado pelo autor.

Os resultados do *Experimento 3* foram obtidos com base nas combinações de atributos

e configurações de hiperparâmetros detalhadas na Tabela 79. Para cada conjunto de atributos e hiperparâmetros, a fase offline utilizou 10% do *primeiro conjunto de dados* para o treinamento do modelo. A fase online foi realizada com a unificação de 90% do *primeiro conjunto de dados* com o *segundo conjunto de dados*, simulando assim um novo fluxo. Esta nova fase online integrou os dados utilizados nos *Experimentos 1* e *2*.

Os resultados das métricas de desempenho para cada combinação estão apresentados na Tabela 80, que mostra, para cada métrica, o resultado obtido no *Experimento 1* seguido do resultado correspondente no *Experimento 3*. As linhas destacadas em verde indicam onde o modelo do *Experimento 3* apresentou desempenho superior em comparação aos modelos do *Experimento 1* nas métricas de acurácia, precisão, recall e F1-score. No entanto, os modelos correspondentes às *Matrizes 2 e 23* apresentaram um comportamento diferente: embora tenha obtido uma acurácia e recall ligeiramente melhores no *Experimento 1*, seu desempenho em termos de precisão e F1-score foi superior no *Experimento 2*.

Tabela 80 – Resultados das métricas de desempenho no Experimento 1/Experimento 3 em (%) - CluStream Adaptado.

Matriz de Confusão	**Média Acerto Classes**	**Acurácia**	**Precisão**	**Recall**	**F1-Score**	**Acerto C2 Drop**	**Acerto C3 RESET-BOTH**
5	51,35/50,28	92,1/90,84	90,96/89,57	92,1/90,84	89,67/88,40	8,01/7,20	0/0
32	56,86/60,20	90,33/89,63	85,03/84,36	90,33/89,63	87,27/86,68	0/0	32,35/48,08
34	56,81/58,98	90,27/87,99	84,99/82,38	90,27/87,99	87,21/85,01	0/0	32,25/47,40
33	53,22/55,67	90,18/89,87	84,92/84,98	90,18/89,87	87,12/86,96	0/0	18,05/28,82
23	59,46/66,99	89,41/88,92	86,34/87,51	89,42/88,92	86,68/87,93	2,61/16,36	43/62,89
18	62,56/65,37	87,75/85,04	85,92/83,82	87,75/85,04	86,37/83,96	14,77/14,84	48,86/65,58
17	59,46/64,58	89,49/90.18	83,56/84,50	89,49/90,18	86,33/87,13	0/0	45,33/64,77
29	58,65/59,30	87,15/85,47	86,11/83,78	87,15/85,47	85,79/83,81	14,3/4,63	33,77/47,95
2	47,59/47,49	88,17/88,16	82,54/82,72	88,17/88,16	85,05/85,15	0/0	0/0
6	52,20/53,46	85,95/86,41	86,38/86,55	85,95/86,41	84,43/84,85	13,73/12,93	10,35/15,67
11	58,37/64,07	87,19/85,69	81,54/83,82	87,19/85,69	84,13/84,01	0/9,71	45,64/63,71
21	62,75/54,17	77,44/61,51	83,57/59,20	77,44/61,51	79,89/55,21	51,19/0	42,8/95,22

Fonte: Elaborado pelo autor.

Para os 12 modelos do *Experimento 3*, serão apresentadas tanto a matriz de confusão percentual quanto a tradicional, conforme mostrado nas Tabelas 81 a 92. Essas matrizes oferecem uma visualização mais detalhada e um outro entendimento dos resultados obtidos. Assim como nos *Experimentos 1 e 2*, evidencia-se a significativa dificuldade dos modelos em prever corretamente os exemplos da classe C2 (Drop).

Além da apresentação das 12 matrizes de confusão, na Tabela 93 são apresentados os resultados dos Erros de Classificação Críticos para esses modelos. Essa métrica é especialmente relevante, pois valores menores indicam menor confusão dos modelos na distinção entre ações de permissão (C0 - Allow) e bloqueio (C1 - Deny, C2 - Drop e C3 - Reset-Both) de tráfego de rede, refletindo maior precisão na classificação.

Tabela 81 – Experimento 3 - Matriz de Confusão 5 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	97.83%	1.76%	0.41%	0.00%
C 1	3.88%	96.09%	0.03%	0.00%
C 2	6.81%	85.99%	7.20%	0.00%
C 3	100.00%	0.00%	0.00%	0.00%

	C 0	C 1	C 2	C 3
C 0	747974	13488	3137	0
C 1	21530	532841	142	0
C 2	6538	82608	6919	0
C 3	2342	0	0	0

Fonte: Elaborado pelo autor.

Tabela 82 – Experimento 3 - Matriz de Confusão 32 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95.57%	4.42%	0.00%	0.01%
C 1	2.86%	97.13%	0.00%	0.00%
C 2	1.01%	98.99%	0.00%	0.00%
C 3	51.92%	0.00%	0.00%	48.08%

	C 0	C 1	C 2	C 3
C 0	730740	33760	0	99
C 1	15881	538622	0	10
C 2	968	95097	0	0
C 3	1216	0	0	1126

Fonte: Elaborado pelo autor.

Tabela 83 – Experimento 3 - Matriz de Confusão 34 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	95.53%	4.45%	0.00%	0.02%
C 1	7.00%	92.99%	0.00%	0.00%
C 2	8.44%	91.56%	0.00%	0.00%
C 3	52.60%	0.00%	0.00%	47.40%

	C 0	C 1	C 2	C 3
C 0	730422	34041	0	136
C 1	38806	515693	0	14
C 2	8107	87958	0	0
C 3	1232	0	0	1110

Fonte: Elaborado pelo autor.

Tabela 84 – Experimento 3 - Matriz de Confusão 33 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	94.42%	5.57%	0.00%	0.01%
C 1	0.56%	99.43%	0.00%	0.00%
C 2	0.03%	99.97%	0.00%	0.00%
C 3	71.18%	0.00%	0.00%	28.82%

	C 0	C 1	C 2	C 3
C 0	721918	42579	0	102
C 1	3122	551377	0	14
C 2	29	96036	0	0
C 3	1667	0	0	675

Fonte: Elaborado pelo autor.

Tabela 85 – Experimento 3 - Matriz de Confusão 23 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	93.68%	3.45%	2.86%	0.37%
C 1	2.31%	95.04%	2.64%	0.00%
C 2	7.80%	75.85%	16.36%	0.00%
C 3	37.11%	0.00%	0.00%	62.89%

	C 0	C 1	C 2	C 3
C 0	716276	26404	21891	28
C 1	12826	527035	14652	0
C 2	7490	72861	15714	0
C 3	869	0	0	1473

Fonte: Elaborado pelo autor.

Tabela 86 – Experimento 3 - Matriz de Confusão 18 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	88.42%	9.51%	2.07%	0.00%
C 1	4.92%	92.64%	2.44%	0.00%
C 2	12.34%	72.82%	14.84%	0.00%
C 3	33.30%	1.11%	0.00%	65.58%

	C 0	C 1	C 2	C 3
C 0	676024	72696	15855	24
C 1	27294	513699	13520	0
C 2	11850	69955	14260	0
C 3	780	26	0	1536

Fonte: Elaborado pelo autor.

Tabela 87 – Experimento 3 - Matriz de Confusão 17 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	96.85%	3.15%	0.00%	0.29%
C 1	3.29%	96.71%	0.00%	0.00%
C 2	11.64%	88.36%	0.00%	0.00%
C 3	35.23%	0.00%	0.00%	64.77%

	C 0	C 1	C 2	C 3
C 0	740514	24063	0	22
C 1	18258	536255	0	0
C 2	11184	84881	0	0
C 3	825	0	0	1517

Fonte: Elaborado pelo autor.

Tabela 88 – Experimento 3 - Matriz de Confusão 29 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	86.76%	9.74%	3.50%	0.22%
C 1	2.13%	97.87%	0.01%	0.00%
C 2	6.12%	89.26%	4.63%	0.00%
C 3	50.30%	1.75%	0.00%	47.95%

	C 0	C 1	C 2	C 3
C 0	663379	74457	26746	17
C 1	11806	542675	32	0
C 2	5875	85744	4446	0
C 3	1178	41	0	1123

Fonte: Elaborado pelo autor.

Tabela 89 – Experimento 3 - Matriz de Confusão 2 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	93.48%	6.52%	0.00%	0.00%
C 1	3.53%	96.47%	0.00%	0.00%
C 2	15.69%	84.31%	0.00%	0.00%
C 3	100.00%	0.00%	0.00%	0.00%

	C 0	C 1	C 2	C 3
C 0	714771	49828	0	0
C 1	19580	534933	0	0
C 2	15076	80989	0	0
C 3	2342	0	0	0

Fonte: Elaborado pelo autor.

Tabela 90 – Experimento 3 - Matriz de Confusão 6 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	88.06%	11.13%	0.79%	0.03%
C 1	2.34%	97.16%	0.50%	0.00%
C 2	2.60%	84.47%	12.93%	0.00%
C 3	84.33%	0.00%	0.00%	15.67%

	C 0	C 1	C 2	C 3
C 0	673324	85070	6013	192
C 1	12991	538761	2759	2
C 2	2495	81148	12422	0
C 3	1975	0	0	367

Fonte: Elaborado pelo autor.

Tabela 91 – Experimento 3 - Matriz de Confusão 11 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	90.39%	8.49%	1.11%	0.38%
C 1	6.10%	92.46%	1.44%	0.00%
C 2	15.79%	74.50%	9.71%	0.00%
C 3	36.21%	0.09%	0.00%	63.71%

	C 0	C 1	C 2	C 3
C 0	691137	64939	8494	29
C 1	33845	512702	7966	0
C 2	15168	71572	9325	0
C 3	848	2	0	1492

Fonte: Elaborado pelo autor.

Tabela 92 – Experimento 3 - Matriz de Confusão 21 em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	93.35%	6.02%	0.00%	0.63%
C 1	71.87%	28.12%	0.00%	0.01%
C 2	72.00%	27.99%	0.00%	0.00%
C 3	3.71%	1.07%	0.00%	95.22%

	C 0	C 1	C 2	C 3
C 0	713715	46034	0	4850
C 1	398540	155933	0	40
C 2	69171	26894	0	0
C 3	87	25	0	2230

Fonte: Elaborado pelo autor.

