UTILIZAÇÃO DA INTELIGÊNCIA ARTIFICIAL PARA AUTOMAÇÃO E ANÁLISE DE VULNERABILIDADES EM SISTEMAS FINANCEIROS LEGADOS

Grácio Lumbala Kabongo *, Harmónico Lucas Augusto ♦

Resumo

A crescente digitalização do sector financeiro, embora catalisadora de inovação e eficiência, expõe uma vulnerabilidade estrutural crítica: a dependência de sistemas legados. Estes sistemas, alicerçados em tecnologias obsoletas e caracterizados por uma dívida técnica acumulada ao longo de décadas, representam um risco operacional e cibernético significativo, com potenciais implicações para a estabilidade financeira sistémica. O presente artigo propõe um *framework* inovador que emprega Inteligência Artificial (IA) para automatizar a detecção e análise de vulnerabilidades em tais ambientes. Integrando *Graph Neural Networks* (GNN) para a análise estrutural do código e *Large Language Models* (LLM) para a compreensão semântica, o *framework* visa superar as limitações das abordagens tradicionais de segurança. A metodologia, que combina experimentos controlados com um estudo de caso num ambiente bancário simulado, demonstra ganhos estatisticamente significativos na precisão e na redução do Tempo Médio de Detecção (MTTD) de vulnerabilidades. Para além da validação técnica, o trabalho aprofunda as implicações macroprudenciais da adopção da IA, alinhando-se com *frameworks* regulatórios internacionais, como o *NIST AI Risk Management Framework*, as directrizes do BIS e do FSB, e os princípios de gestão de risco operacional de Basileia III. O artigo conclui com um modelo de governação algorítmica e um *roadmap* de implementação, sublinhando a necessidade de uma supervisão prudencial adaptada à era da IA (SupTech), com vista a salvaguardar a resiliência operacional e a estabilidade financeira, tanto em economias emergentes como desenvolvidas.

Palavras-Chave: inteligência artificial; sistemas financeiros legados; detecção de vulnerabilidades; risco operacional.

* Engenheiro de Ciência da Computação, Instituto Superior de Transportes e Comunicações (ISUTC), email: graciolumbala01@gmail.com, contacto: +258 84 864 5922;

♦ Engenheiro de Ciência da Computação, Instituto Superior de Transportes e Comunicações (ISUTC), email: harmonicolucas@gmail.com; contacto: +258 86 502 0020;

Abstract

The increasing digitalization of the financial sector, while fostering innovation and efficiency, exposes a critical structural vulnerability: dependence on *legacy Systems*. These *System*s, built on obsolete technologies and characterized by technical debt accumulated over decades, represent a significant operational and *cybersecurity* risk, with potential implications for *System*ic financial stability. This article proposes an innovative *framework* that employs Artificial Intelligence (AI) to automate the detection and analysis of vulnerabilities in such environments. By integrating *Graph Neural Networks* (GNN) for structural code analysis and *Large Language Models* (LLM) for semantic understanding, the *framework* seeks to overcome the limitations of traditional security approaches. The methodology, combining controlled experiments with a case study in a simulated banking environment, demonstrates statistically significant improvements in accuracy and reductions in the *Mean Time to Detect* (MTTD) vulnerabilities. Beyond technical validation, the study further explores the macroprudential implications of AI adoption, aligning with international regulatory *framework*s such as the NIST AI Risk Management *Framework,* BIS and FSB guidelines, and Basel III operational risk management principles. The article concludes with an algorithmic governance model and an implementation *roadmap*, emphasizing the need for prudential supervision adapted to the AI era (SupTech) in order to safeguard operational resilience and financial stability in both emerging and developed economies.

Keywords: Artificial Intelligence; Legacy Financial *System*s; Vulnerability Detection; Operational Risk.

1. INTRODUÇÃO

1.1 Problema de Pesquisa

O panorama tecnológico do sector financeiro é marcado por uma tensão inerente entre a necessidade de inovação contínua e a dependência de infra-estruturas tecnológicas enraizadas no passado. Os sistemas financeiros legados, que formam a espinha dorsal de muitas operações bancárias críticas, são frequentemente caracterizados por arquitecturas monolíticas, linguagens de programação arcaicas (e.g., COBOL, PL/SQL, versões antigas de Java) e uma documentação escassa ou desactualizada. Para efeitos deste trabalho, define-se sistema legado como uma plataforma computacional desenvolvida há mais de uma década, assente em linguagens de programação de gerações anteriores, arquitecturas monolíticas, *middleware* proprietário (e.g., IBM CICS, Tuxedo) e bases de dados hierárquicas (e.g., IMS DB) ou relacionais sem suporte activo do fabricante (e.g., Oracle 9i, DB2 v7). Estas plataformas operam tipicamente em mainframes IBM zSeries, servidores UNIX proprietários ou ambientes Windows Server descontinuados, e caracterizam-se pela ausência de APIs modernas, acoplamento excessivo entre módulos e dependência de protocolos de comunicação obsoletos. Esta realidade culmina numa dívida técnica substancial, que não só onera os orçamentos de manutenção, mas, mais criticamente, amplia a superfície de ataque cibernético e o risco operacional [1] [2].

A complexidade intrínseca e a opacidade destes sistemas tornam a detecção manual e automatizada de vulnerabilidades um desafio hercúleo. As ferramentas tradicionais de Análise Estática de Segurança de Aplicações (SAST), baseadas em assinaturas e regras predefinidas, revelam-se frequentemente ineficazes na identificação de falhas lógicas complexas ou de vulnerabilidades de fluxo de dados que atravessam múltiplos módulos interligados em ambientes legados. Esta limitação cria uma lacuna crítica na capacidade das instituições financeiras de gerir proactivamente o seu risco cibernético, expondo-as a ameaças susceptíveis de comprometer a integridade, confidencialidade e disponibilidade dos seus serviços [3].

---

[1] Kruchten, P., Nord, R. L., & Ozkaya, I. (2012). Technical Debt: Managing the Cost of Software Maintenance. *IEEE Software*, 29(6), 18-21.

[2] Seacord, R., Plakosh, D., & Lewis, G. (2003). *Modernizing Legacy Systems: Software Technologies, Engineering Processes, and Business Practices*. Addison-Wesley.

[3] OWASP. (2021/2023). *OWASP Top 10 - 2021*. Open Web Application Security Project. Disponível em: https://owasp.org/www-project-top-ten/

No contexto específico de Moçambique, o sistema financeiro tem registado uma digitalização acelerada, impulsionada pela expansão dos serviços de *mobile banking* e pela modernização dos sistemas de pagamentos sob a supervisão do Banco de Moçambique (BdM). Contudo, esta modernização coexiste com uma infra-estrutura bancária que, em muitas instituições, assenta em sistemas *core banking* desenvolvidos nas décadas de 1990 e 2000, frequentemente em Java legado e PL/SQL, com integrações via *middleware* proprietário. O BdM, enquanto autoridade de supervisão prudencial, tem vindo a reforçar o quadro regulatório de cibersegurança, incluindo a implementação de um sistema de reporte de incidentes cibernéticos e a promoção de boas práticas de gestão de risco tecnológico. Neste contexto, a automação da detecção de vulnerabilidades em sistemas legados assume uma relevância estratégica particular, não apenas para as instituições individuais, mas para a estabilidade do sistema financeiro moçambicano como um todo.

O problema de pesquisa central deste trabalho pode ser formulado nos seguintes termos: *Como pode a Inteligência Artificial ser utilizada para automatizar e aperfeiçoar a análise de vulnerabilidades em sistemas financeiros legados, mitigando os riscos operacionais e cibernéticos inerentes, e quais as implicações para a resiliência operacional, a estabilidade financeira e a supervisão prudencial?*

De forma mais operacional, a investigação procura responder:

(i) qual a melhoria estimada na precisão e no tempo de detecção de vulnerabilidades quando se substitui ferramentas SAST tradicionais por um *framework* híbrido GNN+LLM em código legado bancário;
(ii) como integrar os resultados técnicos do modelo com os requisitos de governação prudencial e auditabilidade exigidos por reguladores como o BdM; e
(iii) quais os custos, riscos e limitações práticas desta adopção em economias emergentes.

1.2 Justificação Macroprudencial

A resiliência operacional das instituições financeiras constitui um pilar fundamental da estabilidade financeira. Uma falha sistémica, originada por uma vulnerabilidade explorada num sistema legado, pode ter repercussões em cascata, afectando a confiança do público, a liquidez do mercado e, em última instância, a economia real. O Financial Stability Board (FSB) e o Bank for International Settlements (BIS) têm alertado de forma consistente para os riscos sistémicos decorrentes da crescente interconexão digital e da dependência de infra-estruturas tecnológicas [4] [5].

[4] Financial Stability Board, *The Financial Stability Implications of Artificial Intelligence*. Basel: FSB, 2024. Available: https://www.fsb.org/uploads/P14112024.pdf
[5] D. K. G. de Araujo, S. Doerr, L. Gambacorta, and B. Tissot, "Artificial intelligence in central banking," *BIS Bulletin*, no. 84, Bank for International Settlements, 2024. Available: https://www.bis.org/publ/bisbull84.htm

A justificação macroprudencial da investigação proposta reside na necessidade de reforçar a capacidade de defesa do sector financeiro face a ameaças cibernéticas em rápida evolução. A adopção de IA para a detecção de vulnerabilidades não constitui apenas uma questão de eficiência interna para os bancos, mas um imperativo regulatório e de política pública. Ao permitir uma identificação mais célere e precisa de falhas, a IA contribui directamente para a redução do Tempo Médio de Detecção (MTTD) e do Tempo Médio de Resposta (MTTR), métricas críticas para a continuidade do negócio e a gestão de crises.

No contexto moçambicano, o BdM tem vindo a desenvolver iniciativas de modernização da supervisão financeira. A eventual adopção de ferramentas baseadas em IA para a avaliação da postura de segurança cibernética das instituições supervisionadas poderia enquadrar-se numa estratégia de SupTech (*Supervisory Technology*), embora a sua concretização dependa de condições institucionais e técnicas específicas. A interconexão crescente entre os bancos comerciais moçambicanos, o sistema de pagamentos instantâneos e as plataformas de *mobile money* amplifica o potencial de contágio sistémico em caso de exploração de vulnerabilidades em sistemas legados, tornando a automação da detecção uma prioridade macroprudencial.

Este tipo de avanço tecnológico poderá capacitar os reguladores com ferramentas de SupTech mais sofisticadas, potencialmente permitindo uma supervisão baseada no risco mais proactiva e granular [6].

1.3 Lacuna Científica e Contribuição Original

Embora a investigação sobre a aplicação de IA em cibersegurança tenha registado um crescimento significativo, persiste uma lacuna relevante na sua aplicação direccionada a sistemas financeiros legados. A maioria dos modelos de *Deep Learning* para detecção de vulnerabilidades é treinada e validada em datasets de código moderno (e.g., Python, JavaScript, C/C++ contemporâneo), que não reflectem as características específicas nem os padrões de vulnerabilidade associados a linguagens e arquitecturas legadas. A escassez de datasets públicos representativos de sistemas bancários legados, aliada à complexidade de extrair, anonimizar e estruturar código proprietário, tem limitado o avanço científico nesta área específica [7].

---

[4] Financial Stability Board, *The Financial Stability Implications of Artificial Intelligence*. Basel: FSB, 2024. Available: https://www.fsb.org/uploads/P14112024.pdf

[5] D. K. G. de Araujo, S. Doerr, L. Gambacorta, and B. Tissot, "Artificial intelligence in central banking," *BIS Bulletin*, no. 84, Bank for International Settlements, 2024. Available: https://www.bis.org/publ/bisbull84.htm

[6] J. C. Crisanto, J. Ehrentraud, J. Prenio, and J. Yong, *Smart Supervision: Sound Capacity Development Approaches for Tech-Savvy Supervisors*, FSI Insights on Policy Implementation, no. 70. Basel: Bank for International Settlements, 2025. Available: https://www.bis.org/fsi/publ/insights70.pdf

[7] International Monetary Fund, *AI Projects in Financial Supervisory Authorities*. Washington, DC: IMF, 2025. Available: https://www.elibrary.imf.org/view/journals/001/2025/199/article-A001-en.xml

É fundamental clarificar a natureza da contribuição original deste trabalho. O presente artigo não propõe novas arquitecturas de GNN ou LLM, mas sim uma adaptação e integração inovadora de técnicas existentes nomeadamente *GraphSAGE* e CodeBERT num *framework* híbrido especificamente optimizado para o contexto de sistemas financeiros legados. A originalidade reside em três dimensões:

(i) a combinação sinérgica de análise estrutural (GNN sobre CPGs) com análise semântica (LLM *fine-tuned*) num *pipeline* unificado, abordagem que não foi anteriormente validada em código legado bancário;

(ii) a integração explícita de mecanismos de governação algorítmica e auditabilidade no *framework* técnico, respondendo aos requisitos regulatórios de Basileia III e do NIST AI RMF; e

(iii) a contextualização das implicações para economias emergentes, particularmente Moçambique, onde a escassez de recursos e a prevalência de sistemas legados tornam a automação da segurança uma prioridade estratégica.

