UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Gabriel José Bueno Otsuka

# GRASP-HPO: Otimização de Hiperparâmetros baseada na meta-heurística GRASP para a detecção de intrusões

**Uberlândia, Brasil**

**2024**

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Gabriel José Bueno Otsuka

# GRASP-HPO: Otimização de Hiperparâmetros baseada na meta-heurística GRASP para a detecção de intrusões

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Ciência da Computação.

Orientador: Rodrigo Sanches Miani

Coorientador: Silvio Ereno Quincozes

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Bacharelado em Ciência da Computação

Uberlândia, Brasil

2024

Gabriel José Bueno Otsuka

# GRASP-HPO: Otimização de Hiperparâmetros baseada na meta-heurística GRASP para a detecção de intrusões

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Ciência da Computação.

Trabalho aprovado. Uberlândia, Brasil, 21 de novembro de 2024:

---

**Rodrigo Sanches Miani**
Orientador

---

**Paulo Henrique Ribeiro Gabriel**

---

**Kil Jin Brandini Park**

Uberlândia, Brasil
2024

# Agradecimentos

Agradeço primeiramente ao meu pai, Moisés, e à minha mãe, Magaly, pelo incentivo aos estudos e pelo apoio nas decisões que me permitiram alcançar a conclusão da graduação. Aos meus irmãos, Breno, Bruno, e Alana, que, com exemplos, me ensinaram a importância da bondade e do esforço.

Minha sincera gratidão à minha esposa, cujo apoio e compreensão foram fundamentais nos momentos mais desafiadores. Agradeço por sua paciência e amor, que se mostraram essenciais ao longo desse percurso.

Expresso também meu reconhecimento aos colegas de equipe da Universidade de Pittsburgh, Shinwoo Kim, Enoch Li, Jack Bellamy, Zi Han Ding, Zane Kissel e ao professor Daniel Mossé, pela oportunidade de colaborar no projeto *Capstone*, que abrange a proposta deste trabalho. Suas contribuições e orientações foram de grande valor para o desenvolvimento deste estudo.

Aos amigos de turma, Lorena Elias, Rodrigo Zamboni, Lucas Nascimento, Miguel Sanches e Marcos Felipe, com quem compartilhei os desafios de diversas disciplinas, agradeço pela parceria, motivação e aprendizado conjunto ao longo desses anos.

Por fim, agradeço aos professores da Faculdade de Computação da UFU, em especial aos professores Silvio Ereno Quincozes e Rodrigo Sanches Miani, por sua disposição em esclarecer dúvidas e oferecer sugestões engrandecedoras, compartilhando seu conhecimento e facilitando o percurso até a defesa do Trabalho de Conclusão de Curso.

# Resumo

Este trabalho apresenta o Procedimento de Busca Adaptativa Gulosa Randomizada para Otimização de Hiperparâmetros (GRASP-HPO), uma abordagem para Otimização de Hiperparâmetros (HPO) de algoritmos de aprendizado de máquina (AM), aplicado especificamente no contexto de Sistemas de Detecção de Intrusão (IDS). Baseado na metaheurística GRASP, o processo envolve uma seleção aleatória inicial e uma busca local subsequente para refinar as melhores combinações de hiperparâmetros (HPs). Aplicado ao algoritmo XGBoost treinado com o conjunto de dados ERENO, o GRASP-HPO alcançou até 94% de F1-Score em detecção de intrusões, com resultados competitivos em desempenho e tempo de convergência em comparação com abordagens de HPO tradicionais. Esses resultados destacam o GRASP-HPO como uma técnica promissora para aplicações de segurança cibernética e otimização de parâmetros em problemas de alta complexidade.

**Palavras-chave**: Aprendizado de máquina, Sistemas de Detecção de Intrusão, Hiperparâmetros, Otimização de Hiperparâmetros, Procedimento de Pesquisa Adaptativa Randomizada Gulosa

# Lista de ilustrações

# Lista de tabelas

# Lista de abreviaturas e siglas

**AM** aprendizado de máquina

**DDoS** Negação de Serviço Distribuída, do inglês, *Distributed Denial of Service*

**DoS** Negação de Serviço, do inglês, *Denial of Service*

**IDS** Sistema de Detecção de Intrusão, do inglês, *Intrusion Detection System*

**HP** hiperparâmetro

**HPO** otimização de hiperparâmetros

**GRASP** Procedimento de Pesquisa Adaptativa Gulosa Randomizada, do inglês, *Greedy Randomized Adaptive Search Procedure*

**GRASP-FS** Procedimento de Pesquisa Adaptativa Gulosa Randomizada para Seleção de Recursos, do inglês, *Greedy Randomized Adaptive Search Procedure for Feature Selection*

**GRASP-HPO** Procedimento de Pesquisa Adaptativa Gulosa Randomizada para Otimização de Hiperparâmetros, do inglês, *Greedy Randomized Adaptive Search Procedure for Hyperparameter Optimization*

**NIDS** Sistema de Detecção de Intrusão baseado em Redes, do inglês, *Network-based Intrusion Detection System*

**HIDS** Sistema de Detecção de Intrusão baseado em *Host*, do inglês, *Host-based Intrusion Detection System*

**IED** Dispositivos Eletrônicos Inteligentes, do inglês, *Intelligent Electronic Device*

**XGBoost** Aumento de gradiente extremo, do inglês, *eXtreme Gradient Boosting*

**HOAG** Otimização de Hiperparâmetro com Gradiente Aproximado, do inglês, Hyperparameter Optimization with Approximate Gradient

**SMBO** Otimização Baseada em Modelo Sequencial, do inglês, *Sequential Model-Based Optimization*

**HPO-PSO** Otimização de Hiperparâmetro com Otimização de Enxame de Partículas, do inglês, *Hyperparameter Optimization with Particle Swarm Optimization*

**UML** Linguagem de Modelagem Unificada, do inglês, *Unified Modeling Language*

**BOHB** Otimização Bayesiana com Hiperbanda, do inglês, *Bayesian Optimization with Hyperband*

# Sumário

# 1 Introdução

À medida que a digitalização se intensifica em vários setores da sociedade, as ameaças cibernéticas também se tornam mais preocupantes para indivíduos e organizações. Essas ameaças podem causar danos significativos, como evidenciado pelo ataque à empresa *Binance*, que resultou em uma perda de US$ 570 milhões (HOWCROFT, 2022). Nesse contexto, os Sistema de Detecção de Intrusão, do inglês, *Intrusion Detection System* (IDS) surgem como ferramentas essenciais na segurança da Tecnologia da Informação, desempenhando um papel crítico ao identificar precocemente tentativas de invasão, permitindo a adoção de medidas preventivas antes da ocorrência de danos.

Diante do aumento da complexidade dos ataques cibernéticos e do volume de informações disponíveis, os IDSs baseados em aprendizado de máquina (AM) tornaram-se particularmente valiosos. Conforme apontado por Hamid, Sugumaran e Balasaraswathi (2016), esses sistemas utilizam algoritmos que aprendem a partir do treinamento com grandes conjuntos de dados, buscando identificar padrões que permitem a detecção e classificação de ataques cibernéticos, o que torna esses sistemas uma escolha robusta contra as estratégias de intrusão que evoluem rapidamente.

Neste contexto, estudos recentes enfatizam a importância de uma configuração otimizada dos hiperparâmetros (HPs) em algoritmos de AM, ressaltando o impacto direto dessa configuração no desempenho dos modelos (YANG; SHAMI, 2020). Esse processo, denominado otimização de hiperparâmetros (HPO), é um problema de otimização combinatória, que envolve a busca pela combinação ideal de valores dos HPs que maximizam uma métrica específica resultante do treinamento dos algoritmos (BERGSTRA; BENGIO, 2012).

Diante disso, o Procedimento de Pesquisa Adaptativa Gulosa Randomizada, do inglês, *Greedy Randomized Adaptive Search Procedure* (GRASP) se destaca como uma abordagem robusta para resolver problemas de otimização combinatória. Essa meta-heurística opera em duas fases iterativas: a fase construtiva e a fase de busca local. Na fase construtiva, o GRASP utiliza uma estratégia gulosa (*greedy*) para gerar uma solução inicial promissora, priorizando combinações mais vantajosas de cada iteração. Em seguida, na fase de busca local, essa solução inicial é refinada para encontrar um ótimo local, explorando combinações próximas para melhorias adicionais. A simplicidade e a flexibilidade dessa abordagem permitem sua adaptação a diversos problemas de otimização, incluindo HPO.

Diversas meta-heurísticas já foram aplicadas ao problema de HPO, como a *Tabu Search* e *Simulated Annealing*. A *Tabu Search*, por exemplo, explora o espaço de soluções

de forma iterativa e evita retornar às soluções já visitadas, sendo comumente usada para resolver problemas de busca combinatória. No entanto, sua aplicação em *HPO* encontra limitações, especialmente em cenários com grandes espaços de busca, devido ao custo computacional para gerenciar a lista tabu e ao risco de convergência para ótimos locais, o que limita a exploração de novas regiões do espaço de busca (GENDREAU; POTVIN, 2005). Já o *Simulated Annealing* realiza uma busca estocástica inspirada no processo de resfriamento de materiais, mas sua convergência pode ser lenta em problemas complexos, como o HPO, em que uma quantidade significativa de combinações de HPs deve ser explorada (KIRKPATRICK; GELATT; VECCHI, 1983).

Em comparação, o GRASP é reconhecido por sua eficiência em encontrar soluções com menor risco de convergência prematura, pois a fase construtiva permite uma exploração inicial ampla, e a fase de busca local refina as melhores soluções de forma iterativa (RESENDE; RIBEIRO, 2016). Essa eficácia sugere que o GRASP também pode ser promissor para resolver o problema de HPO, oferecendo uma abordagem equilibrada entre exploração e intensificação da busca, características que são essenciais para lidar com o espaço multidimensional dos HPs.

Um exemplo específico da aplicabilidade do GRASP em problemas de otimização combinatória é o Procedimento de Pesquisa Adaptativa Gulosa Randomizada para Seleção de Recursos, do inglês, *Greedy Randomized Adaptive Search Procedure for Feature Selection* (GRASP-FS), que fornece conjuntos de características selecionadas, potencializando a eficiência dos IDSs e garantindo maior exatidão nas respostas desses sistemas através da redução de falsos positivos e negativos (QUINCOZES et al., 2020). Essa aplicação do GRASP à seleção de características evidencia sua versatilidade e seu potencial para abordar uma ampla gama de desafios de otimização em IDSs, indicando que o método pode ser aplicável para abordar outros problemas de otimização, como a HPO. No entanto, a eficácia do GRASP para HPO no contexto de IDSs ainda precisa ser validada, sendo esta a proposta investigada neste estudo.

## 1.1 Objetivos

Este trabalho tem como objetivo desenvolver uma nova metodologia de HPO baseada na meta-heurística GRASP, denominada Procedimento de Pesquisa Adaptativa Gulosa Randomizada para Otimização de Hiperparâmetros, do inglês, *Greedy Randomized Adaptive Search Procedure for Hyperparameter Optimization* (GRASP-HPO), visando automatizar a configuração dos HPs e maximizar assertividade na detecção de intrusões, especificamente melhorando o *F1 Score* do modelo ao ser testado com novos dados de IDSs. As principais contribuições deste trabalho são listadas a seguir:

- Propor uma plataforma para comparação dos resultados de diferentes algoritmos de

otimização;

- Desenvolver um novo algoritmo, o GRASP-HPO, baseado na meta-heurística GRASP, para otimizar HPs em IDSs baseados em AM;
- Aplicar o GRASP-HPO no contexto de IDSs;
- Avaliar o desempenho do GRASP-HPO em um contexto de detecção de intrusões, comparando-o com os métodos existentes no estado da arte.

## 1.2 Justificativa

A HPO é uma etapa crucial no desenvolvimento de modelos de AM, sendo responsável por determinar uma configuração otimizada dos parâmetros que regem o comportamento dos algoritmos de AM. As técnicas de HPO do estado da arte enfrentam desafios significativos, incluindo o alto custo computacional e exigências específicas quanto aos tipos de HP, além da necessidade de conhecimento extenso na área de AM como apontado na pesquisa de Yang e Shami (2020). Diante dessas limitações, o GRASP-HPO surge como uma alternativa promissora. Essa metodologia simplifica o processo de HPO por meio de uma abordagem simplificada, além de ser altamente modular, permitindo a integração de novas estratégias de otimização em cada uma de suas fases. O GRASP-HPO supera as restrições relacionadas aos tipos de HPs presentes em algumas das estratégias tradicionais de HPO, podendo ser aplicado a uma ampla variedade de algoritmos de AM.

## 1.3 Organização da Monografia

Esta monografia está organizada em cinco capítulos principais. Após esta introdução, o Capítulo 2, "Referencial Teórico", apresenta os conceitos fundamentais relacionados à HPO e IDSs, além de revisar o estado da arte e trabalhos relacionados, oferecendo a base teórica para o trabalho. O Capítulo 3, "Metodologia", detalha as etapas realizadas na pesquisa, detalhando as técnicas e processos empregados para a obtenção dos resultados. O Capítulo 4, "Resultados Experimentais", analisa o desempenho das técnicas aplicadas, analisa os dados obtidos e discute as interpretações dos resultados no contexto de HPO e IDSs. Finalmente, o Capítulo 5, "Conclusão", sintetiza as principais contribuições do trabalho, aponta limitações observadas e sugere possíveis direções para trabalhos futuros. O texto é encerrado com as referências bibliográficas que sustentam os argumentos e métodos discutidos ao longo do trabalho.

# 2 Referencial Teórico

Neste capítulo serão apresentados os conceitos fundamentais para a compreensão do desenvolvimento do trabalho, assim como trabalhos relacionados à HPO e à meta-heurística GRASP que representam o estado da arte.