Tabela 93 – Resultados da métrica de desempenho “Erros de Classificação Críticos” no Experimento 3 - CluStream Adaptado

Matriz de Confusão	**Erro Permitir X Bloquear (%)**
5	3,32
32	3,66
34	5,81
33	3,35
23	4,90
18	9,07
17	3,83
29	8,47
2	6,13
6	7,67
11	8,70
21	36,59

Fonte: Elaborado pelo autor.

7.4.4.1 Conclusões e Discussões dos Resultados do Experimento 3

Os resultados do *Experimento 3* foram apresentados na Tabela 80, com diferentes combinações de atributos e configurações de hiperparâmetros utilizados no algoritmo CluStream Adaptado. O objetivo principal deste experimento foi avaliar as mesmas configurações que geraram os melhores resultados nos *Experimentos 1* e *2*, mas utilizando uma fase online unificada, composta por 90% do *primeiro conjunto de dados* e o *segundo conjunto de dados*, ambos capturados através do *firewall* Palo Alto.

Este novo cenário teve como objetivo simular um ambiente de FCDs, proporcionando uma análise mais abrangente da eficácia das configurações testadas. A proposta foi observar o comportamento dos modelos diante de fluxos de dados distintos e verificar se as métricas do primeiro fluxo melhoram ou pioram em relação ao segundo. Como o algoritmo é incremental, o modelo se ajusta continuamente à medida que o fluxo de dados avança, permitindo analisar como ele se adapta ao longo do tempo e como isso influencia os resultados obtidos. Dessa forma, é possível avaliar o desempenho dos modelos em três cenários distintos: (1) aplicação do primeiro conjunto de dados, utilizando 10% para o treinamento inicial e 90% para simulação do fluxo (Experimento 1); (2) aplicação direta do segundo conjunto de dados após uma fase de treinamento com 10% do primeiro conjunto (Experimento 2); e (3) atualização incremental do modelo ao longo do primeiro fluxo, utilizando o modelo já adaptado ao primeiro fluxo para realizar previsões do segundo (Experimento 3). O estudo permite analisar como o modelo, após o treinamento inicial, responde à introdução contínua de novos exemplos do primeiro fluxo e como ele se adapta e evolui ao lidar com o segundo fluxo de dados.

Desempenho Geral e Critérios de Avaliação

Os resultados foram avaliados com base nas mesmas métricas de desempenho utilizadas nos *Experimentos 1 e 2*, incluindo acurácia, precisão, recall, e F1-Score, além da média de acerto das classes e do percentual de acerto das classes "C2 - Drop" e "C3 - Reset-Both". Devido ao desbalanceamento dos dados e à baixa frequência da classe "C3 - Reset-Both", o F1-Score se torna uma métrica mais representativa, pois combina a precisão e o recall, oferecendo uma medida mais equilibrada do desempenho preditivo.

Outro fator a ser analisado é a capacidade dos modelos de se adaptarem a mudanças abruptas no fluxo de tráfego, que são comuns em cenários de FCDs. Em tais condições, o modelo precisa ser suficientemente robusto para capturar variações inesperadas nos padrões de tráfego de rede, especialmente para a classe rara "C3 - Reset-Both". Por isso, foi dada especial atenção aos modelos que evitaram alcançar 0% de acerto na classe "C3 - Reset-Both" e que mantiveram um equilíbrio satisfatório entre todas as classes. Além disso, considerando que o algoritmo enfrentou dificuldades em classificar corretamente os exemplos da classe "C2 - Drop", também foi feita uma análise desse aspecto.

Erros de Classificação Críticos

Uma consideração importante na seleção dos modelos mais apropriados foi a análise dos erros de classificação críticos:

- **Falsos negativos para "Allow (C0)"**: Casos em que tráfego permitido (C0 - Allow) é incorretamente classificado como uma ação de bloqueio ("C1 - Deny", "C2 - Drop" ou "C3 - Reset-Both"). Esses erros podem resultar na interrupção de tráfego legítimo e são indesejáveis em um ambiente de rede;

- **Falsos positivos para "Allow (C0)"**: Casos em que tráfego que deveria ser bloqueado ("C1 - Deny", "C2 - Drop", "C3 - Reset-Both") é erroneamente classificado como permitido (C0 - Allow). Esses erros são especialmente perigosos, pois podem representar falhas de segurança que podem permitir o tráfego potencialmente perigoso.

Análise dos Melhores Modelos

Para a seleção dos melhores modelos, foi adotada uma abordagem equilibrada de análise das métricas de desempenho. Em vez de priorizar modelos que apresentaram resultados excelentes em uma métrica isolada, mas fracos em outras, buscou-se identificar aqueles com desempenho consistente em múltiplas métricas. Essa escolha visa garantir modelos mais robustos e confiáveis no contexto de FCDs.

Os experimentos realizados até o momento indicam que os modelos testados com o algoritmo CluStream Adaptado enfrentam desafios significativos na predição correta das classes C2 (Drop) e C3 (Reset-Both). No entanto, a maioria dos modelos demonstrou uma melhoria no desempenho da Classe C3 no *Experimento 3*, em comparação com o *Experimento 1*. Alguns deles conseguiram alcançar uma taxa de acerto superior a 50% para essa classe, um resultado que não foi observado no *Experimento 1*. A análise da *Matriz 21* revelou que, no *Experimento 1*, as taxas de acerto para a classe C2 foram de 51,19% e de 42,80% para a classe C3. No entanto, no *Experimento 3*, a *Matriz 21* apresentou um desempenho insatisfatório para a classe C2, com a taxa de acerto caindo para 0%. Em contrapartida, para a classe C3, houve um aumento significativo nas predições corretas, subindo de 42,8% para 95,22%. Esses resultados indicam que o modelo não conseguiu se adaptar à classe C2 no novo fluxo de dados implementado na fase online.

Assim como ocorreu nos *Experimentos 1 e 2*, os resultados do *Experimento 3* mostram que os modelos frequentemente confundem exemplos da classe C2 (Drop) com a classe C1 (Deny). Embora essa confusão represente um erro, ela é menos grave, pois são ações diferentes para bloquear um tráfego de rede. Por outro lado, a classe C3 (Reset-Both), que possui um número limitado de exemplos para treinamento e avaliação, é frequentemente confundida com a classe C0 (Allow). Essa confusão é mais preocupante, pois permite o tráfego que deveria ser bloqueado, aumentando o risco de falhas de segurança.

Com base nos critérios mencionados, os três modelos mais adequados, considerando a adaptabilidade a mudanças de fluxo, a minimização dos erros críticos de classificação e o desempenho geral em termos de F1-Score e acertos por classes, são apresentados a seguir:

1. **Matriz de Confusão 23 - Tabela 80** (F1-Score: 87,93%, Acerto C2 - Drop: 16,36%, Acerto C3 - Reset-Both: 62,89%): Este modelo destacou-se pelo seu alto valor de F1-Score (segundo maior entre todos os modelos), acurácia e taxa de acerto classe C3. Além disso, apresentou uma taxa média de acerto de 66,99% nas classes, a maior entre os modelos comparados. Os erros de classificação críticos foram minimizados: “C0 - Allow” foi erroneamente classificado como “C1 - Deny” em 3,45% dos casos, como “C2 - Drop” em apenas 2,86%, e como “C3 - Reset-Both” em 0,37%. Por outro lado, as classes de bloqueio (“C1 - Deny”, “C2 - Drop”, “C3 - Reset-Both”) foram classificadas como “C0 - Allow” com uma taxa de erro baixa (entre 2,31% e 37,11%). Esses resultados indicam um bom desempenho em relação aos outros modelos testados, além de confundir pouco as ações de permissão e bloqueio;

2. **Matriz de Confusão 18 - Tabela 80** (F1-Score: 83,96%, Acerto C2 - Drop: 14,84%, Acerto C3 - Reset-Both: 65,58%): Este modelo se destacou pelo seu bom

valor de F1-Score, acurácia e taxa de acerto na classe C3. Mas o que realmente o diferencia, porém, é a taxa média de acerto nas classes, que é de 65,37%. Com esse desempenho, ele ocupa a segunda posição entre os modelos testados. O erro de classificação de "C0 - Allow" para "C1 - Deny" foi de 9,51%, "C0 - Allow" para "C2 - Drop" foi de 2,07%, e "C0 - Allow" para "C3 - Reset-Both" foi de 0,00%. Enquanto isso, o erro de classificação de "C1 - Deny" como "C0 - Allow" foi de 4,92%, "C2 - Drop" como "C0 - Allow" foi de 12,34%, e "C3 - Reset-Both" como "C0 - Allow" foi de 33,40%. Comparado aos outros modelos, este apresenta um bom desempenho na predição geral e na média de acerto nas classes;

3. **Matriz de Confusão 17 - Tabela 80** (F1-Score: 87,13%, Acerto C2 - Drop: 0%, Acerto C3 - Reset-Both: 64,77%): Embora o modelo não tenha taxa de acerto na classe C2, ele se destacou com o terceiro melhor F1-Score e a terceira melhor taxa média de acerto de 64,58% nas classes, demonstrando um desempenho equilibrado. Além disso, o modelo faz pouca confusão entre o tráfego que deve ser permitido e bloqueado. Os erros de classificação de "C0 - Allow" como "C1 - Deny", "C2 - Drop" e "C3 - Reset-Both" são 3,15%, 0% e 0,29%, respectivamente. Adicionalmente, os erros de "C1 - Deny" como "C0 - Allow" foram de 3,29%, "C2 - Drop" como "C0 - Allow" foram de 11,64%, e "C3 - Reset-Both" como "C0 - Allow" foram de 35,23%.