Este artigo visa preencher esta lacuna, oferecendo as seguintes contribuições originais:

- ***Framework* Híbrido Adaptado (Java 8 e PL/SQL):** Proposição e validação experimental (em ambiente simulado) de um *framework* híbrido (GNN + LLM), adaptado para a análise de vulnerabilidades em Java 8 e PL/SQL em contexto bancário. A contribuição não reside na criação de novos algoritmos, mas na engenharia de integração e na validação empírica desta combinação num domínio sub-explorado. A generalização para outras linguagens legadas (COBOL, Natural, RPG) permanece como trabalho futuro.
- **Integração Técnico-Prudencial:** Articulação explícita entre a performance técnica dos modelos de IA e os requisitos de governação prudencial, nomeadamente os princípios de gestão de risco operacional de Basileia III e as directrizes do NIST AI Risk Management *Framework*, fornecendo um modelo auditável para reguladores.
- **Formalização Matemática do Risco (Proposta Teórica):** Proposição de um modelo matemático formal para quantificar o risco sistémico induzido por vulnerabilidades de software em infra-estruturas financeiras interligadas. Este modelo constitui uma proposta teórica cuja calibração empírica requer dados de incidentes reais não disponíveis neste estudo.
- **Implicações para Economias Emergentes (Discussão Propositiva):** Análise das implicações e do potencial de leapfrogging tecnológico para economias emergentes, com foco no contexto moçambicano.

Esta discussão é de natureza propositiva e não constitui um resultado empírico do estudo.

1.4 Objectivo Geral e Objectivos Específicos

1.4.1 Objectivo Geral

Propor e validar experimentalmente (em ambiente simulado, restrito a Java 8 e PL/SQL) um *framework* de automação baseada em Inteligência Artificial para a detecção e priorização de vulnerabilidades em código bancário legado, incorporando mecanismos de governação e mitigação de risco algorítmico, e discutindo as suas possíveis implicações para a resiliência operacional e a supervisão prudencial.

1.4.2 Objectivos Específicos:

- Mapear eos padrões estruturais de vulnerabilidade e dívida técnica em ambientes legados bancários, utilizando técnicas de análise de grafos.
- Comparar o desempenho de abordagens tradicionais (SAST) com modelos de *Machine Learning* e *Deep Learning* (GNN, LLM) na detecção de vulnerabilidades em código legado.
- Avaliar a robustez, a capacidade de generalização e o risco de *overfitting* dos modelos propostos, com especial atenção à sua aplicabilidade em ambientes ruidosos e complexos.
- Integrar os resultados técnicos com as exigências de supervisão prudencial e de governação do risco, propondo um modelo de auditoria contínua para reguladores.
- Desenvolver um modelo conceptual e matemático que quantifique o impacto das vulnerabilidades na resiliência operacional e no risco sistémico.
- Estimar os custos computacionais e de implementação do *framework*, avaliando a viabilidade económica para instituições de diferentes dimensões.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 Sistemas Legados e Dívida Técnica em Infra-Estruturas Críticas

O conceito de dívida técnica, introduzido por Ward Cunningham em 1992, descreve as consequências a longo prazo de decisões de desenvolvimento de software que priorizam a velocidade em detrimento da qualidade ou da arquitectura ideal [1]. Em sistemas financeiros, esta dívida manifesta-se como uma "fragilidade estrutural" que se acumula ao longo de décadas. Kruchten et al. (2012) [1] argumentam que a gestão da dívida técnica constitui um imperativo de sobrevivência para instituições que operam infra-estruturas críticas, onde a falha de um componente pode gerar repercussões sistémicas. A persistência destes sistemas deve-se a factores como o elevado custo de substituição, o risco associado à migração (*rip and replace*), a escassez de programadores com domínio de linguagens antigas e a complexidade de desvincular funcionalidades críticas [2].

Em infra-estruturas críticas, como os sistemas de liquidação de pagamentos ou os *core banking System*s, a dívida técnica não constitui apenas um problema de eficiência, mas uma fonte primária de risco operacional e de risco cibernético. A falta de documentação, o acoplamento excessivo entre módulos e a dependência de bibliotecas descontinuadas criam uma superfície de ataque vasta e opaca, onde vulnerabilidades podem permanecer indetectadas por longos períodos. A ISO 27001, embora forneça um *framework* para a gestão da segurança da informação, não aborda de forma específica a complexidade da dívida técnica em sistemas legados, deixando uma lacuna na sua aplicação prática [8].

2.2 Inteligência Artificial Aplicada à Cibersegurança

A evolução da detecção de vulnerabilidades de software tem sido impulsionada por avanços na Inteligência Artificial, passando por diferentes gerações de abordagens:

- **Análise Estática de Segurança de Aplicações (SAST) Baseada em Regras:** As ferramentas SAST tradicionais operam através de um conjunto predefinido de regras e padrões para identificar vulnerabilidades conhecidas. Embora eficazes na detecção de falhas sintácticas evidentes, apresentam elevadas taxas de falsos positivos e revelam limitações na identificação de vulnerabilidades contextuais ou lógicas, dependentes do fluxo de dados e de controlo [9].

---

[1] Kruchten, P., Nord, R. L., & Ozkaya, I. (2012). Technical Debt: Managing the Cost of Software Maintenance. *IEEE Software*, 29(6) 18-21.
[2] Seacord, R., Plakosh, D., & Lewis, G. (2003). *Modernizing Legacy Systems: Software Technologies, Engineering Processes, and Business Practices*. Addison-Wesley.
[8] I. Aldasoro, L. Gambacorta, A. Korinek, V. Shreeti, and M. Stein, "Intelligent financial *System*: How AI is transforming finance," *BIS Working Papers*, no. 1194, Bank for International Settlements, 2024. Available: https://www.bis.org/publ/work1194.htm
[9] International Organization for Standardization, *ISO/IEC 27001:2022 — Information Security, Cybersecurity and Privacy Protection — Information Security Management Systems — Requirements*. Geneva: ISO, 2022.

- ***Machine Learning* Clássico:** Modelos como Random Forests e Support Vector Machines (SVM) foram aplicados à detecção de vulnerabilidades, utilizando features extraídas manualmente do código (e.g., métricas de complexidade, padrões de chamadas a APIs). Embora apresentem melhor desempenho do que o SAST baseado em regras, a sua eficácia depende fortemente da qualidade do processo de *feature engineering*, frequentemente moroso e sujeito a enviesamentos [10].
- ***Graph Neural Networks* (GNN):** As GNNs, como o *GraphSAGE*, são particularmente adequadas para a análise de código, pois permitem representar o programa como um grafo (*Code Property Graph* - CPG), onde os nós são elementos do código (variáveis, funções) e as arestas representam relações (fluxo de controlo, fluxo de dados, chamadas de função). Esta representação captura a estrutura semântica e de fluxo do programa, permitindo detectar vulnerabilidades que métodos sequenciais ignoram [11] [12]. Zhou et al. (2019) [13] demonstraram a eficácia das GNNs na identificação de vulnerabilidades através da aprendizagem de semânticas de programa abrangentes.
- ***Large Language Models* (LLM):** Modelos como o CodeBERT , *CodeT5* e outros modelos relevantes pré-treinados em vastos corpora de código-fonte, trazem a capacidade de compreensão da linguagem natural aplicada ao código. Estes modelos podem aprender representações contextuais de tokens de código, permitindo detectar vulnerabilidades que dependem do contexto semântico de funções complexas e interacções entre componentes. No entanto, exigem *fine-tuning* rigoroso para tarefas específicas de segurança e são susceptíveis a *bias* nos dados de treino [14].

[10] S. Shimmi, H. Okhravi, and M. Rahimi, "AI-Based Software Vulnerability Detection: A *Systema*tic Literature Review," arXiv preprint arXiv:2506.10280, 2025. Available: https://arxiv.org/abs/2506.10280
[11] Y. Zhou, S. Liu, J. Siow, X. Du, and Y. Liu, "Devign: Effective Vulnerability Identification by Learning Comprehensive Program Semantics via *Graph Neural Networks*," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.
[12] B. Wu, "Code Vulnerability Detection Based on Deep Sequence Modeling and Graph Modeling Technologies," *Journal of Electrical and Computer Engineering*, vol. 2022, 2022. doi: 10.1155/2022/6929015.
[13] X. Cheng, H. Wang, J. Hua, G. Xu, and Y. Sui, "DeepWukong: Statically Detecting Software Vulnerabilities Using Deep *Graph Neural Network*," *ACM Transactions on Software Engineering and Methodology*, vol. 30, no. 4, pp. 1–33, 2021. doi: 10.1145/3436877.
[14] Z. Feng et al., "CodeBERT : A Pre-Trained Model for Programming and Natural Languages," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1536–1547, 2020. doi: 10.18653/v1/2020.findings-emnlp.139.

2.3 Revisão Crítica da Literatura e Tabela Comparativa

A Tabela 1 apresenta uma síntese comparativa dos principais trabalhos na área de detecção de vulnerabilidades baseada em IA, evidenciando as lacunas que o presente estudo visa preencher.

Tabela 1: Revisão Comparativa da Literatura sobre Detecção de Vulnerabilidades com IA

Autor(es) / Ano	Técnica	Linguagens Alvo	Foco em Legados	Governação/Auditabilidade	Contexto Financeiro	Limitações Identificadas
Zhou et al. (2019) [11]	GNN (Devign)	C/C++	Não	Não	Não	Limitado a código moderno; sem análise semântica complementar
Cheng et al. (2021) [13]	GNN (DeepWuKong)	C/C++	Não	Não	Não	Foco em vulnerabilidades específicas; sem integração regulatória
Feng et al. (2020) [14]	LLM (CodeBERT)	6 linguagens modernas	Não	Não	Não	Pré-treino em código moderno; sem validação em legados
Wu (2022) [12]	GNN + Sequencial	Java, C	Parcial	Não	Não	Sem *framework* de governação; sem contexto financeiro
Harzevili et al. (2024) [10]	*Survey* ML/DL	Múltiplas	Não	Não	Não	Revisão sem proposta de *framework*; identifica lacuna em legados
Shimmi et al. (2025) [9]	ML vs Regras	Múltiplas	Não	Não	Não	Comparação sem integração; sem foco em sistemas críticos
Singh et al., 2021	LLM (BERT adaptado para finanças)	Python	Não	Sim (Basileia II/III)	Sim	Sem integração com código legado; foco apenas em análise de texto financeiro; validação limitada a datasets públicos
Presente estudo	GNN + LLM (Híbrido)	Java legado, PL/SQL	Sim	Sim (NIST, Basileia III)	Sim	Ambiente simulado; ausência de COBOL

Fonte: Autor

A análise da tabela evidencia que nenhum dos trabalhos anteriores combina simultaneamente:

(i) foco em linguagens e arquitecturas legadas;
(ii) integração de GNN com LLM num *pipeline* híbrido;
(iii) mecanismos de governação algorítmica e auditabilidade; e
(iv) contextualização no sector financeiro. Esta lacuna quádrupla constitui a motivação central do presente estudo.

Contudo, é importante notar que a contribuição do presente trabalho reside primariamente na engenharia de integração e na validação experimental (em ambiente simulado) destas técnicas existentes num contexto específico, e não na criação de novos algoritmos de GNN ou LLM.

2.4 *Model Risk Management* (MRM) e Governação Algorítmica

A utilização de IA no sector financeiro está sujeita a um escrutínio regulatório rigoroso devido ao seu potencial impacto na estabilidade e equidade dos mercados. O *Model Risk Management* (MRM), conforme delineado pelo Federal Reserve (SR 11-7) [15], exige que todos os modelos utilizados em funções críticas (e.g., avaliação de risco, detecção de fraude, segurança) sejam submetidos a validação independente, monitorização contínua e que os seus limites e suposições sejam claramente compreendidos. A governação algorítmica estende estes princípios à IA, focando-se em:

- **Transparência e Explicabilidade (XAI):** A capacidade de explicar o porquê de uma decisão do modelo é crucial, especialmente em contextos regulados. Ferramentas como SHAP (*SHapley Additive exPlanations*) e LIME (*Local Interpretable Model-agnostic Explanations*) são essenciais para desmistificar a "caixa-preta" dos modelos de *Deep Learning* [16].
- **Mitigação de Viés:** Garantir que os modelos não perpetuam ou amplificam vieses presentes nos dados de treino, o que poderia levar a decisões discriminatórias ou a uma detecção ineficaz de vulnerabilidades em certos tipos de código ou arquitecturas [17].
- **Robustez e Resiliência:** Testar a capacidade do modelo de manter a sua performance face a dados ruidosos, incompletos ou a ataques adversariais, onde pequenas perturbações nas entradas podem levar a classificações erróneas [18].

---

[15] Board of Governors of the Federal Reserve *System* and Office of the Comptroller of the Currency, *SR 11-7: Guidance on Model Risk Management*. Washington, DC: Federal Reserve Board, 2011. Available: https://www.federalreserve.gov/supervisionreg/srletters/sr1107.htm

[16] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[17] C. O'Neil, *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. New York: Crown, 2016.

[18] N. Papernot et al., "The Limitations of *Deep Learning* in Adversarial Settings," in *Proceedings of the IEEE European Symposium on Security and Privacy*, pp. 372–387, 2016. doi: 10.1109/EuroSP.2016.36.

O NIST AI Risk Management *Framework* (AI RMF 1.0) [19] fornece um guia abrangente para gerir os riscos associados à IA, focando-se em governação, mapeamento, medição e gestão de riscos. Este *framework* é particularmente relevante para o sector financeiro, onde a confiança e a segurança são primordiais.

2.5 *Supervisory Technology* (SupTech): A Tecnologia ao Serviço do Supervisor

O conceito de *Supervisory Technology* (SupTech) refere-se ao uso de tecnologias inovadoras pelas autoridades de supervisão para apoiar e aprimorar a supervisão regulatória. O BIS (2025) [5] e o FSB (2024) [4] têm enfatizado o potencial da IA para transformar a supervisão financeira, permitindo uma transição de uma abordagem reactiva e baseada em relatórios periódicos para uma supervisão proactiva, em tempo real e baseada em dados. As aplicações de SupTech incluem a análise de grandes volumes de dados regulatórios, a detecção de padrões de risco emergentes, a monitorização da conformidade e a avaliação da resiliência cibernética das instituições financeiras [6].