## 2.1 Conceitos Fundamentais

Nesta seção será apresentada a base teórica que sustenta a pesquisa, explorando os principais conceitos que fundamentam o estudo de IDSs e algoritmos de HPO. Serão detalhados os princípios operacionais dos IDSs e sua importância no contexto da segurança cibernética, assim como a definição do problema de HPO e as diferentes técnicas do estado da arte que buscam solucioná-lo para aprimorar o desempenho dos algoritmos de AM.

### 2.1.1 Ataques cibernéticos

Como mencionado no Capítulo 1, o aumento da digitalização e conectividade tem gerado uma crescente ameaça de ataques cibernéticos, caracterizados por tentativas de exploração, acesso não autorizado, interrupção ou destruição de sistemas da informação. Tais ameaças comprometem os principais mecanismos de segurança da informação, incluindo identificação, autenticação, autorização, integridade, confidencialidade, não repúdio e disponibilidade. Esses mecanismos são essenciais para proteger os sistemas, garantindo a confiabilidade, o controle de acesso e a proteção contra acessos indevidos ou alterações não autorizadas. Os ataques cibernéticos podem ser classificados em diversas categorias, dependendo da técnica utilizada e do objetivo do atacante. De acordo com Biju, Gopal e Prakash (2019), os mais comuns são:

- Ataques de força bruta: O atacante tenta adivinhar senhas ou chaves de criptografia por meio de tentativas exaustivas sobre possíveis combinações de caracteres, buscando acesso não autorizado a qualquer sistema com mecanismo de autenticação e autorização.
- Negação de Serviço, do inglês, *Denial of Service* (DoS) e Negação de Serviço Distribuída, do inglês, *Distributed Denial of Service* (DDoS): Esses ataques buscam sobrecarregar sistemas ou redes, tornando-os indisponíveis para usuários legítimos, através do envio massivo de requisições ao sistema atacado. A diferença entre eles é que o DoS consiste em um sistema atacando outro, enquanto o DDoS é um conjunto de sistemas atacando outro.

- Injeção de *SQL* e *XSS*: Ambas exploram a falta de sanitização adequada de dados de entrada em aplicativos, permitindo que invasores insiram comandos maliciosos. A injeção de *SQL* compromete sistemas que interagem com bancos de dados, o que pode resultar na manipulação de informações, acessos não autorizados ou alterações nos dados. O *Cross-Site Scripting* (*XSS*) afeta aplicativos web ao executar *scripts* maliciosos no navegador das vítimas, possibilitando o roubo de informações sensíveis, o sequestro de sessões ou outros ataques contra os usuários.

- *Malware*: Engloba uma ampla gama de programas maliciosos, incluindo vírus e *ransomwares*, que podem comprometer a integridade de sistemas e dados.

- *Phishing*: Ataque no qual o invasor utiliza técnicas de engenharia social para induzir usuários a fornecer informações confidenciais, como senhas e dados bancários.

### 2.1.2 Sistemas de Detecção de Intrusão:

Os IDSs são componentes de segurança cibernética projetados para identificar atividades maliciosas e comportamentos anômalos em sistemas computacionais (SCARFONE, 2007). Para tal, esse tipo de sistema monitora o tráfego de rede ou atividades em *hosts* específicos para identificar potenciais incidentes de segurança, como ataques cibernéticos, violações de políticas internas, acesso não autorizado e outras atividades suspeitas. Ao detectar possíveis ameaças, o IDS notifica administradores ou, em alguns casos, toma ações automatizadas para impedir o ataque, dependendo do tipo e da configuração do sistema. Os IDSs são caracterizados em duas principais abordagens, cada uma com suas aplicações específicas (DAS; SARKAR, 2014):

- Sistema de Detecção de Intrusão baseado em Redes, do inglês, *Network-based Intrusion Detection System* (NIDS): analisa o tráfego de rede para identificar atividades suspeitas entre dispositivos conectados. É eficiente para monitorar sistemas altamente distribuídos, mas apresenta limitações na detecção de atividades maliciosas locais.

- Sistema de Detecção de Intrusão baseado em *Host*, do inglês, *Host-based Intrusion Detection System* (HIDS): monitora atividades de um *host*, que, por dispor de dados detalhados de dispositivos específicos, torna-se eficaz na detecção de intrusões localmente, mas é limitado em ambientes distribuídos.

#### 2.1.2.1 Métodos de Detecção em IDS

Os métodos de detecção de intrusões podem ser classificados em duas abordagens principais, a baseada em assinaturas e a baseada em anomalias. Na primeira, identifica-se atividades maliciosas comparando o tráfego de rede ou atividades de *hosts* com uma base

de dados de assinaturas de ataques conhecidos. Embora esta abordagem seja altamente eficaz contra ameaças previamente catalogadas, apresenta limitações significativas. Além de não detectar novas ameaças sem assinaturas registradas, pode ser contornada por atacantes que mascaram ou ofuscam suas ações para evitar a correspondência com os padrões conhecidos (ÜRÜN; SÖNMEZ, 2024). Já na segunda abordagem, o IDS cria um modelo de comportamento normal do sistema ou rede e identifica como intrusão qualquer desvio significativo desse padrão. Com essa abordagem, a detecção de ameaças desconhecidas torna-se possível, porém um dos desafios dessa estratégia é a alta taxa de falsos positivos, já que atividades legítimas também podem desviar do comportamento padrão de um sistema (LIPPMANN et al., 2000).

#### 2.1.2.2 Evolução dos IDSs

Com a crescente sofisticação das ameaças cibernéticas, os IDSs também evoluíram para incorporar técnicas avançadas, como AM e análise de comportamento, visando aprimorar a proteção dos dados e reduzir as taxas de falsos positivos (ZHANG; JIANG, 2011). No entanto, o estudo de Khraisat et al. (2019) aponta que a implementação de técnicas baseadas em AM introduz novos desafios, como a necessidade de dados rotulados e os altos custos computacionais. Além disso, as técnicas de evasão, como a fragmentação de pacotes e o uso de tráfego criptografado, dificultam a detecção por IDSs tradicionais, reforçando a necessidade de desenvolver IDSs robustos que se adaptem a novos padrões de ataques e da atualização constante de assinaturas de ataques conhecidos.

### 2.1.3 Aprendizado de Máquina

O uso de AM tem ganhado destaque em diversas áreas, incluindo segurança cibernética, devido à sua capacidade de identificar padrões complexos em grandes volumes de dados. Em AM, algoritmos aprendem a partir de conjuntos de dados e buscam reconhecer padrões que ajudam a prever eventos ou classificar informações. Dada a natureza dos ataques cibernéticos, que frequentemente envolvem padrões complexos e imprevisíveis de informações de rede ou de um dispositivo, o AM é uma escolha eficaz para detectar intrusões nesses ambientes (BUCZAK; GUVEN, 2016).

Os algoritmos de AM são geralmente classificados em três categorias distintas, dependendo da forma como os dados para treinamento são estruturados:

- Aprendizado Supervisionado: Nesta categoria, os modelos são treinados a partir de dados rotulados, ou seja, cada instância de dados possui uma resposta conhecida associada, o que permite ao modelo aprender padrões específicos para previsão futura. Essa abordagem é eficaz em tarefas de classificação e regressão, nas quais o objetivo é prever uma classe ou valor numérico para uma nova instância. No con-

texto de IDSs, esses modelos podem ser treinados com dados históricos de ataques, tornando possível a classificação de novos eventos como "normais" ou "maliciosos", ou até mesmo identificar o tipo específico de ataque ao qual o sistema pode estar exposto (A et al., 2021).

- Aprendizado Não Supervisionado: O aprendizado não supervisionado opera sem dados rotulados. Neste caso, os algoritmos buscam estruturar os dados a partir de padrões de comportamentos observados nos dados. No contexto de IDSs, o aprendizado não supervisionado, utilizando técnicas como agrupamento e redução de dimensionalidade, é útil na identificação de anomalias e na detecção de desvios comportamentais que podem indicar uma possível intrusão, como mostra Kumari et al. (2016).

- Aprendizado Semi-Supervisionado: O aprendizado semi-supervisionado é uma abordagem que mescla as duas técnicas anteriores, ou seja, o modelo é treinado com uma combinação de dados rotulados e não rotulados. Essa abordagem é valiosa quando há escassez de dados rotulados, ou para baratear os custos de obtenção desses dados, demonstrando bons resultados no contexto de IDSs, pois permite rápida adaptação a novos tipos de ameaças (HARA; SHIOMOTO, 2020).

A escolha do modelo ideal para um IDS depende de diversos fatores, incluindo a quantidade de dados disponíveis, o nível de rotulagem e a complexidade computacional.

#### 2.1.3.1 Métrica F1-Score

O F1-Score é uma métrica de avaliação amplamente utilizada em problemas de classificação, sendo definida como a média harmônica entre a precisão e a revocação, o que equilibra essas duas métricas em uma única medida sobre desempenho do modelo.

- Precisão: Esta métrica mensura a proporção de previsões positivas corretas (verdadeiros positivos) em relação ao total de previsões positivas feitas pelo modelo.

$$\text{Precisão} = \frac{\text{Verdadeiro Positivo}}{\text{Verdadeiro Positivo} + \text{Falso Positivo}} \tag{2.1}$$

- Revocação: Também conhecida como sensibilidade, calcula a proporção de verdadeiros positivos em relação ao total de instâncias reais positivas.

$$\text{Revocação} = \frac{\text{Verdadeiro Positivo}}{\text{Verdadeiro Positivo} + \text{Falso Negativo}} \tag{2.2}$$

- F1-Score: Essa métrica combina precisão e revocação em uma média harmônica, fornecendo uma medida equilibrada do desempenho do modelo. Valores de F1-Score próximos de 1 indicam um bom desempenho, caracterizado por uma alta capacidade

de identificar corretamente as classes positivas (alta revocação) e um baixo índice de falsos positivos (alta precisão). A média harmônica é usada para garantir um valor alto somente quando ambas as métricas são elevadas, penalizando discrepâncias significativas entre precisão e revocação.

$$\text{F1Score} = 2 \times \frac{\text{Precisão} \times \text{Revocação}}{\text{Precisão} + \text{Revocação}} \tag{2.3}$$

#### 2.1.3.2 Sobreajuste no Aprendizado de Máquina

Sobreajuste, do inglês *overfitting*, é um problema comum a ser lidado nos algoritmos de AM. Ele ocorre quando um modelo se ajusta excessivamente aos dados de treinamento, capturando tanto os padrões gerais quanto os ruídos e particularidades desses dados. Como consequência, o modelo apresenta uma alta acurácia nos dados de treinamento, mas perde a capacidade de generalizar adequadamente para dados novos, o que resulta em um desempenho ruim do modelo (YING, 2019).

O sobreajuste pode ser causado por diversos fatores, entre eles a complexidade excessiva do modelo e o pequeno volume de dados de treinamento. Modelos com alta capacidade, como redes neurais profundas e árvores de decisão sem restrições são mais propensos ao sobreajuste, pois conseguem representar não apenas relações complexas, mas também ruídos presentes nos dados de treinamento.

Em algoritmos de AM, diversas abordagens são aplicadas para mitigar o sobreauste, incluindo:

- Regularização: Técnicas de regularização, como L1 (lasso) e L2 (*ridge*), penalizam a complexidade do modelo ao adicionar um termo de regularização à função de custo, ajudando a evitar o ajuste excessivo.
- Validação cruzada: A validação cruzada divide os dados em subconjuntos de treino e validação, permitindo que o modelo seja testado em diferentes partes dos dados.
- Redução da complexidade do modelo: Ajustar HPs (mais detalhado em 2.1.5), como profundidade máxima em árvores de decisão ou o número de neurônios em redes neurais, pode limitar a capacidade do modelo de aprender detalhes irrelevantes.
- Aumento do volume de dados: Um maior volume de dados geralmente melhora a capacidade de generalização do modelo, especialmente se os dados adicionais forem diversificados e representativos das condições reais de uso.

Além das técnicas mencionadas, é essencial que os dados de teste permaneçam completamente separados do processo de treinamento e validação. Isso significa que, durante o desenvolvimento do modelo, apenas os dados de treinamento e validação devem

ser utilizados para ajuste de HPs, seleção de características e otimização do modelo. Os dados de teste são utilizados exclusivamente ao final, para uma avaliação definitiva do desempenho do modelo em um conjunto de dados que ele não viu anteriormente. Essa prática reduz o risco de sobreajuste, proporcionando uma avaliação mais confiável do modelo (BISHOP; NASRABADI, 2006).

### 2.1.4 Conjuntos de dados

Em AM, a escolha do conjunto de dados é um fator crucial para o desenvolvimento, treinamento e validação de modelos. Conjuntos de dados fornecem a base para que os algoritmos identifiquem padrões, ajustem parâmetros e sejam avaliados em cenários reais ou simulados (BISHOP; NASRABADI, 2006). Nesta seção, são apresentados diferentes tipos de conjuntos de dados que foram utilizados ao longo do trabalho.

#### 2.1.4.1 *Toy Datasets*

*Toy Datasets* (GALLATIN; ALBON, 2023) são conjuntos de dados simples e de pequeno porte, amplamente utilizados em AM para testes rápidos e validações de modelos e algoritmos. Esses conjuntos de dados são caracterizados pelo número limitado de amostras, o que facilita a experimentação e detecção de possíveis erros de configuração antes de serem aplicados em conjuntos mais complexos e específicos. No contexto deste trabalho, os *toy datasets* são utilizados para verificar a funcionalidade básica e a implementação dos diversos algoritmos de HPO, garantindo que eles operem conforme o esperado.

#### 2.1.4.2 Conjunto de dados ERENO IEC-61850

O ERENO IEC-61850 (QUINCOZES et al., 2023) é um conjunto de dados sintético realista, desenvolvido para avaliar a detecção de intrusão em redes elétricas. Ele foi gerado através da plataforma "*Efficacious Reproducer Engine for Network Operations*" (ERENO), que simula comportamentos normais e cenários de ataques contra Dispositivos Eletrônicos Inteligentes, do inglês, *Intelligent Electronic Devices* (IEDs).