Comparação de Desempenho Entre os Experimentos 1 e 3

Ao comparar os resultados do *Experimento 3* com o *Experimento 1*, utilizando a Tabela 80, observa-se que alguns modelos apresentaram um desempenho superior em termos de métricas como acurácia, precisão, recall e F1-Score. Esses modelos são destacados a seguir:

- **Matriz de Confusão 23 (Experimento 3 - F1-Score: 87,93,62%, Acerto C2 - Drop: 16,36%, Acerto C3 - Reset-Both: 62,89%)**: Este modelo apresentou um pequeno aumento no F1-Score (87,93%) em comparação com o Experimento 1 (86,68%). Embora a acurácia tenha registrado uma leve queda de 0,49%, a taxa média de acerto nas classes aumentou em 7,53% no Experimento 3. O acerto da classe C2 cresceu de 2,61% para 16,36%, enquanto o acerto da classe C3 aumentou de 43% para 62,89% no Experimento 3 em relação ao Experimento 1;
- **Matriz de Confusão 17 (Experimento 3 - F1-Score: 87,13%, Acerto C2 - Drop: 0%, Acerto C3 - Reset-Both: 64,77%)**: Similar à Matriz de Confusão 23, este modelo apresentou um F1-Score ligeiramente superior no Experimento 3 (87,13%) em comparação ao Experimento 1 (86,33%). Além disso, a taxa média de acerto nas classes aumentou em 5,12% no Experimento 3. Para a classe C3, a taxa de acerto subiu de 45,33% no Experimento 1 para 64,77% no Experimento 3;

- ❑ **Matriz de Confusão 2 (Experimento 3 - F1-Score: 85,15%, Acerto C2 - Drop: 0%, Acerto C3 - Reset-Both: 0%)**: Este modelo apresentou um F1-Score ligeiramente superior no Experimento 3, alcançando 85,15% em comparação com 85,05% no Experimento 1. No entanto, a taxa média de acerto nas classes teve uma leve queda de 0,1%. As classes C2 e C3 mantiveram uma taxa de acerto de 0%, sem alterações entre o Experimento 1 e o Experimento 3;

- ❑ **Matriz de Confusão 6 (Experimento 3 - F1-Score: 84,85%, Acerto C2 - Drop: 12,93%, Acerto C3 - Reset-Both: 15,67%)**: Este modelo apresentou um F1-Score ligeiramente superior no Experimento 3, atingindo 84,85%, em comparação com 84,43% no Experimento 1. Além disso, houve um pequeno aumento de 1,26% na taxa média de acerto nas classes. O acerto da classe C2, no entanto, diminuiu de 13,73% (Experimento 1) para 12,93% (Experimento 3). Por outro lado, a taxa de acerto da classe C3 aumentou de 10,35% (Experimento 1) para 15,67% (Experimento 3).

Conclusão

Os três modelos selecionados - Matriz de Confusão 23, 18 e 17, seguidas pela Matriz 11 - demonstraram um desempenho robusto na previsão das ações corretas para os exemplos do fluxo, um equilíbrio no acerto entre as classes e na minimização de erros de classificação críticos, sendo os mais equilibrados para cenários de FCDs entre todos os avaliados até o momento. Se considerássemos apenas o fluxo de dados atual e o melhor desempenho de classificação - com foco em métricas como Acurácia, Precisão e F1-Score - sem levar em conta o equilíbrio entre as classes, ou a capacidade de projeção para classificações futuras, onde o tráfego pode mudar de maneira abrupta, as melhores opções para a previsão da ação recomendada no tráfego de rede seriam as Matrizes de Confusão 5, 23 e 17. Esses modelos apresentaram F1-Scores de 88,4%, 87,93% e 87,13% respectivamente, destacando-se por sua taxa de acerto elevada.

Nos experimentos, inicialmente foram selecionados os melhores conjuntos de atributos com base nos resultados obtidos utilizando o algoritmo MINAS. Para cada um desses conjuntos, foram definidas seis combinações de hiperparâmetros, resultando em um total de 36 modelos no *Experimento 1.* Nos *Experimentos 2 e 3*, as mesmas combinações de atributos e configurações de hiperparâmetros foram reutilizadas, focando nos 11 melhores desempenhos do *Experimento 1*, definidos pelos maiores valores de *F1-Score.* Adicionalmente, o modelo referente à *Matriz 21*, que obteve a melhor taxa média de acerto nas classes, também foi incluído. Posteriormente, foram selecionados os modelos que apresentaram o melhor equilíbrio entre precisão e robustez, considerando não apenas a acurácia geral, mas também o acerto equilibrado entre as classes e a minimização de erros de

classificação críticos. Por exemplo, a classe C3, que possui poucos exemplos, pode experimentar um aumento exponencial abrupto no número de ocorrências. Dado que as características do fluxo de dados não são constantes e o perfil das classes existentes muda ao longo do tempo, é fundamental alcançar um equilíbrio entre o desempenho geral do modelo e sua capacidade de lidar de forma eficaz com cada classe individualmente. Além disso, é preciso que o modelo consiga distinguir o que deve ser bloqueado (classes C1, C2 e C3) e o que deve ser permitido (classe C0). Todos esses fatores foram considerados na seleção dos modelos.

É importante ressaltar que apenas um modelo do *Experimento 1*, que apresentou 0% de acerto nas classes C2 ou C3, conseguiu melhorar seu desempenho nos *Experimentos 2 e 3* (outros fluxos). Esse modelo é o da *Matriz 11*, que teve 0% de acerto na classe C2 no *Experimento 1*, 0,87% no *Experimento 2* e 9,71% no *Experimento 3*.

Ao analisar os experimentos realizados com diferentes combinações de atributos, é possível observar o seguinte:

- Todos os conjuntos de atributos utilizados nos experimentos apresentaram um desempenho de predição insatisfatório para a classe C2 (Drop). O único modelo que superou 50% de acerto nessa classe foi o da Matriz 21 (Conjunto: Todos os Atributos Importantes Sem IPs e com Combinação) no Experimento 1, mas seu desempenho caiu para 0% nos Experimentos 2 e 3. Portanto, excluindo o resultado da Matriz 21 no Experimento 1, não houve nenhum outro resultado que alcançasse mais de 50% de acerto para essa classe;

- O desempenho dos modelos na classe C3 (Reset-Both) no Experimento 1 também deixou a desejar, pois nenhum modelo alcançou uma taxa de acerto superior a 50% para essa classe. No Experimento 2, alguns modelos conseguiram superar essa marca, incluindo os conjuntos de atributos: Combinação-12, Todos os Atributos Importantes sem Atributos Relacionados a Endereços IP e NAT, Todos os Atributos Importantes Sem IPs, Combinação-10 e Todos os Atributos Importantes. Já no Experimento 3, os conjuntos que alcançaram mais de 50% de acerto para a classe C3 foram: Todos os Atributos Importantes Sem IPs e com Combinação, Todos os Atributos Importantes sem Atributos Relacionados a Endereços IP e NAT, Todos os Atributos Importantes e Todos os Atributos Importantes Sem IPs e com Combinação;

- O conjunto de atributos Combinação-12 não conseguiu acertar nenhum exemplo da classe C2 nos Experimentos 1, 2 e 3. No entanto, os modelos desse conjunto apresentaram um alto F1-Score em comparação a alguns modelos testados, alcançando 87% nos Experimentos 1 e 2, e 86% no Experimento 3.

Comparação de Desempenho Entre os Experimentos 2 e 3

Na Tabela 94 é apresentada a comparação de desempenho entre os Experimentos 2 e 3 para cada métrica. Os valores são exibidos na ordem: primeiro o resultado do *Experimento 2*, seguido pelo correspondente do *Experimento 3*. As linhas destacadas em verde indicam os casos em que os modelos do *Experimento 3* superaram os do *Experimento 2* nas métricas de acurácia, precisão, recall e F1-score.

Tabela 94 – Resultados das métricas de desempenho no Experimento 2/Experimento 3 em (%) - CluStream Adaptado.

Matriz de Confusão	Média Acerto Classes	Acurácia	Precisão	Recall	F1-Score	Acerto C2 Drop	Acerto C3 RESET-BOTH
5	50,93/50,28	92,36/90,84	91,46/89,57	92,36/90,84	89,97/88,40	6,60/7,20	0/0
32	61,98/60,20	89,31/89,63	84,10/84,36	89,31/89,63	86,51/86,68	0/0	57,08/48,08
34	63,20/58,98	90,69/87,99	85,91/82,38	90,69/87,99	87,94/85,01	0/0	58,41/47,40
33	57,77/55,67	90,72/89,87	86/84,98	90,72/89,87	87,95/86,96	0/0	36,36/28,82
23	56,95/66,99	89,84/88,92	88,12/87,51	89,84/88,92	88,17/87,93	12,24/16,36	25,22/62,89
18	70,61/65,37	89,73/85,04	87,70/83,82	89,73/85,04	88,22/83,96	13,59/14,84	79,94/65,58
17	67,03/64,58	88,70/90,18	83,88/84,50	88,70/90,18	85,88/87,13	0/0	78,83/64,77
29	61,80/59,30	88,01/85,47	85,32/83,78	88,01/85,47	85,32/83,81	0/4,63	58,85/47,95
2	28,46/47,49	59,96/88,16	64,30/82,72	59,96/88,16	49,45/85,15	0/0	0/0
6	75,73/53,46	91,63/86,41	90,03/86,55	91,63/86,41	89,94/84,85	12,31/12,93	96,90/15,67
11	62,59/64,07	80,62/85,69	78,06/83,82	80,62/85,69	78,44/84,01	0,87/9,71	76,70/63,71
21	30,97/54,17	62,90/61,51	60,62/59,20	62,90/61,51	56,88/55,21	0/0	0/95,22

Fonte: Elaborado pelo autor.

Observa-se que a maioria dos modelos obteve desempenho superior no Experimento 2 em comparação com o Experimento 3. No Experimento 2, os modelos foram treinados apenas com o *primeiro conjunto de dados* e testados diretamente no segundo conjunto. Por outro lado, no Experimento 3, os modelos foram inicialmente treinados com o *primeiro conjunto de dados* e, posteriormente, atualizados incrementalmente com os dados remanescentes desse mesmo conjunto, seguidos pelo *segundo conjunto de dados*, coletado 15 dias depois.

Embora os modelos do Experimento 2 tenham apresentado desempenho superior nas métricas clássicas de avaliação, os resultados obtidos foram bastante próximos aos valores do Experimento 3. Apenas a matriz de confusão 2 apresentou resultados significativamente melhores no Experimento 3 em comparação ao Experimento 2.

Em relação à taxa de acerto da classe *Drop*, ambos os experimentos apresentaram baixa taxa de acerto, sendo que o único modelo com uma diferença relevante entre os experimentos foi a matriz de confusão 11, que obteve um aumento de 8,84% na taxa de acerto no Experimento 3 em comparação com o Experimento 2.

Quanto à taxa de acerto da classe *Reset-Both*, houve uma variação significativa entre os experimentos. Por exemplo, a matriz de confusão 6 obteve 96,90% de acerto no

Experimento 2, mas teve apenas 15,67% no Experimento 3. Já a matriz de confusão 21 não acertou nenhum exemplo para essa classe no Experimento 2, mas alcançou 95,22% de acerto no Experimento 3.

Apesar de todos os experimentos realizados, é necessário conduzir estudos futuros para analisar mais profundamente o comportamento desses modelos à medida que o fluxo de dados evolui com novos exemplos. Isso permitirá uma maior compreensão de sua adaptabilidade e eficácia em cenários com maior representatividade e variabilidade dos dados ao longo do tempo.