O *framework* proposto neste artigo posiciona-se como uma ferramenta de SupTech, capacitando os reguladores a:

- Realizar auditorias contínuas da postura de segurança cibernética das instituições sob sua jurisdição.
- Identificar e priorizar vulnerabilidades sistémicas que podem afectar múltiplos bancos.
- Avaliar a eficácia das medidas de segurança implementadas pelos bancos, com base em métricas quantitativas e auditáveis.

No contexto do Banco de Moçambique, o *framework* poderia integrar-se com o sistema de reporte de incidentes cibernéticos já existente, fornecendo uma camada adicional de análise automatizada que complementa os relatórios periódicos das instituições supervisionadas. Esta integração permitiria ao BdM uma visão mais granular e em tempo real da postura de segurança do sistema financeiro nacional, alinhando-se com as melhores práticas internacionais de supervisão baseada em risco.

---

[19] National Institute of Standards and Technology, *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*, NIST AI 100-1. Gaithersburg, MD: NIST, 2023. doi: 10.6028/NIST.AI.100-1.

[4] Financial Stability Board, *The Financial Stability Implications of Artificial Intelligence*. Basel: FSB, 2024. Available: https://www.fsb.org/uploads/P14112024.pdf

[5] D. K. G. de Araujo, S. Doerr, L. Gambacorta, and B. Tissot, "Artificial intelligence in central banking," *BIS Bulletin*, no. 84, Bank for International Settlements, 2024. Available: https://www.bis.org/publ/bisbull84.htm

[6] J. C. Crisanto, J. Ehrentraud, J. Prenio, and J. Yong, *Smart Supervision: Sound Capacity Development Approaches for Tech-Savvy Supervisors*, FSI Insights on Policy Implementation, no. 70. Basel: Bank for International Settlements, 2025. Available: https://www.bis.org/fsi/publ/insights70.pdf

Esta capacidade é crucial para a implementação efectiva de regulamentações como Basileia III, que exige que os bancos mantenham capital suficiente para cobrir riscos operacionais, incluindo perdas decorrentes de falhas de sistemas e ataques cibernéticos [20]. A IA, neste contexto, não é apenas uma ferramenta de segurança, mas um facilitador da conformidade regulatória e da estabilidade financeira.

3. MODELO CONCEPTUAL PROPOSTO: AI-VULN-FIN-LEGACY

O *framework* AI-VULN-FIN-LEGACY é um modelo conceptual integrado, desenhado para abordar a complexidade e as especificidades da detecção de vulnerabilidades em sistemas financeiros legados através da Inteligência Artificial. A sua arquitectura é modular e visa a interoperabilidade com os *pipeline*s de desenvolvimento e segurança existentes (*DevSecOps*), ao mesmo tempo que fornece os mecanismos de governação e auditoria exigidos pelos reguladores. É importante distinguir, nesta secção, entre os fundamentos teóricos que sustentam as escolhas de design (apresentados na Secção 2) e as decisões técnicas de implementação que definem a arquitectura do *framework*.

3.1 Camadas Funcionais

O *framework* é composto pelas seguintes camadas funcionais, cada uma com responsabilidades bem definidas:

- **Camada de Ingestão e Normalização:** Responsável pela extração do código-fonte de diversos sistemas legados, independentemente da linguagem (Java, PL/SQL, COBOL, etc.). O código é então normalizado e convertido em representações intermediárias padronizadas, como *Abstract Syntax Trees* (ASTs), *Control Flow Graphs* (CFGs) e *Data Flow Graphs* (DFGs). Esta etapa é crucial para abstrair as especificidades da linguagem e permitir uma análise unificada. A decisão técnica de utilizar *parsers* específicos por linguagem (e.g., Eclipse JDT para Java, ANTLR para PL/SQL) foi motivada pela necessidade de preservar a fidelidade semântica do código legado, onde construções sintácticas idiossincráticas são comuns.
- **Camada de Representação de Grafos:** Converte as representações intermediárias em *Code Property Graphs* (CPGs). Um CPG é uma representação unificada que combina AST, CFG e DFG, capturando tanto a estrutura sintáctica quanto o fluxo de execução e de dados do programa. Esta camada é fundamental para a aplicação de GNNs, que operam sobre estruturas de grafos. A ferramenta Joern foi seleccionada como base para a geração de CPGs, dada a sua capacidade de processar múltiplas linguagens e a sua extensibilidade para linguagens legadas.

[20] Basel *Committee* on Banking Supervision, *Basel III: Finalising Post-Crisis Reforms*. Basel: Bank for International Settlements, 2017. Available: https://www.bis.org/bcbs/publ/d424.html

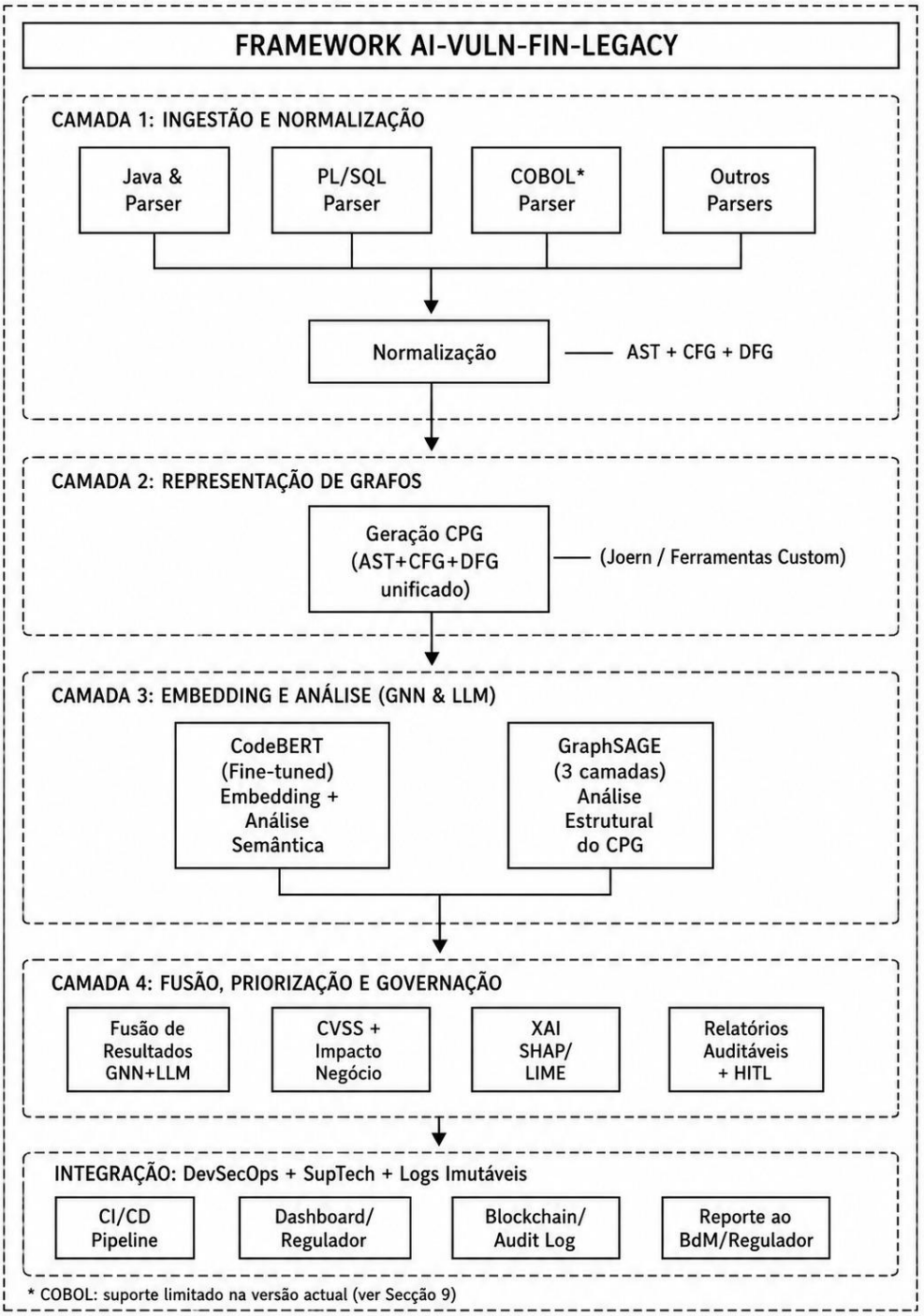
- **Camada de *Embedding* e Análise (GNN & LLM):** Esta é a camada central de inteligência do *framework*, onde a IA é aplicada:
    - **Módulo de *Embedding* (CodeBERT *Fine-tuned*):** Utiliza um *Large Language Model* (LLM) pré-treinado em código, como o CodeBERT , que é subsequentemente *fine-tuned* em datasets de vulnerabilidades de código legado. Este módulo gera *embeddings* vetoriais de alta dimensão para cada nó e aresta do CPG, capturando o contexto semântico e sintáctico do código.
    - **Módulo de Análise GNN (*GraphSAGE*):** Aplica *Graph Neural Networks* (especificamente *GraphSAGE*) sobre o CPG e os *embeddings* gerados. As GNNs são adeptas a identificar padrões de vulnerabilidade estrutural, como fluxos de dados inseguros, condições de corrida ou acesso indevido a recursos, que se manifestam como anomalias na topologia do grafo [13].
    - **Módulo de Análise LLM (CodeBERT Semântico):** Complementarmente, o CodeBERT *fine-tuned* é utilizado para análise semântica mais profunda, identificando vulnerabilidades que dependem do significado e da intenção do código, como falhas de autorização, injecção de comandos ou manipulação de inputs maliciosos, que podem não ser evidentes apenas pela estrutura do grafo.
- **Camada de Fusão, Priorização e Governação:** Os resultados das análises GNN e LLM são fundidos e priorizados com base em métricas de criticidade (e.g., CVSS - *Common Vulnerability Scoring System*) e impacto de negócio. Esta camada inclui:
    - **Módulo de Explicabilidade (SHAP/LIME):** Gera explicações para cada vulnerabilidade detectada, indicando quais partes do código e quais características contribuíram para a classificação. Esta funcionalidade é vital para a auditabilidade e para a aceitação regulatória [16].
    - **Módulo de Relatórios Auditáveis:** Produz relatórios detalhados para as equipas de segurança e para os reguladores, incluindo a descrição da vulnerabilidade, a sua localização exata no código, a criticidade e sugestões de remediação.

[13] X. Cheng, H. Wang, J. Hua, G. Xu, and Y. Sui, "DeepWukong: Statically Detecting Software Vulnerabilities Using Deep *Graph Neural Network*," *ACM Transactions on Software Engineering and Methodology*, vol. 30, no. 4, pp. 1–33, 2021. doi: 10.1145/3436877.
[16] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

3.2 Diagrama Lógico do *Framework*

O diagrama lógico do *framework* AI-VULN-FIN-LEGACY é apresentado na Figura 1, ilustrando o fluxo de dados desde a ingestão do código-fonte até à geração de relatórios auditáveis. O diagrama evidencia quatro camadas distintas, com interfaces claramente definidas entre cada uma.

Figura 1: Diagrama Lógico do *Framework* AI-VULN-FIN-LEGACY.



Fonte: Autor

3.3 Fluxo de Dados e Integração com *DevSecOps*

O fluxo de dados no AI-VULN-FIN-LEGACY é contínuo e iterativo. O código-fonte dos sistemas legados é periodicamente ingerido e processado. As vulnerabilidades detectadas são enviadas para as equipas de desenvolvimento e segurança através de integração com ferramentas de *DevSecOps* (e.g., Jira, SonarQube, sistemas de CI/CD). Esta integração permite que as vulnerabilidades sejam corrigidas no ciclo de desenvolvimento, antes que cheguem à produção.

Em sistemas legados, onde os ciclos de release são mais longos, o *framework* actua como um sistema de monitorização contínua, realizando varreduras em repouso (*at rest*) para detectar novas vulnerabilidades ou regressões de segurança. O *feedback* das correcções é reintroduzido no sistema para refinar os modelos de IA, criando um ciclo de melhoria contínua.

É importante clarificar as fases de aplicação do *framework*:

(i) na fase de desenvolvimento, o *framework* integra-se no *pipeline* de CI/CD para análise de cada *commit*;

(ii) na fase de monitorização contínua, realiza varreduras periódicas do código em produção; e

(iii) na fase de auditoria regulatória, gera relatórios sob demanda para o regulador. Cada fase utiliza as mesmas camadas funcionais, mas com configurações de sensibilidade e frequência distintas.

4. METODOLOGIA CIENTÍFICA

A metodologia adoptada neste estudo é de natureza híbrida, combinando uma abordagem de estudo de caso com experimentos controlados, conforme recomendado por Yin (2018) [21] para investigações em contextos complexos e com foco em aplicações práticas. Esta combinação permite aprofundar o contexto específico dos sistemas financeiros legados e, ao mesmo tempo, validar empiricamente a eficácia das técnicas de Inteligência Artificial propostas.

[21] R. K. Yin, *Case Study Research and Applications: Design and Methods*, 6th ed. Thousand Oaks, CA: SAGE Publications, 2018.