O conjunto de dados ERENO é composto por uma ampla variedade de amostras que incluem tanto comportamentos normais quanto intrusões simuladas. Cada amostra possui 69 atributos e 1 rótulo que determina o cenário, indicando se é um estado ou um ataque contra IEDs. Os atributos cobrem informações detalhadas de rede e dados de *hosts*, refletindo as interações complexas entre dispositivos e sistemas em uma subestação elétrica.

A plataforma ERENO é uma ferramenta de código aberto projetada para produzir conjuntos de dados que se alinham com ambientes ciber-físicos em subestações elétricas digitais sob o padrão IEC-61850. O ERENO suporta a criação de recursos representativos,

incluindo atributos de nível de rede e de aplicação. Além disso, ele é capaz de simular o nível de processo e o nível de barramento da arquitetura IEC-61850, ilustrados na Figura 1. Esses níveis representam camadas distintas de comunicação e operação nas subestações elétricas:

- O nível de processo conecta diretamente os dispositivos de campo, como sensores e atuadores, ao sistema de controle, permitindo a troca de dados em tempo real para monitoramento e comando.
- O nível de barramento interliga os IEDs dentro de uma área específica da subestação, fornecendo comunicação e coordenação local entre equipamentos, como relés de proteção e disjuntores (QUINCOZES et al., 2021)

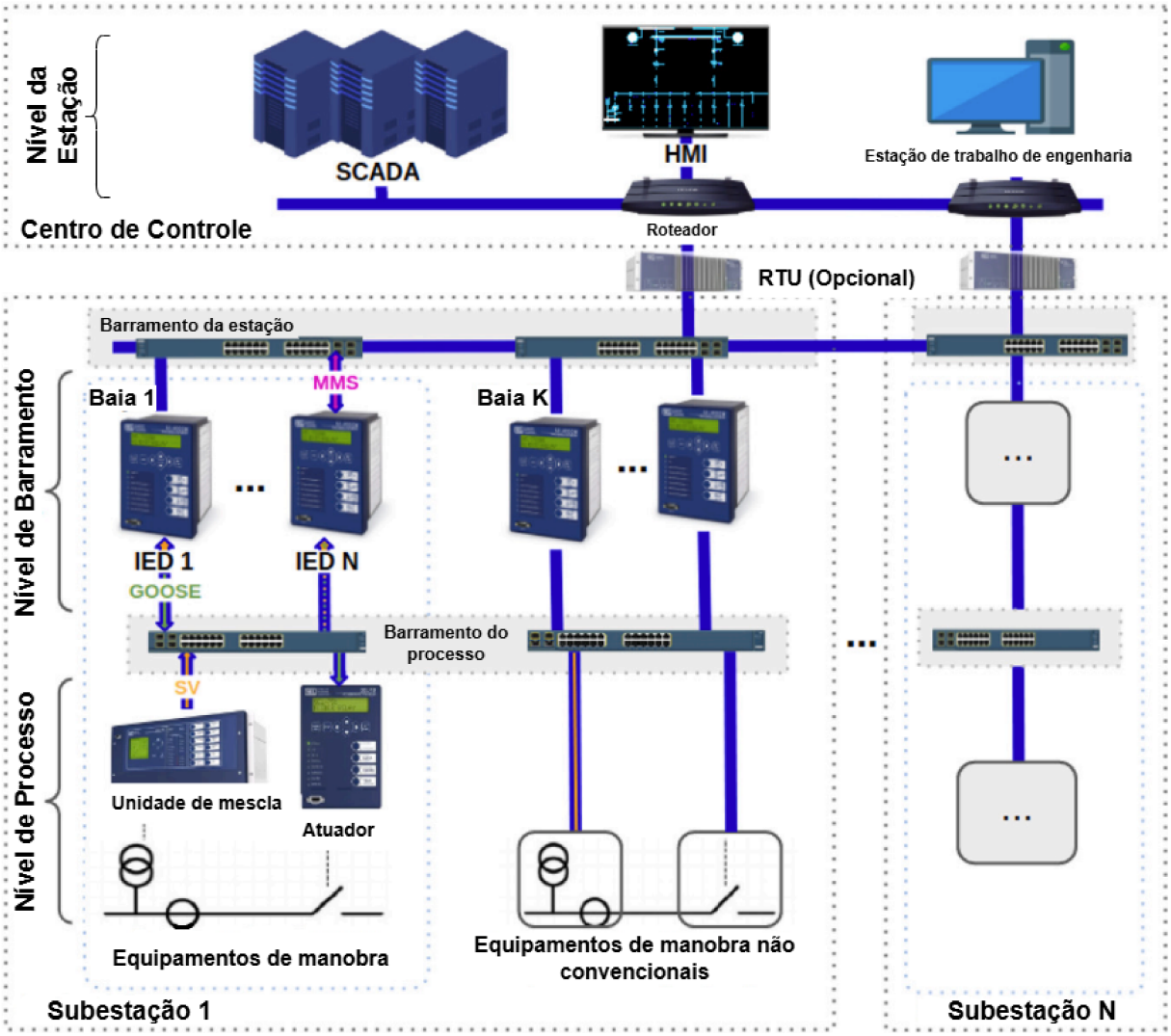


Figura 1 – Arquitetura IEC-61850 convencional. Fonte: Adaptado de Quincozes et al. (2021).

A Tabela 1 descreve exemplifica cada atributo de uma possível instância do conjunto de dados ERENO:

Tabela 1 – Exemplo de instância do conjunto de dados ERENO

| **Atributo** | **Descrição** | **Valor (exemplo)** |
|---|---|---|
| Time | Tempo | 0.01659 |
| isbA | Corrente da fase A | -0.020875638 |
| isbB | Corrente da fase B | 0.080344565 |
| isbC | Corrente da fase C | -0.059088424 |
| vsbA | Tensão da fase A | 0.036322135 |
| vsbB | Tensão da fase B | -0.019330863 |
| vsbC | Tensão da fase C | -0.016977683 |
| isbARmsValue | Valor RMS da corrente da fase A | 0.014291056823441546 |
| isbBRmsValue | Valor RMS da corrente da fase B | 0.025666621172570556 |
| isbCRmsValue | Valor RMS da corrente da fase C | 0.03305609583634606 |
| vsbARmsValue | Valor RMS da tensão da fase A | 0.013977964797116878 |
| vsbBRmsValue | Valor RMS da tensão da fase B | 0.015224260349443944 |
| vsbCRmsValue | Valor RMS da tensão da fase C | 0.012394512349714532 |
| isbATrapAreaSum | Área acumulada da corrente A | -38.75249306135538 |
| isbBTrapAreaSum | Área acumulada da corrente B | -38.3308100382339 |
| isbCTrapAreaSum | Área acumulada da corrente C | -41.40358599379414 |
| vsbATrapAreaSum | Área acumulada da tensão A | -39.51286190971722 |
| vsbBTrapAreaSum | Área acumulada da tensão B | -39.874011174395605 |
| vsbCTrapAreaSum | Área acumulada da tensão C | -39.11321255369313 |
| t | Tempo relativo | 0.01659 |
| GooseTimestamp | Timestamp do pacote GOOSE | 0.016589999999999994 |
| SqNum | Número de sequência | 1 |
| StNum | Número de status | 0 |
| cbStatus | Status do circuito | 1 |
| frameLen | Comprimento do quadro Ethernet | 256 |
| ethDst | Endereço MAC de destino | 01:0c:cd:01:2f:81 |
| ethSrc | Endereço MAC de origem | 01:0c:cd:01:2f:80 |
| ethType | Tipo Ethernet | 0x88B8 |
| gooseTimeAllowedtoLive | Tempo permitido de vida GOOSE | 11000 |
| gooseAppid | ID da aplicação GOOSE | 0x00003001 |
| gooseLen | Comprimento da mensagem GOOSE | 242 |
| TPID | ID de prioridade do VLAN | 0x8100 |
| gocbRef | Referência do bloco GOOSE | LD/LLN0*GO*gcblA |
| datSet | Nome do conjunto de dados | LD/LLN0$IntLockA |
| goID | ID do pacote GOOSE | IntLockA |
| test | Indicador de teste | TRUE |
| confRev | Revisão de configuração | 1 |
| ndsCom | Comunicação de dados NDS | FALSE |

Tabela 1 – (Continuação da Tabela 1)

| **Atributo** | **Descrição** | **Valor (exemplo)** |
|---|---|---|
| numDatSetEntries | Número de entradas do conjunto | 25 |
| APDUSize | Tamanho da APDU | 242 |
| protocol | Protocolo | GOOSE |
| stDiff | Diferença de status | 0 |
| sqDiff | Diferença de sequência | 0 |
| gooseLengthDiff | Diferença no comprimento GOOSE | 0 |
| cbStatusDiff | Diferença no status do circuito | 0 |
| apduSizeDiff | Diferença no tamanho da APDU | 0 |
| frameLengthDiff | Diferença no comprimento do quadro | 0 |
| timestampDiff | Diferença no timestamp | 0.0 |
| tDiff | Diferença de tempo | 0.0 |
| timeFromLastChange | Tempo desde última mudança | -6.938893903907228E-18 |
| delay | Atraso | 8.010864194218925E-10 |
| @class@ | Classe | normal |

O padrão IEC-61850 (BAIGENT et al., 2004) define protocolos de comunicação e as normas de interoperabilidade para dispositivos em redes de energia. Ao adotar esse padrão, o ERENO oferece uma base de dados que reflete os comportamentos e interações reais entre os dispositivos de redes elétricas, aumentando a validade dos experimentos realizados para IDSs. A riqueza dos dados do ERENO permite que a técnica proposta seja testada em um ambiente complexo e relevante para a segurança cibernética, proporcionando uma base sólida para experimentos de HPO.

### 2.1.5 Otimização de Hiperparâmetros

Em modelos de AM, os parâmetros atualizados durante o processo de aprendizado são chamados de parâmetros do modelo, enquanto aqueles definidos antes da execução do algoritmo são conhecidos como HPs. Os HPs ajustam as características arquitetônicas do algoritmo e impactam significativamente a eficácia do aprendizado, levando a resultados mais precisos após o treinamento. Os valores ótimos de HPs dependem do modelo de AM utilizado, do tipo de HPs a serem otimizados e do conjunto de dados de treinamento (HUTTER; KOTTHOFF; VANSCHOREN, 2019).

Há diversas abordagens para a HPO na literatura atual, cada uma com seus prós e contras. Os resultados dessas técnicas dependem dos HPs a serem otimizados, como discutido na pesquisa de Yang e Shami (2020). Algumas dessas abordagens serão discutidas e comparadas nesta seção.

1. Algoritmos que não possuem modelo:

   Esta abordagem envolve a seleção de valores manualmente ou de forma randômica, sem considerar conhecimento prévio da função de otimização. Um exemplo é o *Grid Search*, um algoritmo de força bruta que avalia todas as combinações possíveis dentro de um intervalo especificado. Sua implementação é simples, porém frequentemente requer ajustes manuais do intervalo para alcançar um valor ótimo dos HPs (HUTTER; KOTTHOFF; VANSCHOREN, 2019).

2. Algoritmos baseados em vetor gradiente:

   Essas técnicas de HPO consideram o vetor gradiente da função de otimização dos HPs, alcançando o valor ótimo rapidamente. Apesar de sua eficiência, requerem que os HPs variem em valores contínuos. Caso a função objetivo não seja convexa, o resultado pode convergir para um máximo local. Um exemplo é o algoritmo *Hyperparameter Optimization with Approximate Gradient*, que utiliza um vetor gradiente aproximado para reduzir o custo computacional (PEDREGOSA, 2016).

3. Otimização *Bayesiana*:

   Este é um algoritmo iterativo comum, no qual informações de avaliações anteriores são usadas para definir os valores dos parâmetros nas iterações subsequentes. Isso reduz significativamente o número de iterações necessárias em comparação com abordagens sem modelo, apesar de ser uma técnica complexa. É adequada para funções não convexas, mas o custo computacional por iteração é considerável. Exemplos incluem o *Sequential Model Algorithm Configuration*, baseado em *Random Forest* (HUTTER; HOOS; LEYTON-BROWN, 2010), e o *HyperOpt*, que emprega o algoritmo de *Tree of Parzen Estimators* (KOMER; BERGSTRA; ELIASMITH, 2019).

4. Algoritmos de otimização de multifidelidade:

   Estes métodos visam economizar tempo e recursos quando a configuração dos HPs e/ou o conjunto de dados de treinamento são muito extensos, utilizando uma combinação de avaliações de baixa e alta fidelidade. Um exemplo notável é o *Hyperband*, que otimiza a alocação de recursos para as configurações mais promissoras (LI et al., 2018).

5. Algoritmos meta-heurísticos

   Essas são estratégias que exploram o espaço de busca de soluções de forma eficiente, balanceando a diversificação (exploração) e a intensificação (exploração aprofundada) dos valores (BECCENERI, 2008).

6. Algoritmos híbridos

Na vasta literatura de AM, propostas híbridas, como o Otimização Bayesiana com Hiperbanda, do inglês, *Bayesian Optimization with Hyperband* (BOHB), que combina otimização Bayesiana e multifidelidade, são comuns. Este método otimiza a busca por regiões promissoras de valores com eficiência de recursos, conforme demonstrado em (FALKNER; KLEIN; HUTTER, 2018).

O GRASP-HPO, que será desenvolvido neste trabalho, é um exemplo de algoritmo de HPO meta-heurístico e sua estrutura será detalhada na seção 2.1.7.

### 2.1.6 XGBoost

O Aumento de gradiente extremo, do inglês, *eXtreme Gradient Boosting* (XGBoost) (CHEN; GUESTRIN, 2016) é um algoritmo de AM baseado em árvores de decisão que se encaixa na classe de aprendizado supervisionado, sendo amplamente utilizado para tarefas de classificação e regressão. A seguir, seu funcionamento e características são apresentados.