7.5 MINAS, CluStream Adaptado e Trabalhos Relacionados

Até o momento, não foram encontrados na literatura especializada algoritmos que tratem de forma adequada a análise de *logs* de tráfego de rede em dispositivos de *firewall*, visando prever a ação recomendada (Allow, Drop, Deny e Reset-Both) e classificando as sessões de tráfego de rede de maneira contínua, utilizando algoritmos especificamente projetados para o cenário de FCDs.

A comparação direta entre os experimentos realizados neste trabalho, que utilizam algoritmos para FCDs, e os trabalhos relacionados que empregam algoritmos tradicionais de AM em lote (batch) não é apropriada devido às abordagens e conjuntos de dados diferentes provenientes de locais distintos—Sharma, Wason e Johri (2021) e Ertam e Kaya (2018) (Universidade Firat: Firewall Palo Alto 5020) e Aljabri et al. (2022) (Organização Privada). Além disso, a proposta deste trabalho emprega algoritmos baseados em microgrupos para classificar fluxos de dados em evolução em cenários nos quais os rótulos verdadeiros são considerados sujeitos a um atraso infinito, ou seja, nunca são disponibilizados. Dessa forma, o modelo não depende de rótulos verdadeiros para realizar atualizações durante a fase online.

Apesar de a métrica F1-Score ser apresentada nos experimentos, é importante destacar que ela, por si só, não define o melhor modelo. A escolha do modelo ideal foi baseada em uma análise mais abrangente, considerando várias métricas, conforme explicado e justificado ao longo deste trabalho. Portanto, os valores mais altos de F1-Score obtidos pelos algoritmos MINAS e CluStream Adaptado não representam necessariamente os melhores modelos para o cenário de FCDs.

Por fim, os trabalhos relacionados não avaliam o desempenho de cada classe de forma individual, e os conjuntos de dados utilizados foram pré-processados para garantir um número igual de instâncias por classe, o que facilita o aprendizado e a predição dos modelos. No entanto, essa abordagem não é viável em um cenário de FCDs, onde os dados chegam de forma contínua e desbalanceada, exigindo que os modelos lidem com a

variabilidade e a imprevisibilidade inerentes a esse contexto.

7.5.1 Comparação entre MINAS e CluStream Adaptado

Nesta seção, será realizada uma comparação entre os algoritmos MINAS e CluStream Adaptado com base nos resultados obtidos nos Experimentos 1, 2 e 3. Para cada experimento, foram selecionados os três modelos de cada algoritmo que foram considerados os mais adequados para FCDs. A escolha considerou o bom desempenho em diversas métricas avaliadas, priorizando modelos consistentes em múltiplos critérios, mesmo que não tenham apresentado os melhores resultados em métricas isoladas. A avaliação desses modelos foi conduzida de acordo com as métricas e critérios previamente definidos, permitindo uma análise do desempenho de ambos os algoritmos em diferentes cenários.

Experimento 1

Os resultados das métricas de desempenho dos modelos mais adequados para FCDs dos algoritmos MINAS e CluStream Adaptado no Experimento 1 são apresentados na Tabela 95.

Tabela 95 – Resultado de desempenho dos melhores modelos - MINAS e CluStream Adaptado - Experimento 1.

Algoritmo	Matriz de Confusão	Média Acerto Classes (%)	Precisão (%)	Recall (%)	F1-Score weighted (%)	F1-Score Macro (%)	Acerto C2 Drop (%)	Acerto C3 RESET BOTH (%)	Erro Permitir X Bloquear (%)
MINAS	Matriz 33	76.96	80.12	75.57	77.26	55.99	60.73	95.64	12.74
	Matriz 23	76.51	79.8	75.57	77.17	55.81	59.04	95.44	12.89
	Matriz 32	76.35	79.61	75.23	76.88	55.63	59.03	95.44	13.21
CluStream Adaptado	Matriz 18	62.56	85.92	87.75	86.37	66.46	14.77	48.86	6.17
	Matriz 23	59.46	86.34	89.42	86.68	62.23	2.61	43.00	4.33
	Matriz 21	62.75	83.57	77.44	79.89	53.86	51.19	42.80	10.14

Fonte: Elaborado pelo autor.

Ao comparar os resultados dos três modelos mais adequados para FCDs no experimento 1 (MINAS: Matriz 33, Matriz 23 e Matriz 32) e (CluStream Adaptado: Matriz 18, Matriz 23 e Matriz 21), observa-se uma clara vantagem do MINAS na taxa média de acerto nas classes, especialmente no desempenho das classes C2 (Drop) e C3 (Reset-Both), onde o CluStream Adaptado apresentou grande dificuldade em classificar corretamente esses exemplos. No entanto, apesar dessas dificuldades, o CluStream Adaptado se destacou em todas as demais métricas, incluindo uma menor confusão nas ações de permissão e bloqueio (erros críticos de classificação). Além disso, o CluStream Adaptado obteve melhores resultados no F1-score Macro, que calcula o F1-score individualmente para cada classe e depois faz a média simples desses valores, atribuindo o mesmo peso a cada classe, independentemente do número de amostras. Ele também superou no F1-score weighted, que pondera o F1-score de cada classe de acordo com o número de amostras, dando mais peso às classes com mais exemplos.

Experimento 2

Os resultados das métricas de desempenho dos modelos mais adequados para FCDs dos algoritmos MINAS e CluStream Adaptado no Experimento 2 são apresentados na Tabela 96.

Tabela 96 – Resultado de desempenho dos melhores modelos - MINAS e CluStream Adaptado - Experimento 2.

Algoritmo	Matriz de Confusão	Média Acerto Classes (%)	Precisão (%)	Recall (%)	F1-Score weighted (%)	F1-Score Macro (%)	Acerto C2 Drop (%)	Acerto C3 RESET BOTH (%)	Erro Permitir X Bloquear (%)
MINAS	Matriz 23	73.82	73.30	70.47	71.66	55.55	57.67	97.12	20.38
	Matriz 32	73.09	73.83	67.37	69.59	53.24	60.80	97.12	22.40
	Matriz 33	64.63	82.06	79.71	80.23	58.80	59.43	41.37	11.07
CluStream Adaptado	Matriz 6	75.73	90.03	91.63	89.94	72.16	12.31	96.90	2.54
	Matriz 18	70.61	87.70	89.73	88.22	73.11	13.59	79.94	4.81
	Matriz 17	67.03	83.88	88.70	85.88	67.54	0.00	78.83	6.55

Fonte: Elaborado pelo autor.

Ao comparar os três modelos mais adequados para FCDs no Experimento 2 (MINAS: Matriz 23, Matriz 32 e Matriz 33; CluStream Adaptado: Matriz 6, Matriz 18 e Matriz 17), observa-se uma vantagem clara do MINAS no desempenho da classe C2 (Drop). Embora a taxa média de acerto entre as classes seja relativamente equilibrada, o CluStream Adaptado se destacou em quase todas as demais métricas, especialmente em reduzir a confusão nas ações de permissão e bloqueio, que são erros críticos de classificação. Além disso, o CluStream Adaptado obteve melhores resultados nas métricas de Acurácia, Recall, F1-score weighted e F1-score macro.

Em relação à taxa de acerto na classe C3 (Reset-Both), o MINAS apresentou desempenho excelente nos dois primeiros modelos, mas caiu para abaixo de 50% no terceiro. Por outro lado, o CluStream Adaptado demonstrou desempenho excelente no primeiro modelo, e bom desempenho nos modelos subsequentes.

Experimento 3

Os resultados das métricas de desempenho dos modelos mais adequados para FCDs dos algoritmos MINAS e CluStream Adaptado no Experimento 3 são apresentados na Tabela 97.

Ao comparar os resultados dos três modelos mais adequados para FCDs no Experimento 3 (MINAS: Matriz 23, Matriz 33 e Matriz 32) e (CluStream Adaptado: Matriz 23, Matriz 18 e Matriz 17), observa-se uma vantagem do MINAS na taxa média de acerto nas classes, especialmente no desempenho das classes C2 (Drop) e C3 (Reset-Both). No entanto, o CluStream Adaptado se destacou nas demais métricas, especialmente em reduzir a confusão nas ações de permissão e bloqueio, que são erros críticos de classificação. Além disso, o CluStream Adaptado obteve melhores resultados nas métricas de Acurácia, Recall, F1-score weighted e F1-score macro.

Tabela 97 – Resultado de desempenho dos melhores modelos - MINAS e CluStream Adaptado - Experimento 3.

Algoritmo	Matriz de Confusão	Média Acerto Classes (%)	Precisão (%)	Recall (%)	F1-Score weighted (%)	F1-Score Macro (%)	Acerto C2 Drop (%)	Acerto C3 RESET BOTH (%)	Erro Permitir X Bloquear (%)
MINAS	Matriz 23	75.68	77.63	73.82	75.25	56.14	58.60	96.37	15.53
	Matriz 33	69.59	80.48	77.16	78.41	56.06	60.31	64.09	12.08
	Matriz 32	75.56	77.39	73.58	75.01	56.06	58.58	96.37	15.79
CluStream Adaptado	Matriz 23	66.99	87.51	88.92	87.93	70.62	16.36	62.89	4.90
	Matriz 18	65.37	83.82	85.04	83.96	68.83	14.84	65.58	9.07
	Matriz 17	64.58	84.50	90.18	87.13	66.01	0.00	64.77	3.83

Fonte: Elaborado pelo autor.

7.5.2 Conclusão

Os resultados dos experimentos realizados foram considerados distintos para o cenário de FCDs. Isso se deve ao fato de ambos os algoritmos, MINAS e CluStream Adaptado, terem apresentado desempenhos satisfatórios em diferentes métricas ao longo dos diferentes experimentos. O CluStream Adaptado, em particular, mostrou-se mais eficaz em lidar com os fluxos de dados ao obter resultados superiores em F1-Score weighted e macro, o que indica uma robustez em taxa de acerto e minimização dos erros críticos de classificação (confusão entre as ações de bloqueio e permissão).

Por outro lado, o MINAS, graças ao seu processo de DN, se destacou na taxa média de acerto entre as classes e na classificação de classes mais desbalanceadas, como C2 (Drop) e C3 (Reset-Both). Isso pode ser útil em situações onde ocorrem mudanças abruptas no padrão de tráfego de rede, fazendo com que os exemplos das classes C2 e C3 aumentem significativamente, enquanto outras classes experimentam uma redução. Assim, apesar das dificuldades em algumas métricas, ambos os algoritmos demonstraram uma capacidade razoável de adaptação ao FCDs.