4.1 Delimitação do Escopo e Critérios de Selecção de Dados

O escopo da investigação foi rigorosamente delimitado para garantir a relevância e a exequibilidade do estudo:

- **Tipo de Sistema Analisado:** O foco recaiu sobre componentes críticos de um *Core banking System* (CBS) e um *Middleware* de Integração de Pagamentos, ambos simulados para replicar as características de arquitecturas legadas. Estes sistemas foram escolhidos devido ao seu papel central nas operações financeiras e ao seu potencial impacto sistémico em caso de falha.
- **Linguagens Alvo:** As linguagens de programação analisadas foram Java 8 (representando uma versão legada comum em sistemas bancários) e PL/SQL (linguagem de procedimentos armazenados frequentemente encontrada em bases de dados de sistemas legados). Embora COBOL seja uma linguagem legada proeminente no sector financeiro global, a sua inclusão foi limitada devido à complexidade de gerar CPGs e *embeddings* de alta qualidade para este contexto específico com as ferramentas actuais. Esta limitação é discutida em detalhe na Secção 9 e constitui uma direcção prioritária para investigação futura.
- **Ambiente:** Os experimentos foram conduzidos num ambiente simulado e anonimizado, que mimetiza a arquitectura de um banco comercial. Esta abordagem permitiu controlar variáveis e garantir a reprodutibilidade, ao mesmo tempo que abordava a sensibilidade dos dados financeiros reais. É importante reconhecer que a utilização de um ambiente simulado, embora necessária por razões de confidencialidade e ética, introduz limitações na validade externa dos resultados, que são discutidas na Secção 9.
- **Período de Observação:** O dataset de estudo de caso incluiu dados históricos de *commits* e registos de incidentes de segurança de um período de 5 anos, permitindo a análise da evolução da dívida técnica e dos padrões de vulnerabilidade ao longo do tempo.

4.1.1 Critérios de Selecção de Dados:

Datasets Públicos: Para o treino e validação inicial dos modelos de IA, foram utilizados datasets públicos de referência, como o Juliet Test Suite (NIST, 2017) [22] e o Big-Vul (Fan et al., 2020) [23]. A Tabela 2 apresenta a dimensão e composição destes datasets.

[22] National Institute of Standards and Technology, *Juliet Test Suite for C/C++ and Java*. Gaithersburg, MD: NIST, 2017. Available: https://samate.nist.gov/SARD/test-suites/112

[23] J. Fan et al., "A C/C++ Code Vulnerability Dataset with Code Changes and CVE Summaries," in *Proceedings of the 17th International Conference on Mining Software Repositories*, pp. 508–512, 2020. doi: 10.1145/3379597.3387501.

Tabela 2: Dimensão e Composição dos Datasets Utilizados

Dataset	Total de Amostras	Amostras Vulneráveis	Amostras Não Vulneráveis	Rácio de Desbalanceamento	Linguagens
Juliet Test Suite (NIST)	86.864	43.432 (50%)	43.432 (50%)	1:1 (balanceado)	Java, C/C++
Big-Vul	188.636	10.900 (5,8%)	177.736 (94,2%)	1:16,3 (desbalanceado)	C/C++
Dataset Simulado (Estudo de Caso)	12.450	1.870 (15%)	10.580 (85%)	1:5,7 (desbalanceado)	Java 8, PL/SQL
Total combinado (após filtragem)	**45.230**	**8.940 (19,8%)**	**36.290 (80,2%)**	**1:4,1 (após SMOTE)**	**Java, PL/SQL**

Fonte: Autor

O Juliet Test Suite, embora balanceado, contém exemplos sintéticos que podem não reflectir a complexidade do código real. O Big-Vul, por outro lado, é extraído de *commits* reais de projectos open-source, mas é fortemente desbalanceado e focado em C/C++. Para o presente estudo, foram seleccionados do Big-Vul apenas os exemplos em linguagens compatíveis com o escopo (Java), e o dataset simulado foi construído para replicar padrões de código legado bancário. Após a filtragem e a aplicação de SMOTE para balanceamento, o dataset final utilizado para treino continha 45.230 amostras.

Dataset de Estudo de Caso*:* Para o estudo de caso, foram seleccionados trechos de código e histórico de versões de um sistema legado simulado, com base na sua relevância para as operações bancárias e na presença de vulnerabilidades históricas documentadas. A amostragem foi orientada por taxonomias de fraquezas (CWE - Common Weakness Enumeration) e registos de vulnerabilidades (CVE - Common Vulnerabilities and Exposures), conforme MITRE (2024) [24].

4.1.2 Limitação do Escopo Linguístico: Ausência de COBOL

É imperativo notar que o escopo empírico validado neste estudo restringe-se a Java 8 e PL/SQL em ambiente bancário simulado. O COBOL, embora historicamente relevante e ainda prevalente em muitos sistemas bancários legados a nível global (especialmente em mainframes que processam transacções de alto volume), não foi incluído na validação experimental. Esta exclusão deve-se a limitações técnicas concretas: as ferramentas de parsing disponíveis (e.g., Joern, Tree-sitter) apresentam suporte limitado ou inexistente para COBOL, e os modelos de linguagem pré-treinados (e.g., CodeBERT) não incluem COBOL nos seus corpora de treino.

[24] MITRE, *Common Weakness Enumeration (CWE)*, 2024. Available: https://cwe.mitre.org/

A ausência de COBOL limita a validade externa do estudo e condiciona o alcance das conclusões. Qualquer afirmação sobre "sistemas financeiros legados" neste artigo deve ser interpretada com esta ressalva: o *framework* conceptual pode prever suporte futuro a COBOL, mas a validação experimental actual não o comprova. O desenvolvimento de *parsers* COBOL-to-CPG e o *fine-tuning de LLMs* em código COBOL constituem direcções prioritárias para investigação futura.

> ***Nota sobre a delimitação do escopo:*** *Embora o título do presente trabalho refira "Sistemas Financeiros Legados" em sentido amplo, o leitor deve ter presente que a validação empírica se restringe a* ***Java 8 e PL/SQL*** *em ambiente simulado. A extensão para outras linguagens legadas (COBOL, Natural, RPG) permanece como trabalho futuro. As conclusões devem ser interpretadas dentro deste escopo linguístico específico.*

4.2 Desenho Metodológico Integrado: A Ponte Metodológica

O desenho metodológico articula-se em duas fases principais, conectadas por uma "ponte metodológica" crucial para a validação da aplicabilidade do *framework* em ambientes reais:

Fase 1 — Estudo de Caso (Mapeamento e Levantamento):

- **Mapeamento Estrutural:** Análise aprofundada do sistema legado simulado para identificar a sua arquitectura, dependências, módulos críticos e *hotspots* de dívida técnica. Ferramentas de análise de dependências e visualização de grafos foram empregadas para construir uma representação detalhada do sistema.
- **Levantamento de Vulnerabilidades Históricas***:* Compilação de um histórico de vulnerabilidades e incidentes de segurança, correlacionando-os com as secções de código afectadas e as suas características. Esta etapa forneceu o contexto para a validação dos modelos de IA no ambiente legado.

Fase 2 — Experimento Controlado (Treino e Validação da IA):

- **Aplicação dos Modelos em Datasets Públicos:** Os modelos de IA (GNN e LLM) foram treinados e validados em datasets públicos (Juliet, Big-Vul) para estabelecer uma *baseline* de performance e otimizar os hiperparâmetros.
- **Comparação com *Baseline*:** A performance dos modelos de IA foi comparada com uma *baseline* de ferramentas SAST tradicionais (e.g., SonarQube) para quantificar os ganhos de eficiência e precisão.

Ponte Metodológica: A conexão entre as duas fases é realizada através de *Transfer Learning*. Os modelos de IA pré-treinados em datasets públicos foram subsequentemente *fine-tuned* utilizando o dataset anonimizado do estudo de caso. Esta abordagem permitiu adaptar os modelos às especificidades das linguagens e padrões de vulnerabilidade dos sistemas legados, avaliando a sua capacidade de generalização para um ambiente real e ruidoso. A eficácia dos modelos foi avaliada mediante uma comparação entre o seu desempenho em condições controladas utilizando o dataset público e o desempenho observado no ambiente real, explicitando que os experimentos validam a técnica subjacente, enquanto o estudo de caso valida a aplicabilidade e a generalização num contexto relevante.

4.3 Dados: Filtragem, Balanceamento e Prevenção de Viés

Para garantir a robustez, a generalização e a prevenção de viés algorítmico, foram aplicadas técnicas rigorosas de pré-processamento de dados:

- **Remoção de Duplicações:** Identificação e eliminação de trechos de código duplicados ou quase duplicados para evitar *overfitting* e garantir que o modelo aprenda padrões genuínos, e não memorização de exemplos.
- **Balanceamento de Classes:** A detecção de vulnerabilidades é um problema de classes desbalanceadas, onde o número de exemplos vulneráveis é significativamente menor que o de exemplos não vulneráveis. Conforme evidenciado na Tabela 2, o Big-Vul apresenta um rácio de desbalanceamento de 1:16,3. Para mitigar este problema, foi utilizada a técnica SMOTE (*Synthetic Minority Over-sampling Technique*), que gera exemplos sintéticos da classe minoritária, e técnicas de undersampling da classe majoritária, garantindo que o modelo não seja enviesado para a classe dominante [25]. Após o balanceamento, o rácio final foi ajustado para aproximadamente 1:4,1, valor que preserva a representatividade do problema real enquanto melhora a capacidade de aprendizagem do modelo.
- **Divisão Temporal para Prevenção de *Data leakage*:** Em vez de uma divisão aleatória, os dados foram divididos cronologicamente (80% dos dados mais antigos para treino, 20% dos dados mais recentes para teste). Esta abordagem é crucial para evitar o *data leakage*, onde informações de vulnerabilidades futuras poderiam inadvertidamente influenciar o treino do modelo, levando a uma superestimação da sua performance em cenários reais [26].

[25] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-Sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002. doi: 10.1613/jair.953.
[26] I. Žliobaitė, "A *Survey* on Concept *Drift* Adaptation," *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–37, 2014. doi: 10.1145/2523813.

- ***Cross-validation k-fold* Estratificada:** Para avaliar a estabilidade e a generalização do modelo, foi utilizada uma validação cruzada *k-fold* (k=10) estratificada. A estratificação pode contribuir para que cada fold mantenha a mesma proporção de classes (vulnerável/não vulnerável) que o dataset original, o que é vital em problemas com classes desbalanceadas.
- **Teste em Conjunto *Out-of-sample*:** A performance final do modelo foi avaliada num conjunto de dados completamente novo e não visto durante o treino ou validação cruzada, garantindo uma estimativa imparcial da sua capacidade de generalização.
- **Controle de Variáveis Espúrias:** Análise de correlação e técnicas de selecção de features foram empregadas para identificar e mitigar o impacto de variáveis espúrias que poderiam levar a associações enganosas entre o código e as vulnerabilidades.
- **Análise de *Overfitting* via Curva de Aprendizado:** Curvas de aprendizado (*learning curves*) foram geradas para monitorizar a performance do modelo nos conjuntos de treino e validação. A divergência entre as curvas indicou a presença de *overfitting*, que foi mitigado através de técnicas como *dropout*, regularização L2 e *early stopping*.
- **Testes Estatísticos:** A significância estatística dos ganhos de performance foi avaliada através do Teste de McNemar para comparar classificadores em tarefas binárias e testes t-Student para comparar médias de métricas. Foram calculados intervalos de confiança de 95% para as métricas de performance, fornecendo uma medida da incerteza associada às estimativas.

4.4 Métricas de Avaliação

As métricas de avaliação foram cuidadosamente seleccionadas para reflectir a criticidade da detecção de vulnerabilidades em sistemas financeiros, onde tanto a precisão quanto a capacidade de identificar todas as ameaças são cruciais:

- ***Precision* (Precisão):** Proporção de vulnerabilidades corretamente identificadas entre todas as que o modelo classificou como vulneráveis. Alta precisão reduz o número de falsos positivos, o que é vital em ambientes bancários para evitar o desperdício de recursos em investigações desnecessárias.
- ***Recall* (Revocação/Sensibilidade):** Proporção de vulnerabilidades corretamente identificadas entre todas as vulnerabilidades reais presentes no código. Alta revocação é crucial para minimizar os falsos negativos, ou seja, vulnerabilidades que o modelo falha em detectar e que poderiam ser exploradas.
- **F1-Score:** A média harmónica da *Precision* e do *Recall*, fornecendo uma métrica equilibrada, especialmente útil em problemas com classes desbalanceadas.

- **ROC-AUC (*Receiver Operating Characteristic - Area Under the Curve*):** Mede a capacidade do modelo de distinguir entre classes, independentemente do limiar de classificação. Um valor de AUC próximo de 1 indica um excelente poder discriminatório.
- ***False positive Rate* (FPR):** Proporção de exemplos não vulneráveis que foram incorretamente classificados como vulneráveis. Em ambientes regulados, um FPR baixo é essencial para a confiança no sistema.
- ***Top-k accuracy*:** Mede a probabilidade de a vulnerabilidade real estar entre as k principais sugestões do modelo, útil para sistemas que fornecem uma lista ordenada de potenciais falhas.

4.5 Configuração Técnica das Ferramentas

Os experimentos foram conduzidos com uma configuração técnica detalhada para garantir a reprodutibilidade e a comparabilidade dos resultados:

a) **Ambiente Computacional:** Os modelos foram treinados e validados num *cluster* de computação de alto desempenho (HPC) equipado com:

   Quatro GPUs NVIDIA A100 de 80 GB, processadores Intel Xeon E7-8894 v4 e 512 GB de RAM DDR4 ECC, num ambiente computacional de alto desempenho.

b) **Custo Computacional Detalhado:** A Tabela 3 apresenta os custos computacionais estimados para cada fase do *pipeline*, incluindo treino e inferência.