#### 2.1.6.1 Funcionamento do Algoritmo XGBoost

O XGBoost utiliza o conceito de *boosting*, uma técnica de *ensemble* que cria um modelo preditivo forte a partir da combinação de vários modelos mais fracos. No caso deste algoritmo, os modelos fracos são árvores de decisão, que por si só, já possuem bons resultados no contexto de IDS (AHMIM et al., 2019). O *boosting* funciona treinando novas árvores sequencialmente, corrigindo os erros (diferença entre a predição atual e o valor real) das árvores anteriores, reforçando a precisão do modelo final, como demonstrado em Friedman (2001).

Além disso, o XGBoost minimiza uma função de perda (como *log loss* para classificação, ou erro quadrático médio para regressão) utilizando o gradiente descendente. Em cada iteração, ele calcula o gradiente da função de perda em relação às previsões atuais e ajusta os parâmetros da nova árvore para reduzir esse erro.

Para evitar o sobreajuste, o XGBoost emprega uma forte capacidade de regularização, incluindo L1 (*lasso*) e L2 (*ridge*). Outra característica importantes é sua busca otimizada para encontrar a melhor divisão de cada nó em uma árvore, com um cálculo do ganho para cada possível divisão. Isso permite que o XGBoost construa árvores com divisões mais informativas e melhora o desempenho global.

Por fim, o algoritmo apresenta alta eficiência graças à sua capacidade de paralelização durante a construção das árvores. Isso permite que ele seja aplicado a conjuntos de dados grandes e de alta dimensionalidade de forma eficiente.

#### 2.1.6.2 Principais hiperparâmetros (HPs) do XGBoost

O XGBoost possui diversos HPs importantes que influenciam seu desempenho:

- *n_estimators*: Número inteiro de árvores no *ensemble*. Aumentar esse valor pode melhorar o desempenho do modelo, no entanto aumenta o custo computacional e o risco de sobreajuste.
- *learning_rate*: Taxa de aprendizado que controla o impacto que cada nova árvore do *boosting* terá a partir da correção dos erros residuais da árvore anterior. Se este valor for muito alto, há risco de sobreajuste, e caso seja muito baixo, exigirá uma quantidade muito grande de árvores para alcançar um bom desempenho.
- *max_depth*: Número inteiro que representa a profundidade máxima das árvores. Profundidades maiores permitem que o modelo aprenda padrões mais complexos, mas também aumentam o risco de sobreajuste.
- *subsample*: Fração de amostras utilizadas em cada árvore. Valores menores ajudam a reduzir o sobreajuste ao treinar cada árvore em uma parte diferente dos dados.
- *colsample_bytree*: fração de recursos (variáveis) considerada em cada árvore. Esse parâmetro também ajuda a reduzir o sobreajuste ao limitar o número de variáveis usadas em cada árvore.

### 2.1.7 GRASP

A meta-heurística GRASP foi desenvolvida para resolver problemas de otimização combinatória, que apresentam alta complexidade devido à dificuldade de encontrar uma solução ótima em espaços de busca amplos. Essa técnica constrói soluções a partir de combinações de candidatos selecionados em um conjunto definido, avaliando-os por meio de uma função objetivo específica para o problema.

O GRASP opera em duas fases principais. Na fase de construção, os candidatos são escolhidos de forma aleatória, porém guiada por uma estratégia gulosa, que prioriza aqueles com maior potencial de contribuir para uma solução de alta qualidade, o que resulta em soluções iniciais diversificadas e promissoras. Na fase de busca local, essas soluções são refinadas por meio de pequenas alterações em suas vizinhanças, visando melhorias iterativas na qualidade (FEO; RESENDE, 1989).

O GRASP tem sido amplamente aplicado em diferentes áreas, como planejamento, roteamento de veículos e alocação de recursos, devido à sua flexibilidade e eficiência. A técnica também tem sido adaptada para o contexto de AM. Por exemplo, no GRASP-FS, a meta-heurística foi aplicada para selecionar atributos relevantes em conjuntos de dados

relacionados a tráfego de redes, melhorando o desempenho de classificadores na detecção de intrusões (QUINCOZES et al., 2020).

Neste trabalho, o GRASP foi adaptado para desenvolver a técnica GRASP-HPO, focada na otimização de HPs de algoritmos de classificação utilizados em IDSs. O objetivo do GRASP-HPO é ajustar os valores de HPs para maximizar métricas de desempenho, como o F1-Score, possibilitando a construção de modelos mais eficazes para a identificação de tráfego malicioso.

A principal diferença entre o GRASP-HPO e o GRASP-FS está no foco da otimização. Enquanto o GRASP-FS realiza a seleção de atributos para melhorar a qualidade dos dados de entrada de classificadores, o GRASP-HPO concentra-se na configuração dos parâmetros internos dos algoritmos de classificação, como taxa de aprendizado, número de árvores ou profundidade máxima em modelos como o XGBoost. Essa adaptação é particularmente relevante no contexto de IDSs, onde a escolha adequada de HPs tem impacto direto na eficácia e eficiência do sistema.

A escolha do GRASP como base para o GRASP-HPO justifica-se pela flexibilidade e robustez dessa meta-heurística. Sua capacidade de explorar eficientemente diferentes regiões do espaço de busca por meio de uma abordagem balanceada entre aleatoriedade e estratégia gulosa torna-a uma solução promissora para a otimização de HPs em modelos complexos de aprendizado de máquina.

## 2.2 Trabalhos Relacionados

Nesta seção são discutidas várias pesquisas significativas na área de HPO aplicadas à AM, focando em como cada uma contribuiu para o desenvolvimento da área e identificando oportunidades para inovação. Este trabalho propõe uma nova meta-heurística para HPO baseada em GRASP, aplicando-a ao algoritmo XGBoost no contexto de IDS com o conjunto de dados ERENO. Além disso, é desenvolvido uma plataforma de comparação de HPOs, o qual permite configurar conjuntos de dados e HPOs e visualizar os resultados comparativos de cada técnica ao otimizar o XGBoost.

Yang e Shami (2020) oferece uma visão geral sobre HPs em algoritmos de AM, comparando os resultados das principais técnicas de HPO em diferentes algoritmos. O estudo conclui que a escolha do método de otimização depende fundamentalmente do tipo de parâmetro e do intervalo de valores em que ele pode ser ajustado. Além disso, sugere como trabalho futuro a redução do tempo de avaliação em grandes conjuntos de dados, diminuindo o tempo de execução quando as faixas de valores dos HPs são extensas. Outra proposta é melhorar a escalabilidade dos métodos de HPO nas diversas bibliotecas e plataformas utilizadas no artigo. Diferentemente de Yang e Shami (2020), este trabalho propõe uma nova meta-heurística baseada em GRASP, aplicando-a diretamente

ao XGBoost e em um contexto de IDS, o que permite otimizações específicas para o tipo de dados de cibersegurança.

Kunang et al. (2021) propõe um método para a classificação de ataques usando *Deep Learning* em IDS, combinando *Grid Search* e *Random Search* para HPO. No entanto, conforme discutido anteriormente no item 1, da Seção 2.1.5, ambas técnicas de HPO são métodos de busca exaustiva e apresentam limitações de desempenho quando comparadas a outras estratégias baseadas em modelos. Além disso, os valores dos HPs considerados são específicos para o aprendizado profundo (e.g., número de neurônios e camadas, valor da taxa de aprendizado, pesos de inicialização, função de ativação e função de perda). Em contraste, o presente trabalho aplica GRASP ao XGBoost, um algoritmo baseado em árvores de decisão, visando uma otimização mais ampla e específica para técnicas baseadas em *boosting* e aplicável a contextos além de redes neurais profundas.

*Otimização de Hiperparâmetro com Gradiente Aproximado, do inglês, Hyperparameter Optimization with Approximate Gradient (HOAG)* HOAG (PEDREGOSA, 2016) utiliza informações aproximadas do vetor gradiente da função de custo de HPO. O HOAG permite o ajuste dos valores de HP antes da convergência total dos parâmetros do modelo. Utilizando O HOAG, os modelos convergem significativamente mais rápido do que ao utilizar *Grid Search*, *Random Search*, *SMBO* e *interdiff* em dois de três conjuntos de dados. No entanto, como acontece com outros algoritmos de otimização convexa, HOAG pode ser menos eficaz em funções não convexas ou não diferenciáveis, além de apresentar um alto custo computacional. Este trabalho explora uma abordagem meta-heurística alternativa baseada em GRASP, capaz de realizar ajustes de HP em um conjunto de dados voltado para IDS (ERENO), com uma configuração mais adaptável para funções não convexas, sem exigir gradientes.

O *AccSMBO*, uma melhoria na Otimização Baseada em Modelo Sequencial, do inglês, *Sequential Model-Based Optimization* (SMBO) para HPs, uma classe de otimizações Bayesianas, utiliza informações pré-conhecidas sobre os parâmetros (CHENG et al., 2019). O *AccSMBO* ajusta os valores observados dos HPs com valores de gradiente da função de desempenho, obtidos através da regressão do processo gaussiano, para economizar recursos computacionais. O *AccSMBO* também usa meta-aprendizado para criar uma nuvem de probabilidade de valores de HP, ajustando-os a intervalos com maior probabilidade de conter os melhores HPs. A aceleração do *AccSMBO*, em comparação com a metodologia HOAG, é de 1,4x. No entanto, o *AccSMBO* ainda requer que a função seja não convexa ou não diferenciável. Comparado ao *AccSMBO*, o GRASP-HPO proposto neste trabalho não depende de aproximações Bayesianas, mas sim de uma busca adaptativa e iterativa orientada a soluções promissoras, o que pode ser mais eficaz em cenários de dados de segurança de rede onde as distribuições podem ser menos previsíveis.

Uma abordagem de otimização multifidelidade (SEN; KANDASAMY; SHAK-

KOTTAI, 2018) aborda o tratamento de grandes conjuntos de dados e configurações HP organizando os dados em uma estrutura de árvore, com a raiz sendo o conjunto completo e cada nível dividindo os dados pela metade. Essa estrutura permite avaliar soluções de alta e baixa fidelidade simultaneamente, superando outras técnicas de otimização multifidelidade, embora opere com dados livres de ruído e esteja limitada a apenas uma dimensão de fidelidade. Em contraste, o presente trabalho utiliza uma abordagem de busca gulosa GRASP com múltiplos níveis de fidelidade adaptados ao conjunto de dados ERENO, com múltiplos tipos de ataque e variações intrínsecas, o que é mais desafiador e realista no contexto de IDS.

A Otimização de Hiperparâmetro com Otimização de Enxame de Partículas, do inglês, *Hyperparameter Optimization with Particle Swarm Optimization* (HPO-PSO) (AWATRAMANI; GUPTA, 2021) está na mesma classe da meta-heurística GRASP e explora a Busca Bayesiana para Busca de Arquitetura Neural, estimando a arquitetura ideal de uma rede neural artificial. O HPO-PSO mostrou-se altamente eficiente no treinamento, convergindo rapidamente a rede para um estado de perda mínima. Semelhante ao trabalho de (KUNANG et al., 2021), o HPO-PSO foi aplicado a um HP específico (a taxa de aprendizado da rede), e o otimizador usado para treinar a rede neural foi o *Stochastic Gradient Descent*. O HPO-PSO, no entanto, não foi treinado no contexto dos IDSs. Embora eficiente para aprendizado profundo, o HPO-PSO não foi testado em IDS e não se aplica diretamente a algoritmos baseados em árvores como o XGBoost, foco do presente trabalho. A abordagem GRASP-HPO busca, assim, suprir a lacuna na aplicação de HPO para modelos como o XGBoost em contextos de detecção de intrusão.

Esses estudos demonstram a diversidade de técnicas de HPO e a contínua necessidade de métodos mais eficazes e adaptáveis para diferentes algoritmos e conjuntos de dados. Este trabalho contribui com uma abordagem nova e prática, introduzindo a meta-heurística GRASP-HPO, testada no XGBoost e no conjunto de dados ERENO para IDS, além de uma plataforma de comparação que permite configurar e visualizar os resultados entre diferentes HPOs. Esta plataforma oferece uma visualização comparativa dos desempenhos dos HPOs, facilitando a análise dos resultados para cada conjunto de dados e técnica de HPO aplicada ao XGBoost, fornecendo detalhes sobre a eficiência e aplicabilidade de cada método.

# 3 Desenvolvimento

Este capítulo propõe uma abordagem integrada para HPO no contexto de IDSs. A proposta está estruturada em três principais contribuições interligadas, que serão detalhadas nas seções seguintes: o desenvolvimento do GRASP-HPO, uma meta-heurística para HPO; a construção de uma plataforma de comparação de HPOs, que executa as estratégias de HPO sob as mesmas condições de entrada e saída; e a avaliação do GRASP-HPO em cenários específicos de IDS a partir do treinamento do modelo com o conjunto de dados ERENO. Na Figura 2, cada um desses componentes e suas interconexões são ilustrados.