Em geral, os algoritmos demonstraram uma adaptabilidade intermediária a cenários de fluxo de dados, embora insuficiente para simulações mais complexas e aplicações no mundo real. Futuros experimentos devem ser conduzidos utilizando conjuntos de dados coletados em diferentes dias e horários, integrando-os para simular um cenário de fluxo de dados mais abrangente. Isso permitirá a avaliação do desempenho dos modelos sob diversas condições de tráfego de rede, garantindo sua adequação a ambientes com padrões comportamentais variáveis e possíveis mudanças de conceito. Dessa forma, será possível explorar a capacidade dos modelos de se atualizarem incrementalmente ao longo do tempo, bem como de esquecer padrões obsoletos que não contribuem mais para o processo de classificação.

É importante ressaltar que os erros críticos têm um impacto direto e significativo na segurança da rede. No entanto, também é preciso analisar os erros associados à diferenciação entre os diferentes tipos de bloqueio (Deny, Drop e Reset-Both). Cada uma dessas ações possui um efeito distinto no tráfego de rede e no comportamento do sistema

de segurança. A aplicação incorreta de qualquer um desses tipos de bloqueio pode gerar efeitos indesejados, como a redução do desempenho da rede ou até a criação de novas vulnerabilidades. A análise desses erros não só contribui para a melhoria da precisão das previsões, mas também garante que cada ação de bloqueio seja aplicada de maneira otimizada, levando em consideração suas implicações no tráfego de rede e no desempenho do sistema.

SVM Incremental

Experimentos adicionais foram realizados utilizando um algoritmo supervisionado de latência nula (zero), no qual os rótulos são disponibilizados de forma imediata, utilizando um algoritmo SVM incremental para prever as ações recomendadas para o tráfego de rede. O algoritmo foi uma adaptação do SGDClassifier do Scikit-learn, no qual o modelo foi projetado para realizar aprendizado incremental, permitindo a atualização contínua de seus hiperparâmetros a cada exemplo, sem a necessidade de reiniciar o treinamento, e processando os dados individualmente na fase online (DEVELOPERS, 2025b). Para essa implementação, a maioria dos parâmetros foi mantida com os valores padrão do *Scikit-learn*. Apenas alguns parâmetros foram ajustados para configurar o modelo como um SVM linear, especificamente a função de perda `loss='hinge'`, de modo que o modelo se comporta como um classificador SVM linear treinado de forma incremental.

Na atualização contínua dos hiperparâmetros, o modelo ajusta seus pesos e hiperparâmetros de forma incremental, à medida que novos dados chegam, sem a necessidade de reiniciar o treinamento ou reprocessar o conjunto de dados completo. Essa abordagem é necessária para modelos que operam em ambientes dinâmicos e em FCDs.

Assim como o MINAS e o CluStream Adaptado, a performance do modelo foi avaliada em três experimentos distintos, à medida que novos dados eram processados, comparando as predições com os rótulos reais fornecidos durante a operação. No Experimento 1, o *primeiro conjunto de dados* foi utilizado, com 10% destinado à fase offline e os 90% restantes à fase online. No Experimento 2, a fase online foi realizada utilizando o *segundo conjunto de dados*. Já no Experimento 3, o SVM incremental utilizou 10% do *primeiro conjunto de dados* para treinar o modelo na fase offline, enquanto a fase online foi conduzida com a unificação dos 90% restantes desse conjunto e o *segundo conjunto de dados*.

Assim como o CluStream Adaptado, o SVM incremental enfrenta dificuldades notáveis na previsão da classe C2 (Drop), destacando os desafios comuns na modelagem e classificação dessa classe nos fluxos de dados de tráfego de rede. Esses desafios são evidenciados, por exemplo, nas Tabelas 98, 99 e 100, que apresentam os resultados dos experimentos realizados com o conjunto de atributos "Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT". Esse conjunto de atributos foi escolhido por ser

um dos que apresentaram os melhores resultados nos experimentos realizados com os algoritmos MINAS e CluStream Adaptado.

Tabela 98 – Matriz de Confusão do Algoritmo SVM incremental em forma de Porcentagem e Valores Absolutos - Experimento 1

	C 0	C 1	C 2	C 3
C 0	99.41%	0.59%	0.00%	0.00%
C 1	1.66%	98.34%	0.00%	0.00%
C 2	0.09%	99.91%	0.00%	0.00%
C 3	99.49%	0.10%	0.00%	0.41%

	C 0	C 1	C 2	C 3
C 0	480266	2865	0	0
C 1	5884	348416	0	0
C 2	57	63470	0	0
C 3	981	1	0	4

Fonte: Elaborado pelo autor.

Tabela 99 – Matriz de Confusão do Algoritmo SVM incremental em forma de Porcentagem e Valores Absolutos - Experimento 2

	C 0	C 1	C 2	C 3
C 0	95.56%	4.44%	0.00%	0.00%
C 1	3.89%	96.11%	0.00%	0.00%
C 2	2.23%	97.77%	0.00%	0.00%
C 3	58.04%	0.00%	0.00%	41.96%

	C 0	C 1	C 2	C 3
C 0	268968	12497	0	3
C 1	7784	192429	0	0
C 2	725	31813	0	0
C 3	787	0	0	569

Fonte: Elaborado pelo autor.

Tabela 100 – Matriz de Confusão do Algoritmo SVM incremental em forma de Porcentagem e Valores Absolutos - Experimento 3

	C 0	C 1	C 2	C 3
C 0	96.78%	3.22%	0.00%	0.00%
C 1	2.10%	97.90%	0.00%	0.00%
C 2	0.95%	99.05%	0.00%	0.00%
C 3	37.66%	0.00%	0.00%	62.34%

	C 0	C 1	C 2	C 3
C 0	739959	24630	0	10
C 1	11640	542873	0	0
C 2	908	95157	0	0
C 3	882	0	0	1460

Fonte: Elaborado pelo autor.

Como resultado, conforme apresentado na Tabela 101, os experimentos alcançaram os seguintes resultados em termos de métricas clássicas, média de acerto das classes, acerto das classes C2 e C3 (nas quais os modelos apresentaram dificuldades para previsões corretas) e erros críticos de classificação.

Random Forest e Adaptive Random Forest Classifier

Para fornecer algoritmos finais para comparação, foram realizados experimentos utilizando o Random Forest Classifier do Scikit-learn e o Adaptive Random Forest Classifier do CapyMOA.

Tabela 101 – Resultados Experimentos 1, 2 e 3 - SVM incremental

Algoritmo	Experimento	Média Acerto Classes (%)	Precisão (%)	Recall (%)	F1-Score weighted (%)	F1-Score Macro (%)	Acerto C2 Drop (%)	Acerto C3 RESET BOTH (%)	Erro Permitir X Bloquear (%)
SVM incremental	1	49.54	86.03	92.07	89.09	48.05	0.00	0.40	1.09
	2	58.41	85.06	90.11	87.15	61.02	0.00	41.96	4.23
	3	64.26	85.04	91.07	88.11	66.17	0.00	62.34	2.69

Fonte: Elaborado pelo autor.

O Random Forest Classifier é um algoritmo de aprendizado supervisionado que utiliza um conjunto de árvores de decisão (floresta) para realizar classificações. Ele usa o método de bagging (bootstrap aggregating) para treinar várias árvores em subconjuntos aleatórios dos dados e combina os resultados para melhorar a precisão e reduzir o *overfitting* (DEVELOPERS, 2025a).

Por padrão, o Random Forest Classifier opera em modo batch, ou seja, o modelo é treinado com todo o conjunto de dados disponível de uma vez, sem processamento incremental ou em fluxo contínuo. Isso o torna particularmente adequado para cenários onde os dados estão completos e podem ser carregados integralmente na memória.

No experimento realizado, foi utilizado conjunto de atributos "Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT", o mesmo empregado no SVM incremental. O *primeiro e o segundo conjuntos de dados* coletados para os experimentos foram unificados, permitindo o treinamento em modo batch. O dataset unificado foi então embaralhado e dividido em dois subconjuntos: 20% para treinamento e 80% para teste. Como resultado, o algoritmo alcançou uma acurácia geral de 98%, 79,98% de Acurácia para C2 (Drop), 93,81% para a C3 (Reset-Both), 0,09% de erros críticos, um F1-score weighted de 98% e um F1-score macro de 95%. Na Tabela 102 é apresentada a matriz de confusão, tanto em termos percentuais quanto em valores absolutos.

Tabela 102 – Matriz de Confusão Random Forest Classifier em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	99.96%	0.03%	0.00%	0.00%
C 1	0.14%	99.31%	0.54%	0.00%
C 2	0.04%	19.98%	79.98%	0.00%
C 3	6.19%	0.00%	0.00%	93.81%

	C 0	C 1	C 2	C 3
C 0	777126	272	0	16
C 1	804	560144	3073	0
C 2	36	19595	78440	0
C 3	145	0	0	2198

Fonte: Elaborado pelo autor.

Já o Adaptive Random Forest Classifier é baseado em florestas aleatórias, adapta-se a dados dinâmicos e é adequado para tarefas de aprendizado incremental, onde o modelo atualiza suas previsões à medida que novos dados chegam (GOMES et al., 2017). Como o algoritmo não possui fases distintas de treinamento offline e online, o *segundo conjunto de*

dados (Fase Online 2) foi utilizado para simular o FCDs. O mesmo conjunto de atributos "Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT", que foi empregado no SVM incremental e Random Forest Classifier , foi aplicado também nesse experimento. O modelo alcançou 98% de acurácia geral, 74% de Acurácia para C2 (Drop), 88% para a C3 (Reset-Both), 0,3% de erros críticos, um F1-score weighted de 98% e um F1-score macro de 94%. Na Tabela 103 é apresentada a matriz de confusão, tanto em termos percentuais quanto em valores absolutos.

Tabela 103 – Matriz de Confusão Adaptive Random Forest Classifier em forma de Porcentagem e Valores Absolutos

	C 0	C 1	C 2	C 3
C 0	99.88%	0.11%	0.01%	0.00%
C 1	0.47%	98.99%	0.54%	0.00%
C 2	0.37%	25.31%	74.31%	0.00%
C 3	11.43%	0.15%	0.00%	88.42%

	C 0	C 1	C 2	C 3
C 0	281138	304	24	2
C 1	935	198199	1079	0
C 2	122	8236	24180	0
C 3	155	2	0	1199

Fonte: Elaborado pelo autor.