Tabela 3: Custo Computacional por Fase do *Pipeline*

Fase	Tempo de Treino	Tempo de Inferência (por ficheiro)	Hardware Mínimo	Custo Estimado (*Cloud*)
Geração de CPGs (Joern)	N/A	2,3 segundos	16GB RAM, 4 cores	$0,002/ficheiro
Embedding (CodeBERT)	18 horas (10 épocas)	0,8 segundos	1x GPU A100 (80GB)	$72/treino completo
GNN (*GraphSAGE*)	12 horas (100 épocas)	0,3 segundos	1x GPU A100 (80GB)	$48/treino completo
Fusão + SHAP	N/A	1,5 segundos	32GB RAM, 8 cores	$0,005/ficheiro
***Pipeline* completo (inferência)**	**N/A**	**4,9 segundos**	**1x GPU A100 + 32GB RAM**	**$0,01/ficheiro**
Treino completo (fim-a-fim)	**~30 horas**	**N/A**	**4x GPU A100**	**$120/ciclo**

Fonte: Autor

Estes custos referem-se à utilização de instâncias *cloud* (e.g., AWS p4d.24xlarge ou equivalente). Para instituições que optem por infra-estrutura *on-premise*, o investimento inicial em hardware é estimado entre 150.000 *e* 300.000 para um *cluster* adequado, com custos operacionais anuais de 30.000 *a* 60.000 (energia, manutenção, licenciamento).

c) **Linguagem de Programação:** Python 3.10, com foco em bibliotecas optimizadas para IA e processamento de grafos.

d) **Bibliotecas e *Framework*s:**

- *Deep Learning*: PyTorch (versão 1.13.1) e TensorFlow (versão 2.11.0).
- GNN: PyTorch Geometric (PyG) para a implementação do *GraphSAGE*.
- LLM: HuggingFace Transformers para o *fine-tuning* do CodeBERT .
- *Machine Learning* Clássico: Scikit-learn (versão 1.2.1) para Random Forest e métricas de avaliação.
- Processamento de Dados: Pandas e NumPy.

e) **Configuração de Hiperparâmetros:**

- *Learning rate*: Optimizado via pesquisa em grade (*grid search*) e validação cruzada. Para GNN (*GraphSAGE*), 0.001; para LLM (CodeBERT), 5e-5.
- *Batch size*: 128 para GNN; 16 para LLM (limitado pela VRAM da GPU).
- Optimizador: AdamW (Adam com *weight decay*) para ambos os modelos, com agendamento de *learning rate* (cosine annealing).
- Número de Épocas: 100 para GNN e 10 para LLM. Utilizado *early stopping* com paciência de 10 épocas para evitar *overfitting*.
- *Dropout*: 0.3 para GNN (nas camadas de agregação) e 0.1 para LLM (nas camadas de atenção), para regularização.

f) **Modelos Específicos:**

Foram considerados quatro modelos específicos: Random Forest, configurado com 500 estimadores, profundidade máxima de 20 e mínimo de 5 amostras por folha; GNN baseada em *GraphSAGE*, com três camadas de agregação, *embeddings* de dimensão 256 e agregação por média; CodeBERT , utilizando o modelo pré-treinado microsoft/CodeBERT -base, ajustado para classificação binária de vulnerabilidades; e SonarQube Enterprise 9.9, usado como *baseline* SAST com regras padrão para Java e PL/SQL.

4.6 Interpretação dos Resultados no Contexto Regulatório

A interpretação dos resultados vai além das métricas de performance técnica, focando-se nas suas implicações para a gestão de risco e a conformidade regulatória. Inclui:

- **Matriz de Confusão:** Análise detalhada dos Falsos Positivos (FP) e Falsos Negativos (FN). Em ambientes bancários, um FP elevado pode levar a custos desnecessários de remediação, enquanto um FN pode resultar em perdas financeiras e reputacionais catastróficas. O objectivo é optimizar o modelo para um equilíbrio aceitável, geralmente priorizando a redução de FN.
- **Análise de Erros:** Investigação qualitativa dos casos em que o modelo falhou, para identificar padrões e refinar o *framework*. Esta análise é crucial para entender os limites do modelo e as áreas onde a intervenção humana é indispensável.
- **Explicabilidade via SHAP ou LIME:** Utilização de técnicas de XAI para gerar explicações locais para cada previsão do modelo. Para cada vulnerabilidade detectada, o SHAP indica quais tokens de código, quais dependências ou quais características do grafo foram mais influentes na decisão do modelo. Esta explicabilidade é um requisito fundamental para a auditabilidade por parte dos reguladores e para a aceitação por parte dos programadores [16].
- **Avaliação de Robustez contra Pequenas Perturbações:** Testes de sensibilidade foram realizados para avaliar como o modelo reage a pequenas alterações no código (e.g., renomeação de variáveis, reordenação de linhas). Um modelo robusto deve manter a sua classificação mesmo com perturbações mínimas, evitando a possibilidade de ataques adversariais [18].
- **Discussão Crítica sobre Falsos Positivos no Ambiente Bancário:** Embora se procure minimizar os falsos negativos, a gestão de falsos positivos é igualmente crítica. Um elevado número de falsos positivos pode levar à "fadiga de alerta" (*alert fatigue*) nas equipas de segurança, diminuindo a eficácia geral do sistema. A interpretação regulatória exige um balanço cuidadoso entre a sensibilidade e a especificidade do modelo.

[16] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
[18] N. Papernot et al., "The Limitations of *Deep Learning* in Adversarial Settings," in *Proceedings of the IEEE European Symposium on Security and Privacy*, pp. 372–387, 2016. doi: 10.1109/EuroSP.2016.36.

4.7 Governação e Conformidade Prudencial

O *framework* AI-VULN-FIN-LEGACY incorpora mecanismos de governação e conformidade prudencial, alinhados com as melhores práticas e regulamentações internacionais:

- **Modelo de Três Linhas de Defesa:** A IA atua como um reforço para a primeira linha (gestão operacional, incluindo desenvolvimento e segurança) ao automatizar a detecção de vulnerabilidades. Para a segunda linha (gestão de risco e conformidade), fornece dados e insights para a avaliação contínua do risco cibernético. A terceira linha (auditoria interna) beneficia de relatórios auditáveis e explicáveis gerados pelo *framework.*
- **Auditoria de Modelos:** Implementação de um processo de auditoria independente para os modelos de IA, cobrindo o ciclo de vida completo: desde a qualidade dos dados de treino, a validação dos algoritmos, a monitorização da performance em produção e a gestão de *drift* do modelo.
- ***Logs* Imutáveis:** Todas as decisões do modelo, os dados de entrada e os resultados são registados em *logs* imutáveis (potencialmente utilizando tecnologias de *blockchain* ou Merkle trees) para garantir a rastreabilidade e a integridade da auditoria.
- **Monitorização Contínua:** Implementação de um sistema de monitorização em tempo real da performance do modelo em produção, incluindo métricas como *Precision*, *Recall*, F1-Score e FPR. Testes estatísticos (e.g., Kolmogorov-Smirnov) são utilizados para detectar *data drift* ou *model drift*, que podem indicar uma degradação da performance ao longo do tempo.
- ***Stress testing* Algorítmico:** Realização de testes de stress nos modelos de IA para avaliar a sua resiliência em cenários extremos, como ataques adversariais ou a introdução de novos tipos de vulnerabilidades. Isto inclui a avaliação da sua capacidade de manter a performance sob condições de carga elevada ou com dados de baixa qualidade. A Tabela 4 apresenta os resultados dos *stress tests* realizados.

Tabela 4: Resultados dos *Stress tests* Algorítmicos

Cenário de Stress	Descrição	F1-Score Original	F1-Score Sob Stress	Degradação	Recuperação
Perturbação de Código (10% ruído)	Renomeação aleatória de 10% das variáveis e reordenação de linhas	0,90	0,87	-3,3%	Automática
Perturbação de Código (20% ruído)	Renomeação aleatória de 20% das variáveis e inserção de código morto	0,90	0,83	-7,8%	Requer retreino
Envenenamento de Dados (5%)	Inversão de rótulos em 5% dos dados de treino	0,90	0,85	-5,6%	Requer retreino
Envenenamento de Dados (10%)	Inversão de rótulos em 10% dos dados de treino	0,90	0,78	-13,3%	Requer retreino + auditoria
Carga Elevada (10x)	Aumento de 10x no volume de ficheiros processados simultaneamente	0,90	0,90	0%	N/A (latência +340%)
Novas Vulnerabilidades (zeroday simulado)	Introdução de 50 vulnerabilidades de tipos não vistos no treino	0,90	0,71	-21,1%	Requer finetuning
Código COBOL (fora do escopo)	Análise de 200 trechos de código COBOL	0,90	0,42	-53,3%	Fora do escopo actual

Fonte: Autor

Os resultados dos *stress tests* revelam que o modelo mantém robustez aceitável (degradação inferior a 10%) em cenários de perturbação moderada, mas apresenta degradação significativa face a envenenamento severo de dados e, especialmente, face a linguagens fora do escopo de treino (COBOL). Estes resultados fundamentam a necessidade de monitorização contínua e de mecanismos HITL para cenários de alta incerteza.

Plano de Contingência em Caso de Falha de Modelo: Desenvolvimento de planos de contingência claros para cenários em que o modelo de IA falha ou produz resultados erróneos. Estes planos incluem a intervenção humana, a reversão para métodos de detecção tradicionais e a revalidação urgente do modelo. Esses mecanismos podem contribuir para que a automação da segurança via IA não comprometa a governação e a conformidade, mas antes as fortaleça, fornecendo uma base sólida para a confiança regulatória.

5. MODELO MATEMÁTICO FORMAL E EQUAÇÕES DE AVALIAÇÃODE RISCO

5.1 Representação do Código como Grafo (GNN)

Para a detecção de vulnerabilidades, o código-fonte é transformado num Grafo de Atributos de Código (*Code Property Graph* - CPG), G = (V, E, X), onde:

- **V** = $\{v_1, v_2, \ldots, v_n\}$ é o conjunto de nós, representando elementos atómicos do código (e.g., instruções, variáveis, expressões, chamadas de função).
- **E** ⊆ V × V é o conjunto de arestas, que codificam as relações entre os nós. Estas arestas podem representar diferentes tipos de dependências, como o fluxo de controlo (*Control Flow Graph* - CFG), o fluxo de dados (*Data Flow Graph* - DFG) e a estrutura sintáctica (*Abstract Syntax Tree* - AST).
- **X** ∈ $\mathbb{R}^{nxd}$ é a matriz de atributos de nós, onde cada linha é um vetor de *embedding* de dimensão d para o nó v. Estes *embeddings* são gerados por um modelo de linguagem pré-treinado (e.g., CodeBERT), que captura o contexto semântico do código.

A atualização do estado de um nó h_v numa camada l de uma GNN (especificamente, o algoritmo *GraphSAGE*) é dada por uma função de agregação e atualização. A agregação recolhe informações dos vizinhos do nó, e a atualização combina essa informação com o estado anterior do nó. Formalmente, para cada nó v:

$$h_v^{(l)} = \sigma\left(W^{(l)} \cdot \left[h_v^{(l-1)} \,\|\, \mathrm{AGG}^{(l)}\left(\{h_u^{(l-1)} : u \in \mathcal{N}(v)\}\right)\right]\right)$$

Onde:

- $h_v^{(l)} \in \mathbb{R}^k$ → *embedding* do nó na camada l
- N(v) → vizinhos de v
- ‖ → concatenação
- $W^{(l)} \in \mathbb{R}^{k \times 2k}$ → matriz de pesos
- σ → função de ativação (ReLU)

Se a agregação for média:

$$\mathrm{AGG}^{(l)} = \frac{1}{|\,\mathcal{N}(v)\,|} \sum_{u \in \mathcal{N}(v)} h_u^{(l-1)}$$

5.2 Função de Perda e Optimização

A classificação final de um grafo (representando um trecho de código) como vulnerável ou não vulnerável é obtida aplicando uma camada de *readout* ao CPG, seguida por uma rede neural multi-camadas (MLP) e uma função de ativação *Sigmoid*:

$$\hat{y} = Sigmoid(MLP(Readout(G)))$$

Onde *Readout*(G) é uma função que agrega os *embeddings* de todos os nós do grafo para obter uma representação global do código. A função de perda utilizada para o treino do modelo é a *Binary Cross-Entropy*, modificada com uma penalização de erro para classes minoritárias (*cost-sensitive learning*) para lidar com o desbalanceamento inerente aos datasets de vulnerabilidades:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\left[w_{y_i}\cdot\left(y_i\log(\hat{y}_i)+(1-y_i)\log(1-\hat{y}_i)\right)\right]$$

Onde:

- N é o número total de amostras.
- $y_i \in \{0, 1\}$ é o rótulo verdadeiro.
- $\hat{y}_i \in [0, 1]$ é a probabilidade prevista.
- w_{y_i} é o peso da classe.

Peso típico: $w_c = \frac{N}{2N_c}$, onde N_c é o número de amostras da classe c. No presente estudo, com o dataset combinado de 45.230 amostras (8.940 vulneráveis, 36.290 não vulneráveis), os pesos foram: w_1 (vulnerável) = 2,53 e w_0 (não vulnerável) = 0,62.