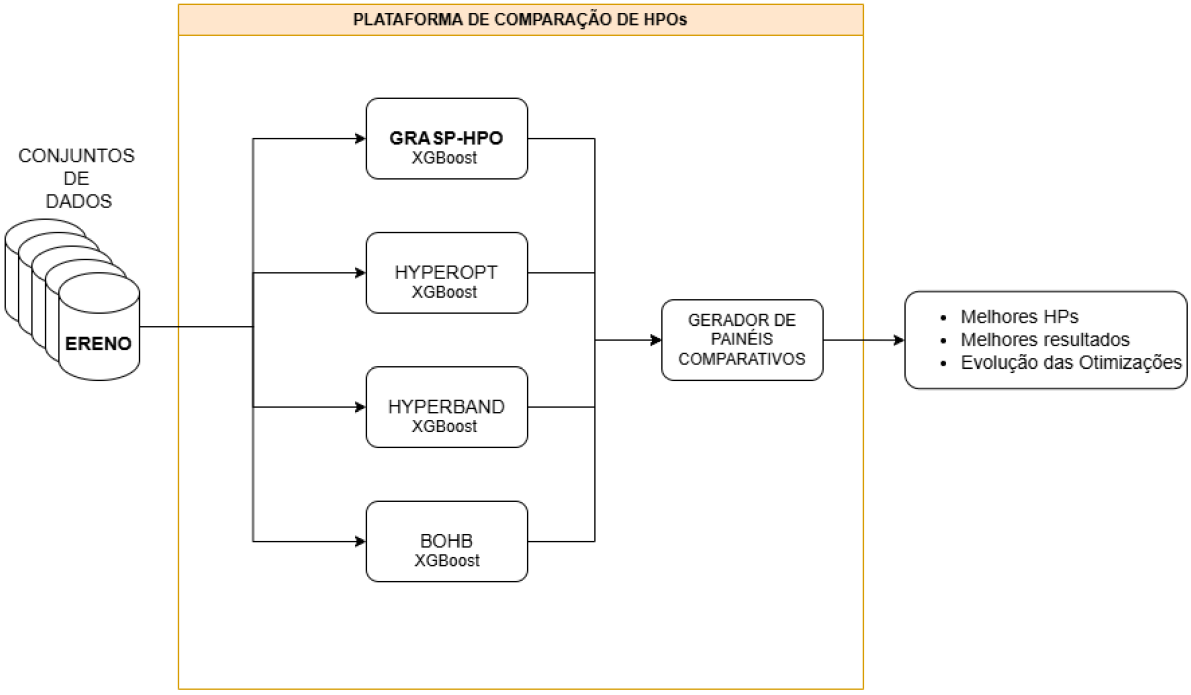


Figura 2 – Visão geral da proposta. Fonte: Do Autor.

## 3.1 Plataforma de comparação de HPOs

Nesta seção é introduzida uma nova estrutura para avaliar e comparar diferentes métodos de HPO. O algoritmo segue o paradigma da programação orientada a objetos e utiliza o Padrão de Projeto *Factory Method* (GAMMA, 1995) para encapsular a lógica de criação de objetos, promovendo maior flexibilidade e extensibilidade na adição de novos algoritmos de HPO e conjuntos de dados sem a necessidade de alterar diretamente o código principal. Essa estrutura possui fábricas de conjuntos de dados e de métodos de HPO, conforme descrito abaixo.

### 3.1.1 Fábrica de Conjuntos de Dados

O componente *Dataset Factory* foi projetado para facilitar a adição de novos conjuntos de dados à plataforma, definindo uma interface de contrato que estabelece os padrões de entrada e saída dos dados. Dessa forma, novos conjuntos de dados podem ser integrados para comparação sem a necessidade de refatoração do código principal da plataforma. Isso ocorre pois os pré-processamentos específicos de cada conjunto, como remoção de metadados, conversão de tipos de dados, carregamento de arquivos, entre outros são responsabilidades da sua classe concreta, que deve respeitar o contrato definido pela interface *InputDataset*. Este componente também tem um papel fundamental para a comparação justa dos resultados, garantindo a mesma divisão de subconjuntos de treinamento, validação e teste para todos os HPOs. Isso padroniza a avaliação dos modelos, assegurando que quaisquer diferenças observadas nos resultados sejam devidas às técnicas de HPO em si, e não à variações nos dados de entrada.

A estrutura da fábrica está descrita na Figura 3, além de demonstrar que para o trabalho atual, foram implementadas duas classes concretas de conjuntos de dados.

- `ERENO Dataset`: Pre processa os atributos do conjunto de dados ERENO (2.1.4.2), disponibilizando as amostras seguindo o padrão definido no contrato da interface para o processo de HPO.
- `Toy Datasets`: Garante a padronização do uso dos conjuntos de dados da biblioteca *sci-kit-learn* (2.1.4.1) para o processo de HPO.

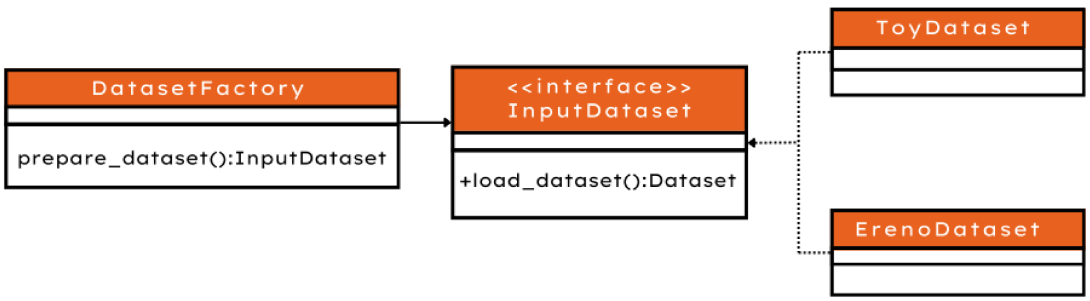


Figura 3 – Diagrama de UML da Fábrica de Conjuntos de dados. Fonte: do Autor.

### 3.1.2 Fábrica de HPOs

A *HPO Factory* é responsável por inicializar as estratégias de HPO para permitir sua comparação. Ela trabalha com uma interface `HPO Strategy`, que fornece um método para estabelecer um contrato com qualquer estratégia de HPO implementada: *hyperparameter_optimization()*. O diagrama Linguagem de Modelagem Unificada, do inglês, *Unified Modeling Language* (UML) da HPO Factory é mostrado na Figura 4.

Na aplicação demonstrativa inicial, implementaram-se três estratégias bem conhecidas de HPO, a saber: BOHB (FALKNER; KLEIN; HUTTER, 2018), Hyperband (LI

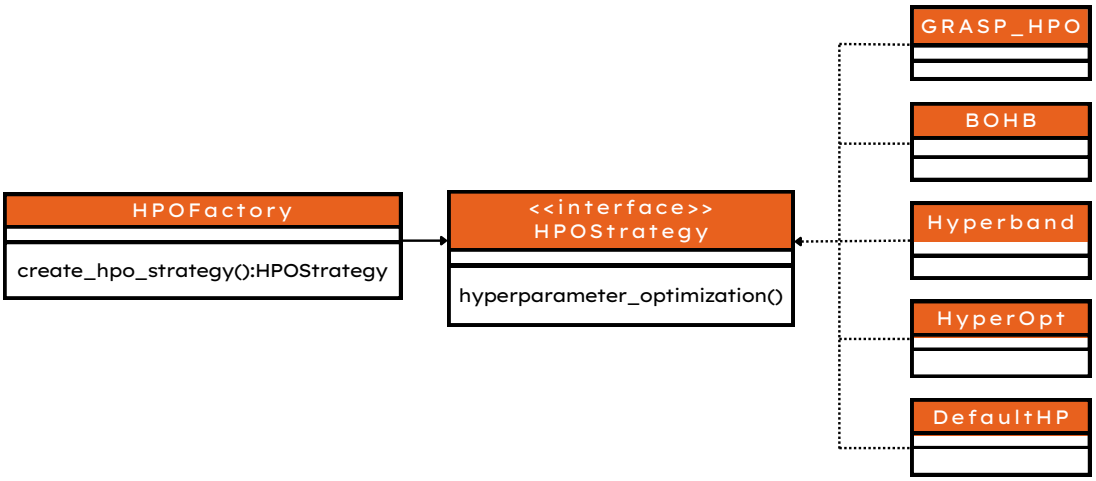


Figura 4 – Diagrama UML da Fábrica de HPO. Fonte: do Autor

et al., 2018) e HyperOpt (BERGSTRA; YAMINS; COX, 2013). Também foi implementado um método básico *Default HPs*, sem HPO. Como discutido nas seções seguintes, este algoritmo foi escolhido para nossa implementação de prova de conceito. Por fim, implementou-se uma interface GRASP-HPO para suportar o método na plataforma de comparação de HPO.

Para inicializar esses diversos métodos de HPO, a classe `HPOFactory` é utilizada. Um objeto HPO é produzido, o qual contém a função para realizar o HPO. A função recebe as divisões do conjunto de dados de treinamento, teste e validação, bem como os HPs definidos pelo usuário e seus respectivos intervalos.

Após o pré-processamento dos dados e a inicialização dos métodos de HPO, a plataforma de comparação proposta executa o HPO respectivo, utilizando o algoritmo XGBoost (CHEN; GUESTRIN, 2016) nas bases de dados mencionadas. Os resultados são então coletados para análise e geração de gráficos, os quais são apresentados na Seção 5.

## 3.2 Método GRASP-HPO

Nesta seção, apresenta-se o GRASP-HPO, um método para HPO voltado para detecção de intrusões. Conforme mencionado anteriormente, o GRASP-HPO possui duas fases principais: construção e busca local. A Figura 5 oferece uma visão geral do método proposto, ilustrando o fluxo de trabalho desde o recebimento dos dados até a seleção dos melhores HPs.

Ao receber os conjuntos de dados de treinamento e validação, o GRASP-HPO inicia uma fase de construção, que gera combinações de HPs repetidamente até que uma condição de parada configurável seja atingida. Durante essa fase, os melhores resultados são armazenados em uma estrutura de dados do tipo fila de prioridade, que organiza os conjuntos de HPs com base em seu desempenho, determinado pela métrica utilizada

(por exemplo, F1-Score). Essa fila tem um tamanho máximo definido e mantém apenas as combinações de HPs mais promissoras, removendo automaticamente as menos eficazes conforme novas combinações são avaliadas.

Após a fase de construção, os conjuntos de HPs presentes na fila de prioridade são utilizados como ponto de partida para a busca local, onde novos valores próximos (vizinhos) são gerados iterativamente até que uma das condições de parada seja atingida. Ao final da busca local, o modelo com o melhor F1-Score é retornado.

Como alternativa, o GRASP-HPO pode ser executado continuamente, reportando de imediato cada novo conjunto de HPs de melhor desempenho encontrado para um IDS, independentemente do alcance dos critérios de parada.

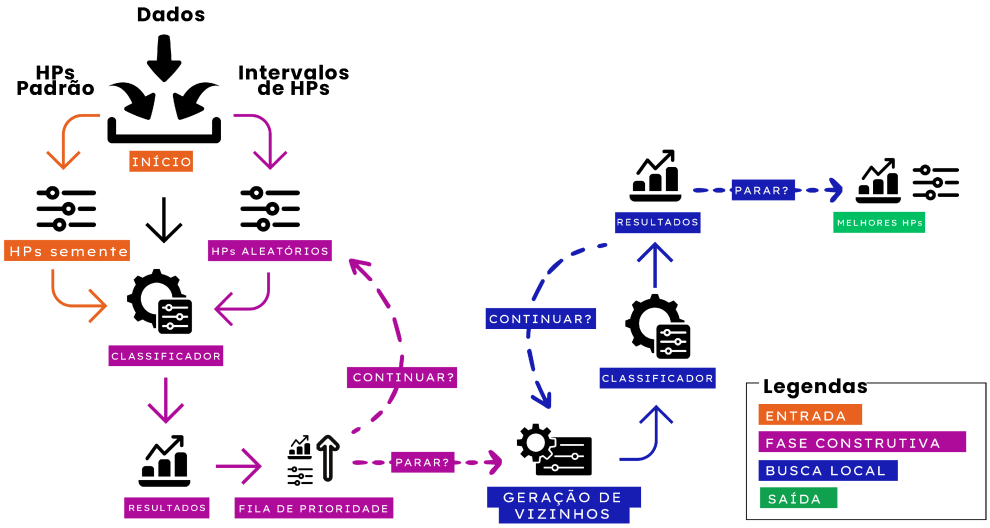


Figura 5 – Diagrama de fluxo do GRASP-HPO. Fonte: Do Autor.

### 3.2.1 Fase de construção

O GRASP é projetado como um algoritmo multi-início, combinando características gananciosas e aleatórias para explorar o espaço de busca de maneira eficiente. No contexto do GRASP-HPO, pode ser empregado um processo de semeação (*seeding*), onde a avaliação inicial utiliza os valores de HPs fornecidos previamente pelo usuário, chamados de HP semente. Após essa etapa inicial, o algoritmo gera combinações aleatórias de HPs dentro de intervalos também definidos pelo usuário, permitindo uma exploração abrangente de configurações alternativas e potencialmente mais eficazes.

O procedimento da fase de construção é detalhado no Algoritmo 1.

**Algorithm 1:** FASE DE CONSTRUÇÃO

```
   input : datasetEreno // conjunto de dados de entradas
           seedHPs // valores iniciais de HPs semente
           HPsRange // faixa de valores de HPs
           priorityQueueMaxSize // tamanho da fila de prioridade
           timeLimitConstruct // limite de tempo da fase de construção
           maxInterationConstrutct // número de iterações da fase de construção
   output: priorityQueue // melhores combinações de valores de HPs
 1 begin
 2     startTime ← 0 // tempo de início
 3     priorityQueue ← ∅ // fila de prioridade de F1-Score das combinações de HPs
 4     addPriorityQueue(eval (seedHPs))
 5     while interationConstruct < maxInterationConstrutct do
 6         if actualTime − startTime > timeLimitConstruct then
 7             break
 8         end
 9         newSolution ← construct(HPsRange)
10         newSolution.F1Score ← eval(newSolution)
11         if priorityQueue.queueSize () < priorityQueueMaxSize then
12             addPriorityQueue(newSolution, newSolution.F1Score)
13         end
14         else if newSolution.F1Score > priorityQueue.lowestF1Score () then
15             priorityQueue.removeLowestF1Score ()
16             addPriorityQueue(newSolution, newSolution.F1Score)
17         end
18         interationConstruct ← interationConstruct + 1
19     end
20     return priorityQueue
21 end
```

A Figura 5 ilustra as entradas do GRASP-HPO: um conjunto de HPs semente, o conjunto de dados e os intervalos de HPs definidos pelo usuário. A execução inicial utiliza essas entradas para uma primeira avaliação, estabelecendo um ponto de partida. Neste sentido, o GRASP-HPO é configurado para gerar resultados com desempenho igual ou superior ao dos HPs semente fornecidos.