É importante destacar que, mesmo ao considerar apenas o *segundo conjunto* de dados (Fase Online 2), que é menor em comparação ao *primeiro conjunto*, conforme explicado anteriormente, o tempo de execução do algoritmo durante a classificação foi significativamente superior ao tempo de execução do MINAS e do CluStream Adaptado. Os experimentos realizados com o *primeiro e primeiro e segundo conjuntos de dados* unificados não foram concluídos devido ao tempo de execução, que ultrapassou semanas sem término. Esse resultado indica que o Adaptive Random Forest Classifier exige um poder de processamento consideravelmente maior para processar o mesmo conjunto de dados utilizados pelos algoritmos MINAS, CluStream Adaptado, Random Forest Classifier e SVM Incremental.

Resultados dos Experimentos

Algoritmos não supervisionados, como MINAS e CluStream Adaptado, são capazes de detectar padrões anômalos ou agrupamentos naturais nos dados sem a necessidade de rótulos predefinidos. Essa característica os torna particularmente úteis para identificar comportamentos incomuns ou regras de configuração incorretas em sistemas, como *firewalls*. No entanto, esses algoritmos têm uma limitação intrínseca: eles não são projetados para prever ações específicas (por exemplo, *Allow, Deny, Drop, Reset-Both*) com a mesma precisão que algoritmos supervisionados, como o Adaptive Random Forest Classifier. Isso ocorre porque algoritmos supervisionados aprendem a mapear entradas para saídas específicas com base em rótulos predefinidos, enquanto algoritmos não supervi-

sionados focam exclusivamente na identificação de padrões intrínsecos nos dados, sem depender de rótulos.

Um exemplo prático dessa diferença pode ser observado em cenários onde há erros nas configurações das regras de um *firewall*. Se ações como A*llow, Deny, Drop* ou *Reset-Both* forem aplicadas de forma incorreta ao tráfego de rede, os logs gerados pelo *firewall* conterão rótulos equivocados. Nesse caso, algoritmos supervisionados, que dependem de rótulos para aprender e fazer previsões durante a fase online, podem ser negativamente impactados por esses erros, pois comparam suas previsões com os rótulos "reais"(que, neste cenário, estão incorretos). Por outro lado, algoritmos não supervisionados, que não dependem de rótulos predefinidos e se baseiam na identificação de grupos ou padrões naturais nos dados, tendem a ser menos afetados por esses erros de rotulagem, uma vez que não utilizam informações de rótulos durante a fase online.

Como esperado, os resultados dos experimentos mostraram que algoritmos supervisionados operando em modo batch apresentam desempenho superior em comparação com algoritmos supervisionados que trabalham com FCDs. Essa diferença ocorre porque o treinamento em batch permite que o modelo tenha acesso a todo o conjunto de dados de uma vez, o que facilita a identificação de padrões globais. Por outro lado, algoritmos supervisionados que operam com FCDs ainda superam algoritmos não supervisionados em FCDs, uma vez que conseguem aproveitar informações de rótulo para fazer previsões mais precisas. Na tabela 104 é apresentada uma comparação de desempenho entre esses algoritmos, considerando o conjunto de atributos denominado “Todos os Atributos Importantes sem atributos relacionados a endereços IP e NAT”, e utilizando o *primeiro* e *segundo* conjuntos de dados unificados. O algoritmo Adaptive Random Forest Classifier, que não possui fases distintas de treinamento offline e online, foi avaliado utilizando o *segundo conjunto de dados* (Fase Online 2), devido ao elevado tempo necessário para a execução do experimento, a fim de simular o FCDs. Já o Random Forest Classifier teve os dois conjuntos de dados unificados e embaralhados para trabalhar em modo batch.

Tabela 104 – Resultado de desempenho dos algoritmos supervisionados, não supervisionados e supervisionado em batch.

Algoritmo	Tipo	Média Acerto Classes (%)	Precisão (%)	Recall (%)	F1-Score weighted (%)	F1-Score Macro (%)	Acerto C2 Drop (%)	Acerto C3 RESET BOTH (%)	Erro Permitir X Bloquear (%)
MINAS	Não Supervisionado	69.59	80.48	77.16	78.41	56.06	60.31	64.09	12.08
CluStream Adaptado	Não Supervisionado	64.58	84.50	90.18	87.13	66.01	0.00	64.77	3.83
SVM incremental	Supervisionado	64.26	85.04	91.07	88.11	66.17	0.00	62.34	2.69
Adaptive Random Forest	Supervisionado	90.40	97.89	97.89	97.79	93.64	74.31	88.42	0.30
Random Forest	Supervisionado (Batch)	93.27	98.33	98.34	98.28	95.42	79.98	93.81	0.08

Fonte: Elaborado pelo autor.

Por fim, embora o MINAS e o CluStream Adaptado não tenham se mostrado totalmente eficientes e adequados para FCDs em todas as métricas avaliadas, é necessário conduzir experimentos futuros com esses modelos utilizando diferentes conjuntos de dados, coletados em dias e horários distintos, para avaliar de maneira mais precisa a consistên-

cia e a robustez desses modelos na tarefa de prever a ação recomendada. Isso permitirá um entendimento mais aprofundado dos comportamentos dos algoritmos em diferentes cenários, além de contribuir para a validação de seu desempenho na previsão das ações recomendadas para o tráfego de rede ao longo do tempo. Esse processo aproveitará tanto a incrementalidade quanto os mecanismos de esquecimento, adaptando e avaliando os modelos progressivamente com o passar do FCDs.

Além dos resultados experimentais obtidos, este trabalho demonstra que a previsão da ação recomendada em FCDs pode ser utilizada como uma ferramenta complementar de monitoramento, com destaque para a detecção de padrões incomuns, mudanças abruptas e o acompanhamento adaptativo do tráfego de rede ao longo do fluxo. Embora os modelos ainda não apresentem desempenho suficiente para validar diretamente as decisões do *firewall* ou suas regras, eles já oferecem utilidade prática em contextos de apoio à segurança. Com modelos mais robustos e alta taxa de acerto, essa abordagem poderia ser expandida para aplicações mais autônomas, como, por exemplo, o desenvolvimento de um sistema independente capaz de capturar dados diretamente da rede e, com base nos padrões aprendidos, decidir permitir ou negar o tráfego, mesmo na ausência de regras pré-configuradas e de *firewalls* tradicionais.

Balanceamento dos Dados

Para avaliar se o desempenho dos algoritmos poderia ser aprimorado com dados mais balanceados, foram aplicadas técnicas de oversampling no conjunto de dados offline, incluindo o SMOTE (*Synthetic Minority Over-sampling Technique*) e o ADASYN (*Adaptive Synthetic Sampling*), que geram novas amostras sintéticas para aumentar a representatividade das classes minoritárias. No entanto, essas abordagens não resultaram em melhorias no desempenho dos modelos. Em trabalhos futuros, pode ser interessante investigar mais profundamente técnicas de balanceamento em FCDs, explorando, por meio de experimentos mais detalhados, se essas estratégias e outras abordagens descritas na literatura podem efetivamente contribuir para aprimorar o desempenho dos modelos.

7.6 Considerações Finais

Neste capítulo, foram apresentados os detalhes dos experimentos realizados e os resultados obtidos na previsão da ação recomendada para o tráfego de rede em *firewalls*. Foram discutidas a base de dados utilizada, os métodos experimentais aplicados e os resultados obtidos com os algoritmos MINAS e CluStream Adaptado. Também foram feitas comparações entre os experimentos deste trabalho e os de estudos relacionados. Por fim, foi realizada uma comparação entre os modelos mais adequados para FCDs, com base

em diversas métricas de desempenho selecionadas dos algoritmos MINAS e CluStream Adaptado, além da apresentação de experimentos utilizando algoritmos supervisionados.

Capítulo 8

Conclusões

Neste capítulo são apresentadas as conclusões deste trabalho. Na Seção 8.1 são apresentadas as principais contribuições; na Seção 8.2 são discutidas algumas limitações das propostas realizadas; e por fim, na Seção 8.3 são apresentados direcionamentos para trabalhos futuros.

8.1 Principais Contribuições

Embora existam trabalhos que utilizam registros de *logs* de *firewalls* para prever a ação recomendada para o tráfego de rede, até o momento não foi encontrado nenhum estudo que aplique técnicas de MD em FCDs para esse propósito. Diante disso, este trabalho apresenta as seguintes contribuições:

- **Aplicação de Algoritmos de MD em FCDs:** Ao utilizar técnicas de MD em FCDs, o trabalho possibilita a classificação de tráfego de rede lidando com grandes volumes de dados em alta velocidade. Além disso, esses modelos têm a capacidade de atualizar seu conhecimento incorporando novos dados periodicamente, sem a necessidade de treinar o modelo do zero com todos os dados disponíveis anteriormente, algo não tratado adequadamente pelos métodos tradicionais de AM;

- **Adaptação do CluStream para Previsão de Ações Recomendadas em Fluxos de Tráfego de Rede:** A implementação de uma versão modificada do CluStream, adaptada para a tarefa específica de prever ações recomendadas (*Allow, Deny, Drop e Reset-Both*) em tráfego de rede, utilizando *logs* de *firewall*;

- **Aplicação do Algoritmo MINAS para a Previsão de Ações Recomendadas em Fluxos de Tráfego de Rede:** Utilização do MINAS, com sua capacidade de detectar extensões de conceito para a previsão de ações recomendadas (*Allow, Deny, Drop e Reset-Both*) em tráfego de rede, utilizando *logs* de *firewall*. Assim, é utilizada uma abordagem com DN para classificar o tráfego de rede, ficando evidente que, graças a essa abordagem, o algoritmo apresentou resultados superiores nas classes desbalanceadas, quando comparado ao CluStream Adaptado.