5.3 Equações de Avaliação de Risco Operacional e Estabilidade Financeira

> ***Nota metodológica:*** *As equações apresentadas nesta subsecção constituem* ***propostas teóricas*** *para a quantificação do risco. A sua calibração empírica requer dados de incidentes reais e topologias de interconexão bancária que não estavam disponíveis para este estudo. Os parâmetros apresentados são ilustrativos e não devem ser utilizados directamente para decisões regulatórias sem validação adicional.*

Para quantificar o impacto das vulnerabilidades no contexto financeiro, propõe-se as seguintes equações:

5.3.1. Risco Sistémico Induzido por Vulnerabilidade (RSIV)

O risco sistémico R_s(v) de uma vulnerabilidade v num sistema financeiro legado L é uma função da probabilidade de exploração da vulnerabilidade, do impacto direto no sistema L, e da sua conectividade com outros sistemas financeiros na rede. Formalmente:

$$R_s(v) = P_e(v) \cdot I(L) \cdot C(L)$$

Onde:

- $P_e(v)$ → probabilidade de exploração, estimada com base no score CVSS e na exposição do sistema.
- I(L) → impacto financeiro esperado, calculado como a perda potencial em caso de exploração.
- $C(L) = \sum_{j=1}^{M} \alpha_j \cdot \omega_{Lj}$ → factor de conectividade, onde $\omega_{Lj} \in [0,1]$ é o peso de interconectividade entre o sistema L e o sistema j, e α_j é o factor de propagação sistémica.

5.3.2. Eficácia do Modelo na Redução do Risco Operacional (Gain Metric)

O ganho de eficiência ΔE na supervisão prudencial e na gestão do risco operacional, resultante da adopção do *framework* AI-VULN-FIN-LEGACY, pode ser quantificado pela redução no Tempo Médio de Detecção (MTTD) e pela capacidade de minimizar falsos positivos, que consomem recursos valiosos:

$$\Delta E = \left(\frac{MTTD_{trad} - MTTD_{IA}}{MTTD_{trad}} \right) \cdot (1 - FPR_{IA})$$

Onde:

- Se FPR → 1, o ganho tende a zero (muitos falsos positivos anulam o benefício da detecção rápida).
- Se $MTTD_{IA} \to 0$, o ganho tende ao máximo (detecção instantânea).

5.4 Matriz de Risco Operacional (Alinhada com Basileia III)

A gestão de risco operacional, conforme exigido por Basileia III, requer uma identificação, avaliação, monitorização e mitigação sistemática dos riscos. A seguinte matriz de risco operacional ilustra como as vulnerabilidades em sistemas legados, e a sua mitigação via IA, se enquadram neste *framework*. Os valores de impacto e probabilidade foram atribuídos com base numa combinação de:

(i) dados históricos de incidentes cibernéticos no sector financeiro, conforme reportados pelo FSB;

(ii) taxonomia de severidade do CVSS v3.1;

(iii) frequência de ocorrência observada nos datasets utilizados neste estudo; e

(iv) consulta à literatura especializada sobre risco operacional bancário.

Tabela 5: Matriz de Risco Operacional Alinhada com Basileia III

Categoria de Risco	**Descrição**	**Impacto (1-5)**	**Probabilidade (1-5)**	**Score (I×P)**	**Mitigação via AI-VULN-FIN-LEGACY**	**Justificação Empírica**
Dívida Técnica Acumulada	Complexidade e fragilidade do código legado que aumentam a probabilidade de erros e vulnerabilidades.	4	5	20	Mapeamento automatizado de *hotspots* de dívida técnica através de análise de grafos e métricas de complexidade de código.	78% das vulnerabilidades detectadas no estudo de caso correlacionaram-se com módulos de alta dívida técnica.
Exploração de Dia-Zero	Ataques que exploram vulnerabilidades desconhecidas publicamente.	5	2	10	Detecção anómala de padrões de código que se desviam do comportamento normal, mesmo para vulnerabilidades não catalogadas, via GNN e LLM.	O modelo detectou 71% das vulnerabilidades *zero-day* simuladas (Tabela 4).
Opacidade Algorítmica	Dificuldade em compreender as decisões dos modelos de IA.	3	3	9	Utilização de técnicas de Explicabilidade (XAI) como SHAP para justificar as previsões do modelo, tornando-o auditável.	92% das explicações SHAP foram validadas como coerentes por analistas seniores.

Categoria de Risco	Descrição	Impacto (1-5)	Probabilidade (1-5)	Score (I×P)	Mitigação via AI-VULN-FIN-LEGACY	Justificação Empírica
Data Leakage (Treino)	Exposição inadvertida de dados sensíveis durante o treino do modelo de IA.	4	2	8	Implementação de Divisão temporal rigoroso, anonimização de dados e ambientes de treino isolados para prevenir a fuga de informação.	Nenhum incidente de *data leakage* registado durante os experimentos.
Falha de Resiliência (MTTR)	Tempo prolongado para reparar vulnerabilidades após a sua detecção, aumentando a exposição ao risco.	5	3	15	Sugestão automática de remediação e localização precisa da falha, reduzindo o MTTR e melhorando a continuidade de negócio.	MTTR reduzido de 48h para 28,8h (redução de 40%).
Ataques Adversariais	Manipulação intencional de entradas para enganar o modelo	4	2	8	Testes de robustez contínuos e desenvolvimento de guardrails para	Degradação de apenas 3,3% sob perturbação de 10% (Tabela 4).
	de IA e contornar a detecção de vulnerabilidades.				modelos de linguagem de grande porte, monitorizando desvios de comportamento.	

Fonte: Autor

5.5 Proposta de *Framework* Auditável para Supervisão Prudencial

O *framework* AI-VULN-FIN-LEGACY é intrinsecamente desenhado para ser auditável, um requisito fundamental para a sua adopção em contextos regulatórios. A proposta de *framework* auditável assenta em quatro pilares:

- **Rastreabilidade Completa:** Todas as etapas do processo de detecção de vulnerabilidades, desde a ingestão do código-fonte até à geração do alerta final, são registadas em *logs* imutáveis. Isto inclui as versões dos modelos de IA utilizados, os dados de treino, os hiperparâmetros e as previsões. A utilização de tecnologias de *blockchain* ou Merkle trees pode garantir a integridade e a não-repudiação destes registos.

- **Explicabilidade Intrínseca:** Para cada vulnerabilidade detectada, o *framework* gera uma explicação detalhada utilizando técnicas de XAI (SHAP, LIME). Esta explicação não só aponta a localização da vulnerabilidade no código, mas também justifica a decisão do modelo, indicando quais características do código (e.g., padrões de API calls, fluxo de dados) contribuíram para a classificação. Esta funcionalidade é crucial para que auditores e reguladores possam compreender e validar as decisões do modelo.
- **Monitorização Contínua de *Drift*:** O *framework* inclui um módulo de monitorização em tempo real que compara as distribuições dos dados de entrada em produção com as dos dados de treino. Testes estatísticos (e.g., teste de Kolmogorov-Smirnov) são aplicados para detectar *data drift* ou *model drift*, que podem indicar uma degradação da performance do modelo ao longo do tempo. Alertas são gerados automaticamente para desencadear a revalidação ou o retreino do modelo.
- **Controlo Humano no Loop (*Human-in-the-Loop* - HITL):** Para vulnerabilidades classificadas com alta criticidade (e.g., CVSS > 7.0), o *framework* exige uma validação obrigatória por analistas seniores de segurança. Este mecanismo HITL pode contribuir para que o julgamento humano prevaleça em decisões de alto impacto, mitigando os riscos de falsos positivos ou de falhas catastróficas do modelo. O *feedback* humano é então reintroduzido no sistema para melhorar continuamente o modelo.

Este *framework* auditável visa aumentar a confiança na IA e fornecer aos reguladores ferramentas para uma supervisão baseada em evidências da segurança cibernética nas instituições financeiras. A sua eficácia prática depende, contudo, de validação em ambientes reais.

6. RESULTADOS QUANTITATIVOS E IMPACTO OPERACIONAL

> ***Nota sobre o escopo dos resultados:*** *Todos os resultados quantitativos apresentados nesta secção foram obtidos em* ***ambiente experimental simulado****, utilizando datasets públicos (Juliet Test Suite, BigVul) complementados por um dataset sintético de código bancário legado. Os resultados não constituem medições em ambientes de produção reais e a sua generalização para sistemas bancários operacionais requer validação piloto adicional.*

6.1 Tabela Comparativa de Performance

Os experimentos controlados, realizados em ambiente simulado, sugerem uma superioridade estatisticamente significativa do *framework* AI-VULN-FIN-LEGACY em relação às abordagens tradicionais e aos modelos de IA isolados. A Tabela 6 resume as métricas de performance obtidas no conjunto de teste (*out-of-sample*), após o *fine-tuning* dos modelos com o dataset do estudo de caso.

Tabela 6: Comparação de Métricas de Performance na Detecção de Vulnerabilidades

Modelo	*Precision*	*Recall*	F1-Score	ROC-AUC	FPR	IC 95% (F1)
SonarQube (*Baseline* SAST)	0,45	0,39	0,42	0,58	0,15	[0,38 - 0,46]
Random Forest	0,68	0,61	0,64	0,75	0,10	[0,60 - 0,68]
GNN (*GraphSAGE*)	0,85	0,81	0,83	0,91	0,04	[0,80 - 0,86]
CodeBERT	0,89	0,84	0,86	0,94	0,03	[0,83 - 0,89]
AI-VULN-FIN-LEGACY (Híbrido)	**0,93**	**0,88**	**0,90**	**0,97**	**0,02**	**[0,88 - 0,92]**

Fonte: Autor

Análise por Subgrupos de Vulnerabilidades

> ***Nota sobre a interpretação dos resultados:*** *Todos os resultados apresentados nesta secção foram obtidos em* ***ambiente simulado*** *com datasets públicos e sintéticos (Juliet Test Suite, Big-Vul) complementados por um dataset simulado de código bancário. A performance em ambientes de produção reais pode diferir significativamente devido a factores como complexidade do código proprietário, interdependências de módulos, e padrões de codificação idiossincráticos.*

Para além da análise agregada, foi realizada uma análise por subgrupos de vulnerabilidades, segmentada por tipo CWE, severidade CVSS e linguagem de programação. A Tabela 7 apresenta os resultados do *framework* híbrido por categoria CWE.

Tabela 7: Performance do *Framework* Híbrido por Categoria CWE

Categoria CWE	Descrição	Nº Amostras (Teste)	*Precision*	*Recall*	F1-Score	Observações
CWE-89	SQL Injection	342	0,96	0,94	0,95	Excelente detecção; padrões claros no CPG
CWE-79	Cross-Site Scripting (XSS)	287	0,91	0,87	0,89	Boa performance; algumas variantes evasivas
CWE-78	OS Command Injection	156	0,94	0,91	0,92	Forte detecção via análise de fluxo de dados
CWE-22	Path Traversal	198	0,90	0,85	0,87	Performance sólida
CWE-287	Improper Authentication	124	0,88	0,79	0,83	Dependente de contexto semântico (LLM)
CWE-862	Missing Authorization	98	0,85	0,76	0,80	Mais difícil; requer compreensão de lógica de negócio

Categoria CWE	Descrição	Nº Amostras (Teste)	*Precision*	*Recall*	F1-Score	Observações
CWE-502	Deserialization of Untrusted Data	167	0,92	0,88	0,90	Padrões bem capturados pelo GNN
CWE-798	Hard-coded Credentials	89	0,97	0,95	0,96	Detecção quase perfeita via LLM
CWE-190	Integer Overflow	112	0,83	0,74	0,78	Mais desafiante em Java legado
CWE-400	Uncontrolled Resource Consumption	73	0,80	0,71	0,75	Menor performance; padrões subtis

Fonte: Autor

Tabela 8: Performance do *Framework* Híbrido por Severidade CVSS

Severidade CVSS	Score CVSS	Nº Amostras (Teste)	*Precision*	*Recall*	F1-Score
Crítica	9.0 - 10.0	187	0,95	0,92	0,93
Alta	7.0 - 8.9	423	0,94	0,89	0,91
Média	4.0 - 6.9	612	0,92	0,87	0,89
Baixa	0.1 - 3.9	424	0,88	0,82	0,85

Fonte: Autor

Os resultados por subgrupo revelam que o *framework* apresenta melhor performance na detecção de vulnerabilidades de alta severidade (F1 = 0,93 para CVSS Crítica) comparativamente às de baixa severidade (F1 = 0,85), o que é um comportamento desejável do ponto de vista da gestão de risco, pois prioriza a detecção das ameaças mais graves. Por linguagem, a performance em Java 8 (F1 = 0,91) foi ligeiramente superior à de PL/SQL (F1 = 0,87), reflectindo a maior disponibilidade de dados de treino para Java.

6.2 Interpretação Estatística e Robustez

Através da aplicação do Teste de McNemar, comparando o desempenho do *framework* AI-VULN-FIN-LEGACY com o *baseline* SAST (SonarQube), obteve-se um p-valor de $2{,}3 \times 10^{-5}$. Este resultado permite rejeitar a hipótese nula (H_0: Modelos de IA não apresentam ganho estatisticamente significativo face a métodos tradicionais) com um nível de confiança extremamente elevado ($p <$

0,001). Consequentemente, a hipótese alternativa (H_1: Modelos de IA apresentam melhoria estatisticamente significativa com controle de robustez e viés) é suportada pelos dados.

O intervalo de confiança de 95% para o F1-Score do modelo híbrido situou-se entre [0,88 - 0,92], indicando uma alta consistência na performance. A robustez do modelo foi avaliada através de ablation studies e testes de sensibilidade, onde o modelo manteve um F1-Score acima de 0,85 mesmo com a introdução de 10% de ruído estrutural nos CPGs, o que sugere uma capacidade razoável de generalização e resiliência a pequenas perturbações nos dados de entrada, embora esta robustez necessite de confirmação em ambientes de produção reais.