A fase de construção (Figura 5, rótulos roxos) começa com os valores de HPs semente (veja a linha 4, Algoritmo 1). Em seguida, uma combinação de valores de HPs é escolhida aleatoriamente e avaliada repetidamente até que os critérios de parada sejam alcançados (veja a linha 5, Algoritmo 1).

A avaliação de cada HP é feita através do treinamento supervisionado do algoritmo *XGBoost Classifier* para o conjunto de dados de entrada; conforme mencionado acima, o *XGBoost* classifica cada amostra da base de dados como um ataque ou tráfego normal. Uma fila de prioridade é usada para facilitar o gerenciamento e armazenamento das melhores soluções avaliadas (veja as linhas 3 e 11-17, Algoritmo 1), com base em seu F1-Score. Uma vez que a fila de prioridade tenha sido preenchida até um certo tamanho, as iterações seguintes substituirão a combinação com o menor F1-Score (veja a linha 14, Algoritmo 1). Dessa forma, as $N$ melhores combinações são mantidas de forma gulosa.

No Algoritmo 1, a fase de construção continua até que: (a) o número máximo de iterações seja alcançado (veja a linha 5) ou (b) o limite de tempo seja atingido (veja a linha 6). Esse limite de tempo permite que a fase de construção termine rapidamente e

**Algorithm 2:** Fase de Busca Local

```
   input : datasetEreno // conjunto de dados de entrada
           HPsRange // intervalo de HPs
           priorityQueue // fila de prioridade com os melhores resultados da fase anterior
           timeLimitLS // tempo limite da busca local
           maxInterationHillClimb // número de iterações do algoritmo de Hill Climbing
   output: bestHPs // Melhor conjunto de HPs
           bestF1Score // F1-Score do melhor conjunto de HPs
 1 begin
 2     startTime ← 0 // Tempo de início
 3     bestHPs ← ∅ // Melhor combinação de HPs
 4     bestF1Score ← priorityQueue.getFirst ()
 5     while priorityQueue.queueSize() ≠ ∅ do
 6         if actualTime − startTime > timeLimitLS then
 7             break
 8         end
 9         newSolution ← priorityQueue.dequeue ()
10         bestLocalF1Score ← hillClimb(newSolution, maxInterationHillClimb)
11         bestLocalF1Score.F1Score ← eval(bestLocalF1Score)
12         if bestLocalF1Score.F1Score > bestF1Score then
13             bestHPs ← bestLocalF1Score
14             bestF1Score ← bestLocalF1Score.F1Score
15         end
16     end
17     return bestHPs
18 end
```

seja mais competitiva em comparação a outros algoritmos de HPO com parada antecipada. No final desta fase, uma fila de prioridade com a melhor combinação de valores é retornada. Alternativamente, com uma pequena modificação, a busca local poderia ser executada em paralelo, tomando como entrada as melhores soluções da fase de construção atual. Uma abordagem semelhante foi proposta para lidar com o problema de seleção de características (SILVA et al., 2023), com resultados que demonstraram uma melhora significativa em termos de precisão e eficiência no tempo de processamento, especialmente em arquiteturas distribuídas.

### 3.2.2 Fase de Busca Local

A busca local (Figura 5, rótulos azuis; veja Algoritmo 2) inicia com a fila de prioridade, da qual os itens são retirados sequencialmente para execução da função de subida de encosta (do inglês, *Hill Climbing*) nas combinações de HPs (linha 10). Esse algoritmo é uma estratégia iterativa de otimização local que começa com uma solução inicial e explora soluções vizinhas em busca de melhorias na função objetivo. Em cada iteração, a solução vizinha com o maior ganho é selecionada, e o processo continua até que nenhum vizinho proporcione melhoria adicional ou que um critério de parada seja atingido.

Na geração de vizinhos, um único HP da combinação atual é selecionado aleatoriamente para sofrer uma alteração. Essa alteração é realizada dentro dos limites definidos no espaço de busca fornecido pelo usuário, garantindo que os valores gerados sejam sempre válidos, desde que os parâmetros de entrada tenham sido configurados corretamente.

A natureza dessa alteração depende do tipo do HP:

- Para HPs contínuos, um novo valor aleatório é gerado dentro dos limites estabelecidos no espaço de busca.
- Para HPs categóricos ou discretos, o valor é alterado aleatoriamente, escolhendo uma nova opção entre as disponíveis.

Essas pequenas alterações são realizadas repetidamente por um número predefinido de iterações ou até que um critério de parada seja atingido. A função de subida de encosta avalia a vizinhança gerada e retorna a melhor combinação de HPs encontrada, substituindo de forma imediata e gulosa a solução atual caso apresente um desempenho superior (linha 12).

A fase de busca local continua até que uma das seguintes condições seja satisfeita: (a) a fila de prioridade esteja vazia (linha 5) ou (b) o limite de tempo configurado seja alcançado (linha 6). Por fim, a combinação de HPs que obteve o melhor desempenho, juntamente com seu respectivo F1-Score, é retornada como resultado.

# 4 Experimentos e Resultados

Esta seção apresenta o desempenho de diferentes técnicas de HPO no conjunto de dados ERENO, avaliando a métrica de F1-Score. O objetivo é verificar se o GRASP-HPO é uma metodologia eficiente e promissora, particularmente no contexto de treinamento de IDSs.

## 4.1 Integração e Testes

Foram implementados os métodos de HPO, BOHB, Hyperband, GRASP-HPO e HyperOpt (descritos na Seção 2.1.5) na plataforma proposta e realizou-se uma avaliação abrangente. Isso envolve a análise estatística das medidas de desempenho no conjunto de dados ERENO para validar as vantagens do método GRASP-HPO para o caso de uso de IDS (que utiliza *XGBoost*). As métricas de desempenho utilizadas foram F1-Score e tempo de execução. A faixa de valores para os HPs foi mantida uniforme em todos os métodos de HPO, conforme detalhado na Tabela 2. Os valores do espaço de busca foram adaptados da documentação do XGBoost (BROWNLEE, 2024) e da pesquisa de Yang e Shami (2020), que também sugere possíveis valores para HPs comuns como boas práticas.

Tabela 2 – Intervalo de valores dos HPs do *XGBoost* considerados.

| **HP** | **Tipo** | **Valor Mínimo** | **Valor Máximo** |
|---|---|---|---|
| max_depth | inteiro | 3 | 10 |
| colsample_bytree | ponto flutuante | 0.5 | 1 |
| subsample | ponto flutuante | 0.5 | 1.0 |
| learning_rate | ponto flutuante | $1 \times 10^{-3}$ | 0.2 |

Além do intervalo de busca dos HPs, foi fornecido como conjunto de HPs de semeação os HPs padrões providos para o XGBoost na biblioteca, como mostrado na Tabela 3.

Tabela 3 – Valores padrão dos HPs do XGBoost na biblioteca *scikit-learn*.

| **HP** | **Tipo** | **Valor Padrão** |
|---|---|---|
| max_depth | inteiro | 6 |
| colsample_bytree | ponto flutuante | 1.0 |
| subsample | ponto flutuante | 1.0 |
| learning_rate | ponto flutuante | 0.3 |

Para garantir a comparabilidade dos resultados, cada base de dados foi dividida usando 60% para treinamento (aprendendo a identificar padrões e características úteis),

20% para validação (para ajuste dos HPs durante o treinamento, sem expor dados de teste) e 20% para teste (para classificação dos dados). Foram realizados 30 experimentos, cada um consistindo na execução de todos os métodos de HPO. Em cada execução de um método de HPO, realizaram-se 10 testes do modelo XGBoost com os HPs fornecidos, e cada teste foi executado por 100 rodadas/épocas (uma característica do classificador XGBoost). Ao final de cada teste, o F1-Score final é retornado ao algoritmo de HPO para a seleção do conjunto de HPs ideal. No total, foram realizados 30 experimentos com 4 métodos de HPO, totalizando 1.200 testes (10 por execução) e 120.000 épocas de XGBoost no total.

## 4.2 Análise Comparativa do F1-Score e do Tempo de Execução

As Figuras (6) e (7) ilustram que GRASP-HPO demonstra resultados competitivos aplicado no contexto de IDS com o conjunto de dados ERENO. Na Figura 6, o gráfico da esquerda mostra 30 experimentos do GRASP-HPO com cinco combinações enquanto o da direita exibe 30 experimentos com apenas 1 combinação na fase de construção (ver Seção 3.2). Mesmo reduzindo o número de combinações na fase de construção, o GRASP-HPO é capaz de alcançar o mesmo F1-Score. O GRASP-HPO destaca-se por sua simplicidade e facilidade de implementação. Trata-se de um algoritmo intuitivo que não exige conhecimento aprofundado sobre o modelo em otimização e, ainda assim, entrega resultados significativos.

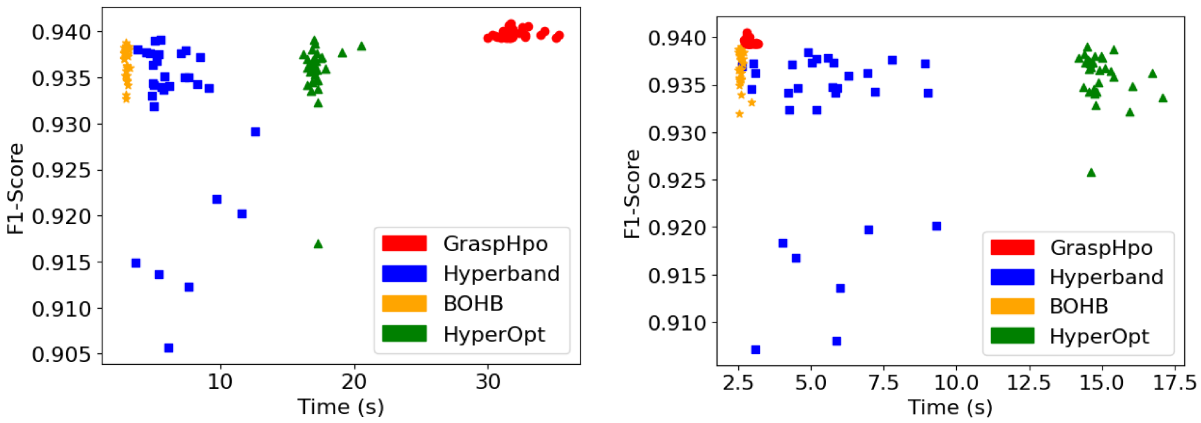


Figura 6 – Gráfico de dispersão do F1-Score versus Tempo de Execução para 30 experimentos. Fonte: do Autor.

O GRASP-HPO oferece F1-scores muito competitivos quando comparado com os três outros algoritmos de HPO ao longo de 30 experimentos independentes. Quando executado sob as mesmas condições de teste (10 testes por HPO), o GRASP-HPO é significativamente mais lento em tempo do que todos os outros métodos de HPO. No entanto, quando o número de combinações é reduzido em favor da velocidade, o GRASP-HPO mantém os altos valores de F1-Score. Espera-se que o Hyperband e o BOHB sejam os

métodos de HPO mais rápidos, devido aos mecanismos de parada antecipada que interrompem testes ruins antes que seu número total de rodadas tenha sido executado. Também se espera que o HyperOpt seja consistente, pois executa os testes até o final. Uma possível razão para o desempenho inferior do GRASP-HPO com 5 combinações pode ser o impacto negativo da configuração padrão da fila de prioridade do Python ou os custos associados à subida de encosta. No entanto, conforme mostrado pela Figura 6, o GRASP-HPO não precisa do mesmo número de testes para alcançar altos valores de F1-Score. Neste experimento em particular, o foco foi na métrica F1-Score.

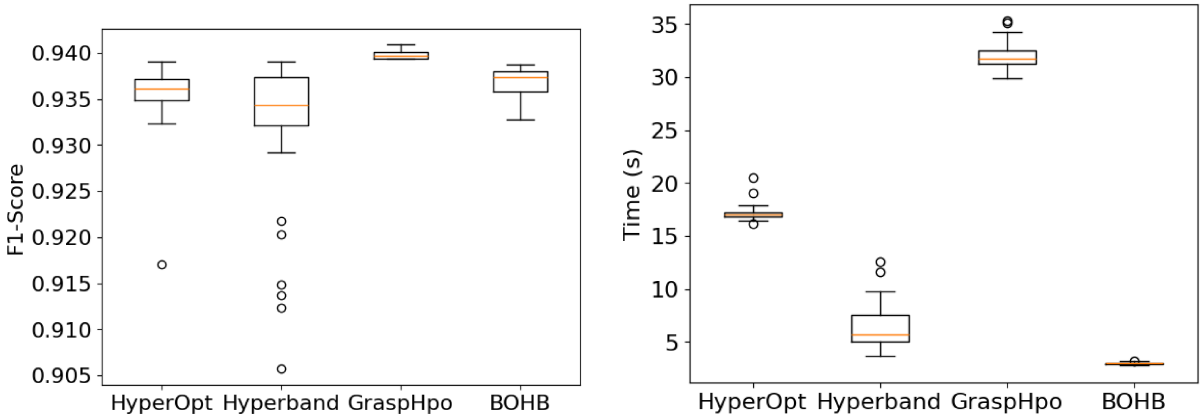


Figura 7 – *Boxplot* do F1-Score (à esquerda) e do Tempo (à direita) para 30 experimentos. Fonte: do Autor.

A Figura 7 oferece uma visão resumida da Figura 6 (gráfico à esquerda). Como pode ser observado, o GRASP-HPO possui uma média de F1-Score muito alta, com baixa variabilidade. Enquanto algoritmos como o Hyperband são rápidos, o GRASP-HPO se destaca em obter um resultado inicial de otimização mais robusto.

Embora o Hyperband e o BOHB tenham tempos mais rápidos, eles tendem a apresentar uma média de F1-Score menor, especialmente o Hyperband, que realiza uma busca aleatória. No entanto, o GRASP-HPO mantém seu alto F1-Score ao longo dos 30 experimentos independentes.