- **Análise e Identificação dos Melhores Conjuntos de Atributos para Cenários Não Supervisionados com Latência Infinita:** Entre os conjuntos de atributos avaliados, os melhores resultados para a classificação da ação recomendada no tráfego de rede, utilizando *logs* de *firewall* com os algoritmos MINAS e CluStream Adaptado, foram obtidos com os conjuntos "Todos os Atributos Importantes sem IP e NAT", "Todos os Atributos Importantes sem IP" e "Todos os Atributos Importantes sem IP com Combinação". Isso demonstra que a exclusão dos atributos relacionados a endereços IP teve um impacto positivo nos resultados. Esses atributos são altamente variáveis, específicos de cada sessão e pouco generalizáveis, o que pode comprometer o desempenho dos modelos. Detalhes sobre os atributos que compõem os melhores conjuntos de atributos podem ser consultados na Seção 7.2 — *Combinações de Atributos*, no capítulo *Experimentos*;

- **Comparação Entre Algoritmos MINAS, CluStream Adaptado e algoritmos Supervisionados:** A avaliação comparativa entre os dois algoritmos fornece uma visão detalhada de como técnicas de MD em FCDs se comportam ao lidar com padrões de tráfego de rede. Essa análise examina a robustez e a eficácia de ambos os algoritmos na previsão das ações recomendadas, destacando suas respectivas vantagens e limitações. Além disso, foi realizada uma comparação desses algoritmos com o Adaptive Random Forest Classifier e o SVM Incremental, que também operam em FCDs, bem como com o Random Forest Classifier, que funciona no modo batch.

8.2 Limitações

As principais limitações deste trabalho são:

- **Dependência de hiperparâmetros:** Tanto o CluStream Adaptado quanto o MINAS requerem a definição de diversos hiperparâmetro (número de microgrupos, *thresholds*, etc.), o que pode influenciar diretamente a performance dos algoritmos. A escolha inadequada desses hiperparâmetro pode prejudicar a eficácia das classificações;

- **Generalização em Ambientes Heterogêneos:** A adaptação dos algoritmos foi testada em logs de firewall capturados por um dispositivo específico (Palo Alto 3260). Outros tipos de *firewalls*, com características diferentes, podem requerer ajustes adicionais nos algoritmos para obter resultados comparáveis;

- **Metodologia de Avaliação:** A escolha do modelo mais adequado é complexa pelo desempenho variável em diferentes métricas de avaliação. A decisão dependerá das prioridades do sistema, como a necessidade de foco em robustez, minimização de erros críticos ou maior precisão na classificação de classes desbalanceadas;

- **Necessidade de Realizar Mais Experimentos com Diferentes Conjuntos de Dados:** Para avaliar de maneira mais precisa a consistência e a robustez dos modelos na tarefa de prever a ação recomendada, é necessário realizar mais experimentos utilizando conjuntos de dados coletados em diferentes dias e horários, integrando-os para simular um fluxo contínuo. Isso permitirá verificar o desempenho dos modelos sob condições variadas de tráfego de rede, assegurando que eles sejam adequados para cenários com diferentes padrões de comportamento e possíveis mudanças de conceito. Dessa forma, será possível explorar a capacidade dos modelos de se atualizarem de forma incremental ao longo do tempo, além de esquecer padrões obsoletos que já não contribuem para o processo de classificação;

- **Algoritmos Baseados em Grupos Hiperesféricos:** Os algoritmos não supervisionados utilizados nos experimentos identificam exclusivamente grupos hiperesféricos. No entanto, existem outras abordagens, como os algoritmos baseados em densidade, que podem se adaptar melhor aos dados no contexto de tráfego de rede, oferecendo potencial para capturar padrões mais complexos e estruturas irregulares.

8.3 Trabalhos Futuros

Os trabalhos futuros podem explorar as seguintes áreas:

- **Explorar Algoritmos Alternativos para FCDs:** Investigar o uso de outros algoritmos de MD voltados para FCDs, como o DenStream e o D-Stream, com o objetivo de comparar seu desempenho em relação ao CluStream Adaptado e ao MINAS na previsão da ação recomendada para o tráfego de rede. Adicionalmente, sugere-se realizar experimentos com uma versão semi-supervisionada do MINAS, a fim de avaliar se a incorporação parcial de rótulos durante a fase online pode contribuir para a melhoria do desempenho na classificação das ações recomendadas.

- **Análise Temporal de Desempenho:** Realizar uma análise temporal do desempenho dos algoritmos, avaliando como a precisão e robustez mudam ao longo do

tempo à medida que o tráfego de rede e as políticas de segurança se modificam. Isso pode ajudar a identificar como os algoritmos se adaptam a mudanças no comportamento da rede durante o FCDs;

- **Técnicas de Balanceamento em FCDs:** Investigar diferentes técnicas de balanceamento de dados aplicadas a FCDs, com o objetivo de verificar se estratégias mais avançadas ou específicas para fluxos contínuos podem contribuir para melhorar o desempenho dos modelos. Essa análise pode ajudar a lidar com a desproporção de classes em cenários dinâmicos e em constante evolução, melhorando a taxa de acerto nas classes em que os modelos atuais apresentaram dificuldades;

- **Aplicação em Diferentes Dispositivos de Segurança:** Explorar a aplicação e adaptação dos algoritmos utilizados em diferentes tipos de *firewalls* e dispositivos de segurança de rede, além do Palo Alto 3260.

Referências

ABDALLAH, Z. S. et al. Anynovel: Detection of novel concepts in evolving data streams: An application for activity recognition. **Evolving Systems**, Springer, v. 7, p. 73–93, 2016.

ACKERMANN, M. R. et al. Streamkm++: A clustering algorithm for data streams. **ACM Journal of Experimental Algorithmics**, v. 17, n. 1, 2012.

AGGARWAL, C. C. A framework for diagnosing changes in evolving data streams. In: **Proceedings of the 2003 ACM SIGMOD international conference on Management of data**. [S.l.: s.n.], 2003. p. 575–586.

_____. **Data Streams: Models and Algorithms**. [S.l.]: Springer Science & Business Media, 2007. v. 31.

_____. A survey of stream classification algorithms. **Data classification: algorithms and applications**, v. 245, 2014.

AGGARWAL, C. C. et al. A framework for clustering evolving data streams. In: VLDB ENDOWMENT. **Proceedings of the 29th International Conference on Very Large Data Bases-Volume 29**. [S.l.], 2003. p. 81–92.

AL-AMRI, R. et al. A review of machine learning and deep learning techniques for anomaly detection in iot data. **Applied Sciences**, MDPI, v. 11, n. 12, p. 5320, 2021.

ALBERTINI, M. K.; MELLO, R. F. de. A self-organizing neural network to approach novelty detection. In: **Machine Learning: Concepts, Methodologies, Tools and Applications**. [S.l.]: IGI Global, 2012. p. 262–282.

ALJABRI, M. et al. Classification of firewall log data using multiclass machine learning models. **Electronics**, MDPI, v. 11, n. 12, p. 1851, 2022.

ASAI, T. et al. Online algorithms for mining semi-structured data stream. In: IEEE. **2002 IEEE International Conference on Data Mining, 2002. Proceedings.** [S.l.], 2002. p. 27–34.

BABCOCK, B. et al. Models and issues in data stream systems. In: ACM. **Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems**. [S.l.], 2002. p. 1–16.

BERKHIN, P. A survey of clustering data mining techniques. In: **Grouping multidimensional data: Recent advances in clustering**. [S.l.]: Springer, 2006. p. 25–71.

BIFET, A. et al. New ensemble methods for evolving data streams. In: **Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.: s.n.], 2009. p. 139–148.

CAMPELLO, R. J. et al. Hierarchical density estimates for data clustering, visualization, and outlier detection. **ACM Transactions on Knowledge Discovery from Data (TKDD)**, ACM New York, NY, USA, v. 10, n. 1, p. 1–51, 2015.

CAO, F. et al. Density-based clustering over an evolving data stream with noise. In: SIAM. **Proceedings of the 2006 SIAM International Conference on Data Mining**. [S.l.], 2006. p. 328–339.

CASSALES, G. W. et al. Idsa-iot: an intrusion detection system architecture for iot networks. In: IEEE. **2019 IEEE Symposium on Computers and Communications (ISCC)**. [S.l.], 2019.

CHEN, Y.; TU, L. Density-based clustering for real-time stream data. In: **Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. [S.l.: s.n.], 2007. p. 133–142.

Cisco. **What is a Firewall?** 2022. Cisco. Disponível em: <https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-firewall.html>.

DEVELOPERS, S. learn. **Random Forest Classifier - Scikit-learn**. 2025. Scikit-learn. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>.

_____. **Stochastic Gradient Descent (SGD) - Scikit-learn**. 2025. Scikit-learn. Disponível em: <https://scikit-learn.org/stable/modules/sgd.html>.

DONG, G. et al. Online mining of changes from data streams: Research problems and preliminary results. In: **Proceedings of the 2003 ACM SIGMOD Workshop on Management and Processing of Data Streams**. [S.l.: s.n.], 2003. p. 739–747.

ERTAM, F.; KAYA, M. Classification of firewall log files with multiclass support vector machine. In: IEEE. **2018 6th International symposium on digital forensic and security (ISDFS)**. [S.l.], 2018. p. 1–4.

FACHINELLI, M.; AHLERT, E. M. Firewall de próxima geração-fortinet. **Revista Destaques Acadêmicos**, v. 11, n. 4, 2019.

FARIA, E. R.; GAMA, J.; CARVALHO, A. C. Novelty detection algorithm for data streams multi-class problems. In: **Proceedings of the 28th annual ACM symposium on applied computing**. [S.l.: s.n.], 2013. p. 795–800.

FARIA, E. R. et al. Novelty detection in data streams. **Artificial Intelligence Review**, v. 45, n. 2, p. 235–269, 2016.

FARIA, E. R. de; CARVALHO, A. C. Ponce de L. F.; GAMA, J. Minas: multiclass learning algorithm for novelty detection in data streams. **Data mining and knowledge discovery**, Springer, v. 30, p. 640–680, 2016.

FARIA, E. R. de et al. Evaluation of multiclass novelty detection algorithms for data streams. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 27, n. 11, p. 2961–2973, 2015.

FARID, D. M.; RAHMAN, C. M. Novel class detection in concept-drifting data stream mining employing decision tree. In: IEEE. **2012 7th International Conference on Electrical and Computer Engineering**. [S.l.], 2012. p. 630–633.

FERNANDES, E. L.; ROTHENBERG, C. E. Openflow 1.3 software switch. **Salao de Ferramentas do XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos SBRC**, UFSC Florianópolis, Brazil, p. 1021–1028, 2014.

FIORENZA, M.; KREUTZ, D. Firewalls em redes definidas por software: Estado da arte. In: SBC. **Anais da XVII Escola Regional de Redes de Computadores**. [S.l.], 2019. p. 81–88.

FISHER, D. Knowledge acquisition via incremental conceptual clustering. **Machine Learning**, v. 2, 1987.

GAMA, J. **Knowledge Discovery from Data Streams**. [S.l.]: Chapman & Hall/CRC, 2010.