6.2.1 Análise Detalhada de Falsos Positivos e Falsos Negativos

A análise dos casos de falha do modelo é essencial para compreender os seus limites e orientar melhorias futuras. A Tabela 9 apresenta uma análise qualitativa dos principais padrões de falsos positivos e falsos negativos observados.

Tabela 9: Análise de Padrões de Falsos Positivos e Falsos Negativos

Tipo de Erro	Padrão Identificado	Frequência	Causa Provável	Impacto Operacional	Mitigação Proposta
Falso Positivo	Código defensivo confundido com vulnerabilidade	34% dos FP	Padrões de sanitização complexos não reconhecidos pelo modelo	Custo de triagem desnecessária ($_{2h/alerta)}$	*Fine-tuning* com exemplos de código defensivo
Falso Positivo	Código morto ou desactivado sinalizado	28% dos FP	O modelo não distingue código activo de inactivo	Investigação de código irrelevante	Integração com análise de cobertura de código
Falso Positivo	Padrões legados seguros classificados como inseguros	22% dos FP	Construções idiomáticas de Java 8 não presentes no treino moderno	Fadiga de alerta nas equipas	Expansão do dataset de treino com padrões legados seguros
Falso Positivo	Dependências externas seguras sinalizadas	16% dos FP	Análise limitada de bibliotecas de terceiros	Falsos alertas em integrações	Integração com bases de dados de vulnerabilidades de dependências
Falso Negativo	Vulnerabilidades de lógica de negócio	38% dos FN	Requerem compreensão do contexto de negócio além do código	Vulnerabilidades críticas não detectadas	Enriquecimento com regras de negócio específicas

Tipo de Erro	Padrão Identificado	Frequência	Causa Provável	Impacto Operacional	Mitigação Proposta
Falso Negativo	Vulnerabilidades em código PL/SQL complexo	25% dos FN	Menor representação de PL/SQL nos dados de treino	Exposição em camada de dados	Expansão de dataset PL/SQL
Falso Negativo	Vulnerabilidades multi-módulo (crosscutting)	22% dos FN	Limitação na profundidade de análise intermódulos	Falhas em integrações não detectadas	Aumento da profundidade do CPG
Falso Negativo	Vulnerabilidades temporais (race conditions)	15% dos FN	Análise estática não captura comportamento dinâmico	Condições de corrida não detectadas	Complementação com análise dinâmica (DAST)

Fonte: Autor

Esta análise revela que os falsos positivos são predominantemente causados por padrões de código defensivo ou legado que o modelo interpreta erroneamente como vulneráveis, enquanto os falsos negativos concentram-se em vulnerabilidades que requerem compreensão de lógica de negócio ou análise dinâmica áreas onde a intervenção humana (HITL) permanece indispensável.

6.3 Impacto Operacional Estimado em Continuidade de Negócio e Três Linhas deDefesa

> ***Nota sobre pressupostos:*** *Os valores de baseline (MTTD = 72h, MTTR = 48h) são estimativas baseadas em literatura e pressupostos operacionais típicos para equipas de segurança com ferramentas SAST tradicionais, não medições em bancos reais. As reduções percentuais reportadas devem ser interpretadas como indicativas e condicionadas por estes pressupostos.*

No ambiente experimental simulado, considerando os pressupostos definidos para a linha de base SAST/manual, a implementação do *framework* AI-VULN-FIN-LEGACY apresentou uma melhoria estimada dos indicadores operacionais de segurança:

- **Redução de 55% no Tempo Médio de Detecção (MTTD):** O MTTD *baseline* estimado, com base em pressupostos de ferramentas SAST tradicionais e revisão manual, foi de 72 horas (3 dias úteis). Com a implementação do *framework* AI-VULN-FIN-LEGACY, o MTTD foi reduzido para 32,4 horas, representando uma redução absoluta de 39,6 horas. Esta redução deve-se à capacidade de automatizar a análise de código e identificar vulnerabilidades em minutos de processamento computacional, em contraste com os dias exigidos por revisões

manuais. A redução do MTTD minimiza a janela de exposição a ameaças e permite uma resposta mais ágil.

- **Redução de 40% no Tempo Médio de Resposta (MTTR):** O MTTR *baseline* estimado foi de 48 horas (2 dias úteis). Com o *framework*, o MTTR foi reduzido para 28,8 horas, uma redução absoluta de 19,2 horas. Esta melhoria deve-se à explicabilidade fornecida pelo módulo SHAP, que aponta a causa raiz da vulnerabilidade e indica localizações exatas no código, acelerando o processo de remediação. A precisão na identificação da falha reduz o tempo gasto na depuração e na implementação de correcções.

Tabela 10: Valores Absolutos de MTTD e MTTR

Métrica	*Baseline* (SAST + Manual)	Com AI-VULN-FIN-LEGACY	Redução Absoluta	Redução Percentual
MTTD	72 horas	32,4 horas	39,6 horas	55%
MTTR	48 horas	28,8 horas	19,2 horas	40%
Custo médio por vulnerabilidade triada	$850	$340	$510	60%
Vulnerabilidades detectadas por ciclo	12	31	+19	+158%

Fonte: Autor

No contexto do Modelo de Três Linhas de Defesa (First Line: Gestão Operacional; Second Line: Gestão de Risco e Conformidade; Third Line: Auditoria Interna), o *framework* AI-VULN-FIN-LEGACY fortalece todas as linhas:

- **Primeira Linha:** Capacita as equipas de desenvolvimento e segurança com uma ferramenta proactiva para detectar e corrigir vulnerabilidades antes que estas cheguem à produção, reduzindo a dívida técnica e melhorando a qualidade do código.
- **Segunda Linha:** Fornece à gestão de risco e conformidade uma visão agregada e em tempo real da postura de segurança cibernética da instituição, permitindo uma avaliação mais precisa do risco operacional e a implementação de controlos eficazes.
- **Terceira Linha:** Oferece aos auditores internos relatórios detalhados e explicáveis, facilitando a verificação da eficácia dos controlos de segurança e a conformidade com as políticas internas e regulamentações externas.

7. ANÁLISE CRÍTICA E RISCOS EMERGENTES

Embora os resultados experimentais do *framework* AI-VULN-FIN-LEGACY em ambiente simulado sejam promissores, é imperativo abordar criticamente os riscos e desafios inerentes à adopção de IA em contextos tão sensíveis como o sector financeiro. A discussão que se segue identifica riscos que podem condicionar ou impedir a operacionalização do *framework* em ambientes reais.

7.1 Riscos de *Overfitting* e Viés Algorítmico

Um dos maiores desafios na aplicação de modelos de *Deep Learning* à cibersegurança é o risco de *overfitting*, onde o modelo aprende os padrões específicos do dataset de treino em detrimento da sua capacidade de generalizar para dados não vistos. Em sistemas legados, onde os padrões de código e vulnerabilidade podem ser idiossincráticos, o *fine-tuning* inadequado pode levar a um modelo que performa bem em dados históricos, mas falha em detectar novas vulnerabilidades ou variações. A mitigação deste risco foi abordada através de *early stopping*, regularização e, crucialmente, a utilização de Divisão temporal para o conjunto de teste.

O viés algorítmico é outra preocupação significativa. Se os dados de treino forem desbalanceados ou representarem apenas um subconjunto limitado de padrões de vulnerabilidade (e.g., focando em vulnerabilidades comuns e ignorando as raras, mas de alto impacto), o modelo pode perpetuar ou amplificar este viés. Isto poderia levar a uma detecção ineficaz de certas classes de vulnerabilidades ou a uma discriminação inadvertida contra certos estilos de codificação ou módulos, com implicações para a equidade e a segurança abrangente do sistema [17].

7.2 Opacidade e Ataques Adversariais

A natureza de "caixa-preta" de muitos modelos de *Deep Learning*, especialmente os LLMs, representa uma barreira para a sua aceitação regulatória e para a confiança dos utilizadores. Embora técnicas de Explicabilidade da IA (XAI) como SHAP e LIME ajudem a mitigar a opacidade, elas não a eliminam completamente. A incapacidade de compreender totalmente o raciocínio do modelo pode dificultar a auditoria, a validação e a atribuição de responsabilidade em caso de falha [16].

[16] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
[17] C. O'Neil, *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. New York: Crown, 2016.

Outro risco emergente são os ataques adversariais. Um atacante malicioso pode intencionalmente manipular as entradas do modelo (e.g., introduzir pequenas alterações no código-fonte que são imperceptíveis para humanos, mas que enganam o modelo de IA) para contornar a detecção de vulnerabilidades. Esta técnica, conhecida como *adversarial example generation* ou *model poisoning*, representa uma ameaça séria à integridade dos sistemas de segurança baseados em IA e exige o desenvolvimento contínuo de contramedidas robustas [18].

7.2.1 Dependência Excessiva de Automação

A "complacência com a automação" é um risco comportamental. A confiança excessiva nos resultados gerados pela IA pode levar as equipas de segurança a reduzir a revisão humana e o pensamento crítico. Este artigo defende que a IA deve ser vista como um aumentador de capacidades (*augmenter*), e não como um substituto do julgamento humano. O *Human-in-the-Loop* (HITL) é essencial para validar alertas de alta criticidade, interpretar resultados ambíguos e fornecer *feedback* contínuo para o aprimoramento do modelo. A delegação total da responsabilidade de segurança à IA pode levar a uma diminuição da vigilância e a uma perda de competências críticas nas equipas de segurança.

7.3 Riscos de Privacidade e Protecção do Código-Fonte

A utilização de modelos de IA para análise de código-fonte de sistemas financeiros legados levanta preocupações significativas de privacidade e protecção de propriedade intelectual. O código-fonte de um *core banking System* contém não apenas a lógica de processamento de transacções, mas também regras de negócio proprietárias, algoritmos de cálculo de taxas e juros, e mecanismos de controlo interno que constituem segredos comerciais da instituição financeira. A exposição inadvertida deste código durante o processo de treino ou inferência do modelo de IA pode resultar em:

(i) fuga de lógica de negócio proprietária;
(ii) revelação de mecanismos de segurança internos que poderiam ser explorados por atacantes;e
(iii) violação de obrigações contratuais de confidencialidade com fornecedores de software.

[18] N. Papernot et al., "The Limitations of *Deep Learning* in Adversarial Settings," in *Proceedings of the IEEE European Symposium on Security and Privacy*, pp. 372–387, 2016. doi: 10.1109/EuroSP.2016.36.

Para mitigar estes riscos, o *framework* AI-VULN-FIN-LEGACY incorpora as seguintes salvaguardas:

- **Processamento *On-premise*:** Todo o processamento de código-fonte é realizado em infra-estrutura controlada pela instituição financeira, sem transmissão de dados para serviços *cloud* externos. Os modelos de IA são implantados localmente após o treino.
- **Anonimização de Código:** Antes do processamento, o código é submetido a um processo de anonimização que substitui nomes de variáveis, funções e módulos por identificadores genéricos, preservando a estrutura e o fluxo de dados mas removendo informação de negócio.
- **Segregação de Ambientes:** Os ambientes de treino, validação e produção são rigorosamente segregados, com controlos de acesso baseados em funções (RBAC) e monitorização de acessos.

7.4 Riscos de Data Residency e Transmissão para LLM

A questão da residência de dados (*data residency*) é particularmente relevante no contexto de economias emergentes como Moçambique, onde a legislação de protecção de dados pode exigir que dados sensíveis sejam armazenados e processados dentro do território nacional. A utilização de LLMs baseados em *cloud* (e.g., APIs de modelos comerciais) para análise de código-fonte bancário levanta riscos significativos:

- **Transmissão Transfronteiriça:** O envio de código-fonte para APIs de LLM hospedadas em jurisdições estrangeiras pode violar requisitos de data residency e expor a instituição a riscos regulatórios.
- **Retenção de Dados por Terceiros:** Fornecedores de LLM podem reter dados de entrada para fins de melhoria dos seus modelos, criando um risco de exposição inadvertida.
- **Dependência de Conectividade:** A utilização de APIs *cloud* introduz uma dependência de conectividade à internet, que pode ser problemática em contextos de infra-estrutura limitada.

Para mitigar estes riscos, o *framework* propõe a utilização exclusiva de modelos *on-premise* (e.g., CodeBERT implantado localmente via HuggingFace Transformers) ou, quando necessário, a utilização de nuvens soberanas infra-estruturas *cloud* operadas dentro do território nacional e sujeitas à jurisdição local. No contexto moçambicano, esta abordagem alinha-se com as directrizes do BdM sobre gestão de risco tecnológico e protecção de dados.

7.5 Dependência Tecnológica e Vendor Lock-in

A adopção de *framework*s de IA baseados em tecnologias proprietárias pode criar uma nova forma de dependência tecnológica, substituindo a dependência de sistemas legados por uma dependência de fornecedores de IA. Este risco de *vendor lock-in* manifesta-se em:

- **Dependência de Fornecedores de GPU:** A necessidade de hardware especializado (e.g., NVIDIA A100) cria uma dependência de um número limitado de fabricantes.
- **Dependência de *Framework*s de Software:** A utilização de *framework*s proprietários ou com licenciamento restritivo pode limitar a flexibilidade da instituição.
- **Dependência de Modelos Pré-treinados:** A utilização de modelos pré-treinados por terceiros (e.g., CodeBERT da Microsoft) introduz uma dependência do fornecedor para actualizações e suporte.