## 4.3 Evolução da Otimização ao Longo do Tempo

Análises adicionais foram conduzidas sobre os detalhes de execução para explorar maneiras de reduzir o tempo de execução do GRASP-HPO. A Figura 8 mostra que o GRASP-HPO atinge resultados quase ótimos com mais rapidez que outros métodos, mas continua sua busca por otimização sem um limite de tempo definido.

Note que a fase de construção do GRASP-HPO é rápida (primeiros 6 pontos), enquanto a fase de busca local é lenta (últimos 5 pontos). Devido à natureza do GRASP-HPO, ele realiza uma avaliação dos HPs padrão inicialmente, resultando em um ponto de dados

extra no início. Se um mecanismo de parada antecipada fosse implementado, seria possível descartar os testes subsequentes na fase de busca local, acelerando ainda mais o processo.

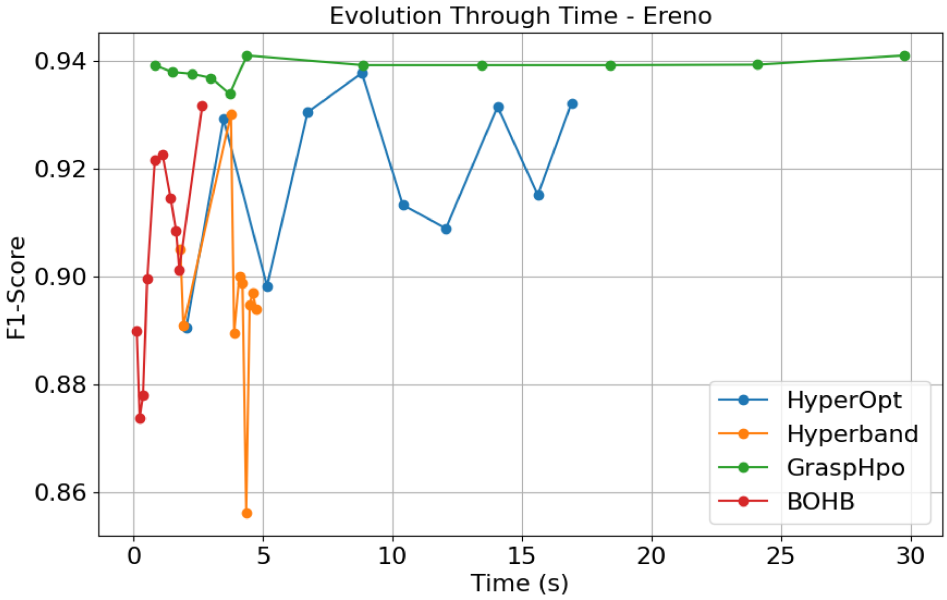


Figura 8 – Evolução do F1-Score ao longo do tempo com 10 testes. Fonte: do Autor.

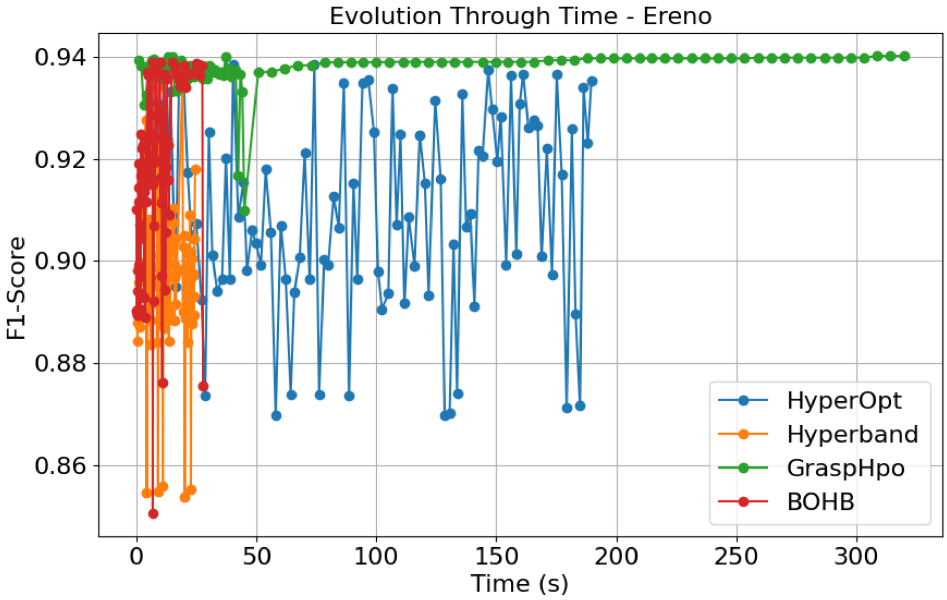


Figura 9 – Evolução do F1-Score ao longo do tempo com 100 testes. Fonte: do Autor.

A falta de um critério de parada relacionado ao tempo indica que a menor eficiência temporal do GRASP-HPO, em comparação com outros algoritmos de HPO, se deve ao uso de uma estratégia de término mais rápida por parte desses algoritmos. Essa estratégia limita a busca por melhores resultados em favor de uma execução mais rápida, o que poderia ser facilmente adaptado alterando os critérios para o GRASP-HPO.

O alto F1-Score inicial observado nas Figuras 8 e 9 sugere que o GRASP-HPO pode ser ajustado para parar muito mais cedo, com menos testes ou um limite de tempo menor, mantendo F1-scores competitivos ou até melhores. Na verdade, o GRASP-HPO pode introduzir mecanismos automáticos de parada antecipada que interromperiam os testes antes de completarem, evitando que o XGBoost execute o número total de épocas planejadas se o F1-Score não melhorar.

Além disso, embora as Figuras 8 e 9 apresentem um experimento individual, os resultados sugerem que a média (ao longo de 30 experimentos) do F1-Score na fase de construção é apenas 0,001 menor do que o F1-Score máximo alcançado pelo GRASP-HPO. O desvio padrão da diferença entre o F1-Score máximo e o F1-Score na fase de construção também é de 0,001, demonstrando resultados estatisticamente significativos que reforçam a possibilidade de implementar a parada antecipada mencionada anteriormente.

# 5 Conclusão

Este capítulo discute as principais contribuições do trabalho e descreve trabalhos futuros potenciais que prometem avanços adicionais no campo da otimização de HPs.

## 5.1 Principais Contribuições

Este trabalho apresentou o GRASP-HPO, uma técnica de HPO destinada a melhorar o desempenho de IDSs. Esta técnica foi comparada a outras abordagens de HPO do estado da arte, utilizando uma plataforma de avaliação desenvolvida para comparar otimizadores de HPs em cenários consistentes e sob as mesmas condições de entrada.

Nos experimentos realizados, o GRASP-HPO demonstrou um desempenho promissor, alcançando até 94% de F1-Score no contexto de IDS. Em especial, os testes mostraram que o GRASP-HPO é eficaz na otimização do algoritmo de AM XGBoost treinado com o conjunto de dados ERENO, voltado para segurança cibernética. Embora o GRASP-HPO tenha superado outras técnicas em F1-Score, a redução do tempo de convergência depende da implementação de um mecanismo de parada antecipada, que interromperia a busca por combinações adicionais de HPs sem ganho significativo. Esses resultados evidenciam sua versatilidade e robustez, bem como sua capacidade de adaptação a cenários de alta dimensionalidade.

Os destaques experimentais que o trabalho proporcionou foram:

- O GRASP-HPO demonstrou melhor desempenho na otimização do F1-Score do XGBoost treinado com o conjunto de dados ERENO, quando comparado a outras técnicas da literatura, como BOHB, *Hyperband* e *HyperOpt*
- A plataforma de comparação desenvolvida permitiu uma análise sistemática e detalhada do desempenho de cada técnica de HPO, facilitando a visualização dos resultados e a incorporação de novos algoritmos de HPO e conjuntos de dados para trabalhos futuros.

A implementação e compreensão da técnica GRASP-HPO são diretas e altamente modulares, o que oferece várias oportunidades para trabalhos futuros visando melhorar seu desempenho e robustez, conforme discutido na seção a seguir.

## 5.2 Trabalhos Futuros

Diversas possibilidades para pesquisas futuras foram identificadas para melhorar o desempenho e a alocação de recursos do GRASP-HPO. A arquitetura do algoritmo permite a execução paralela de suas fases, seja em uma versão *multithread* ou em um sistema distribuído, o que reduziria significativamente o tempo de execução.

Uma direção promissora seria o desenvolvimento de uma versão multiobjetivo do GRASP-HPO, capaz de otimizar não apenas métricas de desempenho, como a F1-Score, mas também outros critérios importantes, como o tempo de treinamento do modelo ou o uso de recursos computacionais. Essa abordagem seria especialmente útil em cenários onde o custo computacional e a eficiência do modelo são fatores críticos.

Expandir a adaptabilidade do GRASP-HPO para otimizar HPs em modelos além do XGBoost é outra direção importante, além de aplicá-lo a contextos e conjuntos de dados diferentes. Isso aumentaria sua robustez e aplicabilidade a uma gama mais ampla de algoritmos e contextos de AM.

Desenvolver novas estratégias para as fases de construção e busca local do GRASP-HPO e comparar o desempenho das diferentes combinações geradas poderia potencialmente melhorar os resultados da meta-heurística.

Trabalhos futuros também poderiam explorar diferentes métodos de HPO de ponta não cobertos neste trabalho, como algoritmos de vetor de gradiente e outros algoritmos meta-heurísticos.

Uma extensão da plataforma de comparação de HPO para torná-la mais genérica e potencialmente disponível publicamente em um servidor permitiria aos usuários comparar os resultados de diferentes estratégias de otimização de HP aplicadas aos seus conjuntos de dados.

Esses aprimoramentos e expansões promoveriam o desenvolvimento de técnicas de HPO e contribuiriam para o campo mais amplo da otimização em AM, fornecendo ferramentas mais eficientes, versáteis e acessíveis à comunidade de pesquisa.

# Referências

A, G. et al. Intrusion detection in computer networks using machine learning algorithms. In: *2021 10th IEEE International Conference on Communication Systems and Network Technologies (CSNT)*. [S.l.: s.n.], 2021. p. 605–611. Citado na página 17.

AHMIM, A.; MAGLARAS, L.; FERRAG, M. A.; DERDOUR, M.; JANICKE, H. A novel hierarchical intrusion detection system based on decision tree and rules-based models. In: **2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)**. [S.l.: s.n.], 2019. p. 228–233. Citado na página 24.

AWATRAMANI, V.; GUPTA, P. Intrusion detection using nature-inspired algorithms and automated machine learning. **Smart and Sustainable Intelligent Systems**, Wiley Online Library, p. 295–306, 2021. Citado na página 28.

BAIGENT, D.; ADAMIAK, M.; MACKIEWICZ, R.; SISCO, G. Iec 61850 communication networks and systems in substations: An overview for users. **SISCO Systems**, Citeseer, 2004. Citado na página 22.

BECCENERI, J. C. Meta-heurísticas e otimização combinatória: Aplicações em problemas ambientais. **INPE, Sao José dos Campos**, 2008. Citado na página 23.

BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. **Journal of machine learning research**, v. 13, n. 2, 2012. Citado na página 11.

BERGSTRA, J.; YAMINS, D.; COX, D. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In: PMLR. **International conference on machine learning**. [S.l.], 2013. p. 115–123. Citado na página 31.

BIJU, J. M.; GOPAL, N.; PRAKASH, A. J. Cyber attacks and its different types. **International Research Journal of Engineering and Technology**, v. 6, n. 3, p. 4849–4852, 2019. Citado na página 14.

BISHOP, C. M.; NASRABADI, N. M. **Pattern recognition and machine learning**. [S.l.]: Springer, 2006. v. 4. Citado na página 19.

BROWNLEE, J. **Suggested Ranges for Tuning XGBoost Hyperparameters | XGBoosting**. 2024. <https://xgboosting.com/suggested-ranges-for-tuning-xgboost-hyperparameters/>. (Accessed on 04/24/2024). Citado na página 36.

BUCZAK, A. L.; GUVEN, E. A survey of data mining and machine learning methods for cyber security intrusion detection. **IEEE Communications Surveys E Tutorials**, v. 18, n. 2, p. 1153–1176, 2016. Citado na página 16.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. **CoRR**, abs/1603.02754, 2016. Disponível em: <http://arxiv.org/abs/1603.02754>. Citado 2 vezes nas páginas 24 e 31.

CHENG, D.; ZHANG, H.; XIA, F.; LI, S.; ZHANG, Y. Using gradient based multikernel gaussian process and meta-acquisition function to accelerate smbo. In: IEEE. **2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)**. [S.l.], 2019. p. 440–447. Citado na página 27.

DAS, N.; SARKAR, T. Survey on host and network based intrusion detection system. **International Journal of Advanced Networking and Applications**, Eswar Publications, v. 6, n. 2, p. 2266, 2014. Citado na página 15.

FALKNER, S.; KLEIN, A.; HUTTER, F. **BOHB: Robust and Efficient Hyperparameter Optimization at Scale**. 2018. Citado 2 vezes nas páginas 24 e 30.

FEO, T. A.; RESENDE, M. G. A probabilistic heuristic for a computationally difficult set covering problem. **Operations research letters**, Elsevier, v. 8, n. 2, p. 67–71, 1989. Citado na página 25.

FRIEDMAN, J. H. Greedy function approximation: A gradient boosting machine. **The Annals of Statistics**, Institute of Mathematical Statistics, v. 29, n. 5, p. 1189 – 1232, 2001. Disponível em: <https://doi.org/10.1214/aos/1013203451>. Citado na página 24.

GALLATIN, K.; ALBON, C. **Machine learning with Python cookbook**. [S.l.]: "O'Reilly Media, Inc.", 2023. Citado na página 19.

GAMMA, E. Design patterns: elements of reusable object-oriented software. **Person Education Inc**, 1995. Citado na página 29.

GENDREAU, M.; POTVIN, J.-Y. Tabu search. In: _____. **Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques**. Boston, MA: Springer US, 2005. p. 165–186. ISBN 978-0-387-28356-2. Disponível em: <https://doi.org/10.1007/0-387-28356-0_6>. Citado na página 12.