GAMA, J.; RODRIGUES, P. P. An overview on mining data streams. In: **Foundations of Computational Intelligence - Volume 6: Data Mining**. Berlin: Springer, 2009, (Studies in Computational Intelligence, v. 206).

GEPPERTH, A.; HAMMER, B. Incremental learning algorithms and applications. In: **European symposium on artificial neural networks (ESANN)**. [S.l.: s.n.], 2016.

GOMES, H. M. et al. Adaptive random forests for evolving data stream classification. **Machine Learning**, Springer, v. 106, p. 1469–1495, 2017.

GONÇALVES, D. G. et al. Ips architecture for iot networks overlapped in sdn. In: IEEE. **2019 Workshop on Communication Networks and Power Systems (WCNPS)**. [S.l.], 2019. p. 1–6.

GUHA, S. et al. Clustering data streams. In: **Annual IEEE Symposium on Foundations of Computer Science**. Piscataway, NJ, USA: IEEE Educational Activities Department, 2000.

HAT, R. What is an intrusion detection and prevention system (idps). 2023. Accessed: 2024-07-02. Disponível em: <https://www.redhat.com/en/topics/security/what-is-an-intrusion-detection-and-prevention-system-idps>.

HOI, S. C. et al. Online learning: A comprehensive survey. **Neurocomputing**, Elsevier, v. 459, p. 249–289, 2021.

HOYER, E.; MEIRELLES, P.; PROTECTOR, R. **Firewall**. 2013. Disponível em: <https://www.gta.ufrj.br/grad/13_1/firewall/index.html>.

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: a review. **ACM computing surveys (CSUR)**, Acm New York, NY, USA, v. 31, n. 3, p. 264–323, 1999.

JUNIOR, D. M. M. Segurança da informação: uma abordagem sobre proteção da privacidade em internet das coisas. 2018.

KAUFMAN, L.; ROUSSEEUW, P. J. **Finding Groups in Data: An Introduction to Cluster Analysis**. [S.l.]: Wiley-Interscience, 2005.

KLINKENBERG, R. Learning drifting concepts: Example selection vs. example weighting. **Intelligent data analysis**, IOS Press, v. 8, n. 3, p. 281–300, 2004.

KOMADINA, A. et al. Comparative analysis of anomaly detection approaches in firewall logs: Integrating light-weight synthesis of security logs and artificially generated attack detection. **Sensors**, MDPI, v. 24, n. 8, p. 2636, 2024.

KRANEN, P. et al. The clustree: Indexing micro-clusters for anytime stream mining. **Knowledge and Information Systems**, v. 29, n. 2, p. 249–272, 2011.

KUROSE, J. F.; ROSS, K. W. **Redes de Computadores e a Internet: Uma abordagem top-down. Trad**. [S.l.]: Sao Paulo: Addison Wesley, 2006.

LANDGREBE, T. C. et al. The interaction between classification and reject performance for distance-based reject-option classifiers. **Pattern Recognition Letters**, Elsevier, v. 27, n. 8, p. 908–917, 2006.

LANGLEY, P. **Learning in Humans and Machines: Towards an Interdisciplinary Learning Science - Order Effects in Incremental Learning**. [S.l.]: Oxford, 1995.

LING, C.; LING-JUN, Z.; LI, T. Stream data classification using improved fisher discriminate analysis. **J. Computers**, v. 4, n. 3, p. 208–214, 2009.

LLOYD, S. P. Least squares quantization in PCM. **IEEE Transactions on Information Theory**, v. 28, n. 2, p. 129–137, 1982.

LOSING, V.; HAMMER, B.; WERSING, H. Incremental on-line learning: A review and comparison of state of the art algorithms. **Neurocomputing**, Elsevier, v. 275, p. 1261–1274, 2018.

LUO, Y. et al. An appraisal of incremental learning methods. **Entropy**, MDPI, v. 22, n. 11, p. 1190, 2020.

MACQUEEN, J. B. Some methods of classification and analysis of multivariate observations. In: **Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability**. [S.l.: s.n.], 1967. p. 281–297.

MARKOU, M.; SINGH, S. Novelty detection: A review – part 1: Statistical approaches. **Signal Processing**, v. 83, n. 12, p. 2481–2497, 2003.

MARSLAND, S.; SHAPIRO, J.; NEHMZOW, U. A self-organising network that grows when required. **Neural Networks**, v. 15, p. 1041–1058, 2002.

MASUD, M. M. et al. Detecting recurring and novel classes in concept-drifting data streams. In: IEEE. **2011 IEEE 11th International Conference on Data Mining**. [S.l.], 2011. p. 1176–1181.

_____. Classification and novel class detection in concept-drifting data streams under time constraints. **IEEE Transactions on knowledge and data engineering**, IEEE, v. 23, n. 6, p. 859–874, 2010.

MENDELSON, S.; LERNER, B. Online cluster drift detection for novelty detection in data streams. In: IEEE. **2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)**. [S.l.], 2020. p. 171–178.

MITCHELL, T. M. **Machine Learning**. 1. ed. New York, NY, USA: McGraw-Hill, Inc., 1997.

NEUPANE, K.; HADDAD, R.; CHEN, L. Next generation firewall for network security: A survey. In: **Proceedings of the SoutheastCon 2018**. St. Petersburg, FL, USA: IEEE, 2018. v. 2018, p. 19–22.

NGUYEN, H.-L.; WOON, Y.-K.; NG, W.-K. A survey on data stream clustering and classification. **Knowledge and Information Systems**, v. 45, n. 3, p. 535–569, 2015.

PAIVA, E. R. d. F. **Detecção de novidade em fluxos contínuos de dados multiclasse**. Tese (Doutorado) — Universidade de São Paulo, 2014.

Palo Alto Networks. **Firewall**. 2023. Palo Alto Networks. Disponível em: <https://www.paloaltonetworks.com/cyberpedia/what-is-a-firewall>.

_____. **Firewall**. 2024. Palo Alto Networks. Disponível em: <https://docs.paloaltonetworks.com/content/dam/techdocs/en_US/pdf/pan-os/10-2/pan-os-admin/pan-os-admin.pdf>.

_____. **Firewall**. 2024. Palo Alto Networks. Disponível em: <https://www.paloaltonetworks.com.br/resources/datasheets/pa-3200-series>.

_____. **Firewall**. 2024. Palo Alto Networks. Disponível em: <https://docs.paloaltonetworks.com/pan-os/9-1/pan-os-admin/policy/security-policy/security-policy-actions>.

PIMENTEL, M. A. et al. A review of novelty detection. **Signal processing**, Elsevier, v. 99, p. 215–249, 2014.

PINTO, C. M. S. **Algoritmos Incrementais para Aprendizagem Bayesiana**. Tese (Doutorado) — Faculdade de Economia da Universidade do Porto, Porto, Portugal, 2005.

ROSHAN, S. et al. Adaptive and online network intrusion detection system using clustering and extreme learning machines. **Journal of the Franklin Institute**, Elsevier, v. 355, n. 4, p. 1752–1779, 2018.

SCARFONE, K.; MELL, P. **Guide to Intrusion Detection and Prevention Systems (IDPS)**. National Institute of Standards and Technology (NIST), 2007. Special Publication 800-94. Disponível em: <https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=50951>.

SCHINDLER, T. Anomaly detection in log data using graph databases and machine learning to defend advanced persistent threats. **arXiv preprint arXiv:1802.00259**, 2018.

SETH, S.; SINGH, G.; CHAHAL, K. Drift-based approach for evolving data stream classification in intrusion detection system. In: **Proceedings of the Workshop on Computer Networks & Communications, Goa, India**. [S.l.: s.n.], 2021. p. 23–30.

SHARMA, D.; WASON, V.; JOHRI, P. Optimized classification of firewall log data using heterogeneous ensemble techniques. In: IEEE. **2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)**. [S.l.], 2021. p. 368–372.

SILVA, J. A. et al. Data stream clustering: A survey. **ACM Computing Surveys (CSUR)**, v. 46, n. 1, p. 13, 2013.

SOUZA, V. M. et al. Classification of evolving data streams with infinitely delayed labels. In: IEEE. **2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)**. [S.l.], 2015. p. 214–219.

SPINOSA, E. J.; CARVALHO, A. P. d. L. F. de; GAMA, J. Novelty detection with application to data streams. **Intelligent Data Analysis**, IOS Press, v. 13, n. 3, p. 405–422, 2009.

SPLUNK. How intrusion detection systems (ids) work: One part of your security arsenal. 2024. Accessed: 2024-07-02. Disponível em: <https://www.splunk.com/en_us/data-insider/what-is-an-ids-intrusion-detection-system.html>.

TANENBAUM, A.; FEAMSTER, N.; WETHERALL, D. **Redes de Computadores (coedição Bookman e Pearson)**. [S.l.]: Bookman Editora, 2021. ISBN 9788582605615.

TU, Y. Machine learning. In: **EEG Signal Processing and Feature Extraction**. Singapore: Springer, 2019.

UCAR, E.; OZHAN, E. The analysis of firewall policy through machine learning and data mining. **Wireless Personal Communications**, Springer, v. 96, p. 2891–2909, 2017.

VENDRAMIN, L.; CAMPELLO, R. J.; HRUSCHKA, E. R. Relative clustering validity criteria: A comparative overview. **Statistical analysis and data mining: the ASA data science journal**, Wiley Online Library, v. 3, n. 4, p. 209–235, 2010.

WIDMER, G.; KUBAT, M. Learning in the presence of concept drift and hidden contexts. **Machine learning**, Springer, v. 23, p. 69–101, 1996.

WINDING, R.; WRIGHT, T.; CHAPPLE, M. System anomaly detection: Mining firewall logs. In: IEEE. **2006 Securecomm and Workshops**. [S.l.], 2006. p. 1–5.

YANG, Y. et al. Astream: Data-stream-driven scalable anomaly detection with accuracy guarantee in iiot environment. **IEEE Transactions on Network Science and Engineering**, IEEE, 2022.

YUAN, X. et al. A concept drift based ensemble incremental learning approach for intrusion detection. In: IEEE. **2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)**. [S.l.], 2018. p. 350–357.

ZHANG, T.; RAMAKRISHNAN, R.; LIVNY, M. Birch: An efficient data clustering method for very large databases. In: **ACM Sigmod Record**. [S.l.]: ACM, 1996. v. 25, p. 103–114.

ZUBAROĞLU, A.; ATALAY, V. Data stream clustering: a review. **Artificial Intelligence Review**, Springer, v. 54, n. 2, p. 1201–1236, 2021.