HAMID, Y.; SUGUMARAN, M.; BALASARASWATHI, V. Ids using machine learning-current state of art and future directions. **British Journal of Applied Science & Technology**, v. 15, n. 3, p. 1–22, 2016. Citado na página 11.

HARA, K.; SHIOMOTO, K. Intrusion detection system using semi-supervised learning with adversarial auto-encoder. In: **NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium**. [S.l.: s.n.], 2020. p. 1–8. Citado na página 17.

HOWCROFT, E. **Blockchain vinculada à Binance sofre ataque digital de US$ 570 milhões**. 2022. <https://www.cnnbrasil.com.br/business/blockchain-vinculada-a-binance-sofre-ataque-digital-de-us-570-milhoes/>. (Accessed on 15/01/2023). Citado na página 11.

HUTTER, F.; HOOS, H. H.; LEYTON-BROWN, K. Sequential model-based optimization for general algorithm configuration (extended version). **Technical Report TR-2010–10, University of British Columbia, Computer Science, Tech. Rep.**, 2010. Citado na página 23.

HUTTER, F.; KOTTHOFF, L.; VANSCHOREN, J. (Ed.). **Automatic machine learning: methods, systems, challenges**. Germany: Springer, 2019. (Challenges in Machine Learning). ISBN 978-3-030-05317-8. Citado 2 vezes nas páginas 22 e 23.

KHRAISAT, A.; GONDAL, I.; VAMPLEW, P.; KAMRUZZAMAN, J. Survey of intrusion detection systems: techniques, datasets and challenges. **Cybersecurity**, Springer, v. 2, n. 1, p. 1–22, 2019. Citado na página 16.

KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. **Science**, v. 220, n. 4598, p. 671–680, 1983. Disponível em: <https://www.science.org/doi/abs/10.1126/science.220.4598.671>. Citado na página 12.

KOMER, B.; BERGSTRA, J.; ELIASMITH, C. Hyperopt-sklearn. **Automated Machine Learning: Methods, Systems, Challenges**, Springer International Publishing, p. 97–111, 2019. Citado na página 23.

KUMARI, R.; SHEETANSHU; SINGH, M. K.; JHA, R.; SINGH, N. Anomaly detection in network traffic using k-mean clustering. In: **2016 3rd International Conference on Recent Advances in Information Technology (RAIT)**. [S.l.: s.n.], 2016. p. 387–393. Citado na página 17.

KUNANG, Y. N.; NURMAINI, S.; STIAWAN, D.; SUPRAPTO, B. Y. Attack classification of an intrusion detection system using deep learning and hyperparameter optimization. **Journal of Information Security and Applications**, Elsevier, v. 58, p. 102804, 2021. Citado 2 vezes nas páginas 27 e 28.

LI, L.; JAMIESON, K.; DESALVO, G.; ROSTAMIZADEH, A.; TALWALKAR, A. **Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization**. 2018. Citado 2 vezes nas páginas 23 e 31.

LIPPMANN, R.; HAINES, J. W.; FRIED, D. J.; KORBA, J.; DAS, K. The 1999 darpa off-line intrusion detection evaluation. **Computer Networks**, v. 34, n. 4, p. 579–595, 2000. ISSN 1389-1286. Recent Advances in Intrusion Detection Systems. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1389128600001390>. Citado na página 16.

PEDREGOSA, F. Hyperparameter optimization with approximate gradient. In: BALCAN, M. F.; WEINBERGER, K. Q. (Ed.). **Proceedings of The 33rd International Conference on Machine Learning**. New York, New York, USA: PMLR, 2016. (Proceedings of Machine Learning Research, v. 48), p. 737–746. Disponível em: <https://proceedings.mlr.press/v48/pedregosa16.html>. Citado 2 vezes nas páginas 23 e 27.

QUINCOZES, S. E.; ALBUQUERQUE, C.; PASSOS, D.; MOSSÉ, D. A survey on intrusion detection and prevention systems in digital substations. **Computer Networks**, Elsevier, v. 184, p. 107679, 2021. Citado 2 vezes nas páginas 5 e 20.

_____. ERENO: A Framework for Generating Realistic IEC–61850 Intrusion Detection Datasets for Smart Grids. **IEEE Transactions on Dependable and Secure Computing**, IEEE, 2023. Citado na página 19.

QUINCOZES, S. E.; PASSOS, D.; ALBUQUERQUE, C.; OCHI, L. S.; MOSSÉ, D. Grasp-based feature selection for intrusion detection in cps perception layer. In: IEEE. **2020 4th Conference on cloud and internet of things (CIoT)**. [S.l.], 2020. p. 41–48. Citado 2 vezes nas páginas 12 e 26.

RESENDE, M. G.; RIBEIRO, C. C. **Optimization by GRASP**. [S.l.]: Springer, 2016. Citado na página 12.

SCARFONE, K. Guide to intrusion detection and prevention systems (idps). **NIST special publication**, 2007. Citado na página 15.

SEN, R.; KANDASAMY, K.; SHAKKOTTAI, S. Multi-fidelity black-box optimization with hierarchical partitions. In: DY, J.; KRAUSE, A. (Ed.). **Proceedings of the 35th International Conference on Machine Learning**. PMLR, 2018. (Proceedings of Machine Learning Research, v. 80), p. 4538–4547. Disponível em: <https://proceedings.mlr.press/v80/sen18a.html>. Citado na página 28.

SILVA, E. F.; NAVES, N.; QUINCOZES, S. E.; QUINCOZES, V. E.; KAZIENKO, J. F.; CHEIKHROUHOU, O. GDLS-FS: scaling feature selection for intrusion detection with GRASP-FS and distributed local search. In: SPRINGER. **International conference on advanced information networking and applications**. [S.l.], 2023. p. 199–210. Citado na página 34.

ÜRÜN, M. B.; SÖNMEZ, Y. Nelder-mead optimized weighted voting ensemble learning for network intrusion detection. **Düzce Üniversitesi Bilim ve Teknoloji Dergisi**, Duzce University, v. 12, n. 4, p. 2139–2158, 2024. Citado na página 16.

YANG, L.; SHAMI, A. On hyperparameter optimization of machine learning algorithms: Theory and practice. **Neurocomputing**, v. 415, p. 295–316, 2020. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231220311693>. Citado 5 vezes nas páginas 11, 13, 22, 26 e 36.

YING, X. An overview of overfitting and its solutions. **Journal of Physics: Conference Series**, IOP Publishing, v. 1168, n. 2, p. 022022, feb 2019. Disponível em: <https://dx.doi.org/10.1088/1742-6596/1168/2/022022>. Citado na página 18.

ZHANG, H.; JIANG, Y. The improved k-means algorithm in intrusion detection system research. **Advanced Engineering Forum**, v. 1, p. 204–208, 09 2011. Citado na página 16.

# Parte I

# Anexos

# GRASP-HPO: Hyperparameter Optimization using GRASP-Based Techniques[1]

Daniel Mossé, Silvio Quincozes

Shinwoo Kim, Enoch Li, Jack Bellamy, Zi Han Ding, Zane Kissel, Gabriel Otsuka

# Introduction

Our project expands on previous work by Dr. Silvio Quincozes, et al.[2] The main goal is to apply the principles of the GRASP (Greedy Randomized Adaptive Search Procedure) meta-heuristic—which is typically used to solve combinatorial optimization problems—to hyperparameter analysis. We aim to do this in the context of detecting intrusions in substations operating under the GOOSE (Generic Object-Oriented Substation Event) protocol (as defined in IEC 61850). Hence, our implementation must calibrate hyperparameters to maximize the detection of attacks while minimizing errors. Minimizing the number of false positives and false negatives is of crucial importance.

The development of our project will consist of two smaller teams working in parallel. The first team will focus on developing the workflow for GRASP-HPO, and the second team will focus on implementing existing algorithms—mainly, Hyperopt, HyperBand, and BOHB (Bayesian Optimization and Hyperband)—which can be used against the comparison of the performance of GRASP-HPO. The algorithm implementations will be tested using the ERENO IEC-61850 Intrusion Dataset[3] which has been developed by Dr. Silvio Quincozes.[4] Towards the end of the project, the two teams will come together to present a final report that compares the performance of each algorithm using various metrics.

This report, as well as any other deliverables produced, will be used to form the basis for novel research. The code that is produced will be configurable to test on various datasets, and the final ‘report’ may be used as evidence in the literature which will be authored by Dr. Daniel Mossé and Dr. Silvio Quincozes. Figures and data that are generated in our project may be used directly in such literature.

---

[1] Project was originally titled “Hyper Parameter Optimization” and “Explainable Artificial Intelligence (XAI)” (Applied to Machine-Learning Based Intrusion Detection Systems)

[2] S. E. Quincozes, D. Mossé, D. Passos, C. Albuquerque, L. S. Ochi and V. F. dos Santos, "*On the Performance of GRASP-Based Feature Selection for CPS Intrusion Detection*," in IEEE Transactions on Network and Service Management, vol. 19, no. 1, pp. 614-626, March 2022, doi: 10.1109/TNSM.2021.3088763.

[3] https://www.kaggle.com/datasets/sequincozes/ereno-iec61850-ids

[4] https://github.com/sequincozes/ERENO

# Project Design

Our project will primarily be developed using the Python programming language due to its vast array of available libraries (especially in the areas of machine learning). Python is also a useful choice because it will enable us to merge the code, data, and analysis into a single document using Jupyter Notebooks. Consequently, we expect libraries such as NumPy and Matplotlib to be useful in producing figures and graphics that help explain the output of our algorithm.

The project deliverables will be available via GitHub on a private repository[5] that is shared by team members and project sponsors. We will utilize the features of the Git Version Control System/GitHub to manage various branches, track issues, and approve pull-requests. Other collaboration tools such as Google Docs may be used to draft documents, but the final result will always be copied onto the repository by the document owner.

Note that some unresolved implementation issues exist. Mainly, we must finalize the selection of a criteria/metric that can be used to measure the performance of each model. It is possible that multiple metrics are needed to measure the performance depending on use case. Similarly, implementation details for GRASP such as feature selection, feature engineering, and data cleaning need to be finalized.

Many of our team members have a background in data science and/or machine learning. Consequently, our collective strengths lie in the expertise in knowledge related to hyperparameter optimization. However, because our project applies itself to Intrusion Detection Systems (IDSs), it is possible that team members lack specific domain knowledge related to IEC-61850. Resources have been supplied which aims to provide such knowledge in an abridged format[6], but communication with the sponsor may be needed to better comprehend the underlying dataset.

# Expected Timeline

There are three main deliverables for this project, as described below. The first two deliverables will be developed in parallel as they do not rely on one another; the third final deliverable will need to be developed after the first two deliverables are fully implemented.

1. Deliverable: GRASP-HPO Workflow Implementation

---

[5] https://github.com/shinwookim/GRASP-HPO

[6] Quincozes, Silvio & Albuquerque, Célio & Passos, Diego & Mossé, Daniel. (2022). *ERENO: AN EXTENSIBLE TOOL FOR GENERATING REALISTIC IEC-61850 INTRUSION DETECTION DATASETS.* 10.13140/RG.2.2.15457.79206.

    - Description: Implement the GRASP-HPO workflow, integrating the GRASP meta-heuristic into the hyperparameter optimization process for intrusion detection under the GOOSE protocol (IEC 61850). Ensure the workflow is configurable for the ERENO dataset.
    - Due Nov 1, 2023
2. Deliverable: Implementation of Algorithms to Be Used for Performance Comparison
    - Description: Implement existing hyperparameter optimization algorithms (Hyperopt, Hyperband, and BOHB) to serve as benchmarks for performance comparison using the ERENO dataset.
    - Due Nov 1, 2023
3. Deliverable: Final Report Comparing the Various Algorithm Performance
    - Description: Compile a comprehensive report comparing the performance of GRASP-HPO against other implemented hyperparameter optimization algorithms using various metrics.
    - Due Dec 8, 2023

# Communication

Team members will meet with sponsors in-person at Sennott Square (Chair's Conference Room) every Thursday at 7:00 PM. Dr. Silvio Quincozes will join the meeting via Zoom. At each meeting, members will present their assigned deliverables (or progress) to sponsors.

Discord will be the primary means of communication for team members. Members are expected to check Discord on a regular basis (no later than 48 hours), and check-in with the sprint's Scrum Master every few days. Naturally, team members will also meet once weekly during the designated time for the Capstone course (Friday 12:00 - 2:00 PM in IS building). If needed, team members will also meet in–person in Sennott Square or the Hillman Library (or via Zoom). Dr. Silvio Quincozes will be available to answer any questions on Discord, and Dr. Daniel Mossé will be available via E-mail.

# Initial User Stories

| |
|---|
| As a user<br>I want to use the GRASP-HPO algorithm<br>So that I can produce hyperparameter values for a machine learning model that will increase its performance |
| As a user<br>I want to use the Hyperopt algorithm |

| |
|---|
| So that I can produce hyperparameter values for a machine learning model that will increase its performance |
| As a user<br>I want to use the HyperBand algorithm<br>So that I can produce hyperparameter values for a machine learning model that will increase its performance |
| As a user<br>I want to use the BOHB (Bayesian Optimization HyperBand) algorithm<br>So that I can produce hyperparameter values for a machine learning model that will increase its performance |
| As a user<br>I want to see the performances of GRASP HPO algorithm<br>So that I can analyze and present this data in research |
| As a user<br>I want to see the performances of Hyperopt HPO algorithm<br>So that I can analyze and present this data in research |
| As a user<br>I want to see the performances of HyperBand HPO algorithm<br>So that I can analyze and present this data in research |
| As a user<br>I want to see the performances of BOHB HPO algorithm<br>So that I can analyze and present this data in research |
| As a user<br>I want to see visual representation of the performances of each HPO algorithm compared to one another<br>So that I can visually analyze and present the visual data in research |
| As a user<br>I want an interactive web dashboard which allows the algorithm to be easily tweaked<br>So that easily extend the algorithm without modifying code. |