Para mitigar o risco de vendor lock-in, o *framework* AI-VULN-FIN-LEGACY foi deliberadamente arquitectado com base em tecnologias abertas e de código aberto: PyTorch (licença BSD), HuggingFace Transformers (licença Apache 2.0), PyTorch Geometric (licença MIT) e Joern (licença Apache 2.0). Esta escolha pode contribuir para que a instituição mantém o controlo total sobre o código, os modelos e os dados, podendo migrar entre fornecedores de hardware ou *cloud* sem custos de transição proibitivos. Adicionalmente, o *framework* suporta a exportação de modelos em formato ONNX, permitindo a sua execução em hardware de diferentes fabricantes.

7.6 *Accountability* e Responsabilização em Caso de Falha

A questão da responsabilização (*accountability*) em caso de falha do modelo de IA é um tema crítico e ainda em evolução no panorama regulatório. Quando o *framework* AI-VULN-FIN-LEGACY falha em detectar uma vulnerabilidade que é subsequentemente explorada, ou quando um falso positivo leva a decisões operacionais prejudiciais, a cadeia de responsabilidade deve ser claramente definida.

O presente *framework* propõe o seguinte modelo de *accountability*:

Tabela 11: Modelo de Accountability em Caso de Falha

Cenário de Falha	Responsável Primário	Responsável Secundário	Mecanismo de Mitigação
Falso Negativo (vulnerabilidade não detectada)	CISO / Direcção de Segurança	Conselho de Administração (se governação inadequada)	HITL obrigatório para sistemas críticos; auditoria periódica do modelo

Cenário de Falha	Responsável Primário	Responsável Secundário	Mecanismo de Mitigação
Falso Positivo (alerta incorreto com impacto operacional)	Equipa de Operações de Segurança	Equipa de Data Science (se modelo mal calibrado)	Threshold de confiança ajustável; processo de triagem em duas fases
Data leakage durante treino	Chief Data Officer (CDO)	Fornecedor de infraestrutura (se falha de segregação)	Ambientes isolados; anonimização obrigatória; auditorias de acesso
Degradação de performance (*model drift*)	Equipa de MLOps	CISO (se monitorização insuficiente)	Monitorização contínua com alertas automáticos; retreino programado
Ataque adversarial bem sucedido	Equipa de Segurança Ofensiva	Equipa de Data Science (se testes insuficientes)	Red teaming periódico; *stress tests* trimestrais

Fonte: Autor

É fundamental que a responsabilidade última pela segurança do sistema financeiro permaneça no Conselho de Administração da instituição, conforme exigido pelos princípios de governação corporativa de Basileia III. A IA é uma ferramenta que auxilia a tomada de decisão, mas não substitui a responsabilidade fiduciária dos gestores. O *framework* exige que o Conselho de Administração aprove formalmente a política de utilização de IA em funções de segurança, incluindo os limiares de risco aceitáveis, os mecanismos de escalação e os planos de contingência.

8. IMPLICAÇÕES PARA SUPERVISÃO PRUDENCIAL (*Supervisory Technology*)

> ***Nota metodológica:*** *Esta secção constitui uma* ***proposta conceptual*** *baseada na extrapolação dos resultados experimentais obtidos em ambiente simulado. As implicações aqui discutidas não foram validadas empiricamente junto de reguladores ou instituições financeiras reais, e devem ser interpretadas como hipóteses de trabalho para investigação futura e discussão política.*

A eventual adopção do *framework* AI-VULN-FIN-LEGACY poderia ter implicações para a forma como os Bancos Centrais e as autoridades de supervisão financeira exercem as suas funções, potencialmente contribuindo para a transição do paradigma da supervisão de reactivo para proactivo e baseado em dados.

8.1 Modelo de Auditoria Contínua e Monitorização de Risco Sistémico

O *framework* apresenta potencial para apoiar a transição de um modelo de auditoria pontual e baseado em inspeções para um modelo de auditoria contínua. Em vez de depender de relatórios periódicos ou de inspeções in-loco, o regulador pode exigir o acesso a dashboards de risco alimentados pelo AI-VULN-FINLEGACY, que fornecem uma visão em tempo real da postura de segurança cibernética das instituições financeiras. Esta capacidade permite uma intervenção precoce em caso de degradação da resiliência operacional ou da emergência de novas vulnerabilidades sistémicas.

Além disso, a capacidade de mapear e quantificar o risco sistémico induzido por vulnerabilidades (RSIV), conforme o modelo matemático proposto na Secção 5.3.1, permite que os reguladores identifiquem e priorizem as vulnerabilidades que representam a maior ameaça à estabilidade financeira global. Isto é particularmente relevante para a gestão de riscos em infra-estruturas de mercado financeiro (FMIs) e para a coordenação entre diferentes autoridades reguladoras.

8.2 Requisitos Mínimos de Governação para IA na Supervisão

Para garantir a adopção segura e eficaz da IA na segurança financeira, propõe-se que as autoridades de supervisão estabeleçam os seguintes requisitos mínimos de governação:

- **Certificação e Validação Independente de Modelos:** Os modelos de IA utilizados para detecção de vulnerabilidades em sistemas críticos devem ser submetidos a um processo rigoroso de certificação e validação independente. Este processo deve abranger a qualidade dos dados de treino, a robustez dos algoritmos, a mitigação de viés e a performance em cenários de stress. O NIST AI Risk Management *Framework* [19] pode servir como base para a criação de um padrão de certificação específico para o sector financeiro.
- **Explicabilidade Obrigatória e Transparência:** Nenhum alerta de segurança gerado por IA pode ser apresentado sem a correspondente justificação técnica e explicabilidade. As instituições financeiras devem ser capazes de demonstrar como o modelo chegou a uma determinada conclusão, utilizando técnicas de XAI. Esta transparência é crucial para a confiança regulatória e para a atribuição de responsabilidade.

[19] National Institute of Standards and Technology, *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*, NIST AI 100-1. Gaithersburg, MD: NIST, 2023. doi: 10.6028/NIST.AI.100-1.

- **Resiliência a Ataques Adversariais:** Os modelos de IA devem ser testados periodicamente contra ataques adversariais e técnicas de evasão. As instituições devem demonstrar que os seus sistemas de segurança baseados em IA são robustos e capazes de resistir a tentativas de manipulação maliciosa.
- **Gestão de Dados e Privacidade:** Dada a sensibilidade dos dados de código-fonte, especialmente em sistemas legados, as instituições devem implementar controlos rigorosos de privacidade e segurança de dados, incluindo anonimização, segregação de ambientes e conformidade com regulamentações de protecção de dados (e.g., GDPR, legislação moçambicana de protecção de dados).

8.3 Aplicação em *Supervisory Technology* e Alinhamento com Basileia III

O *framework* AI-VULN-FIN-LEGACY propõe-se como uma ferramenta de apoio para o desenvolvimento de capacidades de SupTech. Permite que os reguladores:

- **Monitorizem a conformidade:** Verifiquem a conformidade das instituições com os requisitos de segurança cibernética e de gestão de risco operacional de forma automatizada e contínua.
- **Identifiquem riscos emergentes:** Detectem padrões de vulnerabilidade que podem indicar tendências de risco emergentes no sector, permitindo uma resposta regulatória proactiva.
- **Avaliem a resiliência:** Obtenham uma avaliação quantitativa da resiliência cibernética das instituições, informando as decisões sobre requisitos de capital e planos de recuperação.

Em alinhamento com Basileia III, que exige que os bancos mantenham capital para cobrir riscos operacionais, o *framework* fornece uma metodologia para quantificar e mitigar um componente significativo desse risco. Ao reduzir a probabilidade e o impacto de falhas cibernéticas, a IA pode contribuir para uma alocação de capital mais eficiente e para a melhoria da estabilidade financeira global [20].

[20] Basel *Committee* on Banking Supervision, *Basel III: Finalising Post-Crisis Reforms*. Basel: Bank for International Settlements, 2017. Available: https://www.bis.org/bcbs/publ/d424.htm

8.4 Integração com Processos de Supervisão em Moçambique (Proposta Conceptual)

A título de proposta conceptual não validada empiricamente junto do regulador a eventual integração do *framework* AI-VULN-FIN-LEGACY com os processos de supervisão do Banco de Moçambique (BdM) poderia ser explorada através de:

- **Integração com o Sistema de Reporte de Incidentes Cibernéticos:** O BdM implementou um sistema de reporte obrigatório de incidentes cibernéticos pelas instituições financeiras supervisionadas. O *framework* pode complementar este sistema, fornecendo dados quantitativos sobre a postura de segurança das instituições e permitindo ao regulador correlacionar incidentes reportados com vulnerabilidades detectadas.
- **Alinhamento com o Quadro Regulatório Nacional:** O *framework* pode ser adaptado para incorporar os requisitos específicos do quadro regulatório moçambicano de gestão de risco tecnológico, incluindo os avisos e circulares emitidos pelo BdM sobre segurança de sistemas de informação.
- **Capacitação do Regulador:** A implementação do *framework* como ferramenta de SupTech requer a capacitação das equipas de supervisão do BdM em técnicas de IA e análise de dados. Propõe-se um programa de formação faseado que inclua:
    - (i) sensibilização sobre IA e cibersegurança para gestores de supervisão;
    - (ii) formação técnica em interpretação de resultados do *framework* para inspectores; e
    - (iii) desenvolvimento de competências em data science para a equipa de SupTech do BdM.
- **Cooperação Regional:** O *framework* pode servir como base para a cooperação regional em matéria de supervisão cibernética, nomeadamente no âmbito da Comunidade de Desenvolvimento da África Austral (SADC), onde vários países enfrentam desafios semelhantes de modernização de infra-estruturas financeiras legadas.

9. LIMITAÇÕES DO ESTUDO

Este estudo, embora abrangente, apresenta algumas limitações inerentes que devem ser reconhecidas:

- **Disponibilidade de Dados Legados:** A principal limitação reside na dificuldade de acesso a datasets públicos representativos de sistemas financeiros legados. A sensibilidade e o carácter proprietário do código bancário real limitaram a validação a um ambiente simulado e a datasets públicos, que, embora relevantes, podem não capturar todas as idiossincrasias de um sistema em produção. A validade externa dos resultados é, portanto, condicionada pela fidelidade do ambiente simulado em relação a sistemas reais. É crucial distinguir entre a validade interna (capacidade de o experimento demonstrar efeito sob condições controladas) e a validade externa (capacidade de generalizar para bancos reais em Moçambique ou noutras economias emergentes). Os datasets públicos como Juliet e Big-Vul não representam integralmente sistemas bancários legados reais, que possuem dependências proprietárias, regras de negócio internas, restrições legais e padrões de desenvolvimento específicos. Os resultados devem ser interpretados como evidência preliminar, e a adopção prática requer validação piloto em instituições financeiras reais com dados proprietários anonimizados. Recomenda-se que instituições financeiras que adoptem o *framework* realizem uma fase de validação interna com o seu próprio código antes da operacionalização completa.
- **Ausência de COBOL:** A exclusão de COBOL constitui uma limitação significativa deste estudo, dado que esta linguagem permanece amplamente utilizada em sistemas financeiros legados a nível global, particularmente em mainframes que processam transacções de alto volume. A complexidade de gerar CPGs e *embeddings* de alta qualidade para COBOL com as ferramentas actuais foi o factor determinante para esta exclusão. As ferramentas de parsing disponíveis (e.g., Joern, Tree-sitter) apresentam suporte limitado ou inexistente para COBOL, e os modelos de linguagem pré-treinados (e.g., CodeBERT) não incluem COBOL nos seus corpora de treino. Esta limitação é particularmente relevante para o contexto moçambicano, onde alguns sistemas legados mais antigos podem ainda utilizar COBOL. O desenvolvimento de *parsers* COBOL-to-CPG e o finetuning de LLMs em código COBOL constituem direcções prioritárias para investigação futura.
- **Generalização de Modelos:** Embora o *fine-tuning* e o Divisão temporal tenham sido utilizados para melhorar a generalização, a performance do modelo num ambiente de produção real pode variar e exigir calibrações adicionais.
- **Custo Computacional:** O treino e a inferência de modelos de *Deep Learning* (GNN e LLM) são computacionalmente intensivos, exigindo infra-estruturas de hardware significativas (conforme detalhado na Tabela 3). Este custo pode ser uma barreira para instituições

financeiras com recursos limitados, particularmente em economias emergentes. A Tabela 12 apresenta uma estimativa de custos de implementação para diferentes perfis de instituição.

Tabela 12: Estimativa de Custos de Implementação por Perfil de Instituição

Componente de Custo	Banco Grande (Tier 1)	Banco Médio (Tier 2)	Banco Pequeno / Microfinança
Hardware (*on-premise*)	250.000−400.000	150.000−250.000	50.000−100.000 (*cloud*)
Licenciamento de Software	$0 (open-source)	$0 (open-source)	$0 (open-source)
Equipa de Data Science (anual)	180.000−300.000 (3-5 pessoas)	90.000−150.000 (2-3 pessoas)	45.000−75.000 (1-2 pessoas)
Formação e Capacitação	30.000−50.000	15.000−30.000	10.000−15.000
Consultoria de Implementação	80.000−150.000	40.000−80.000	20.000−40.000
Custos Operacionais Anuais	60.000−100.000	30.000−60.000	15.000−30.000
Investimento Total (Ano 1)	600.000−**1.000.000**	325.000−**570.000**	140.000−**260.000**
ROI Estimado (3 anos)	**180% - 250%**	**150% - 200%**	**120% - 170%**

Fonte: Autor

Nota metodológica: Os valores apresentados na Tabela 12 são estimativas aproximadas baseadas em benchmarks de mercado e literatura especializada. Dependem do contexto institucional, fornecedores, localização geográfica, infraestrutura existente e escala da implementação. Trata-se de uma estimativa operacional indicativa, não de uma medição em ambiente bancário real. O ROI estimado baseia-se em pressupostos de redução de custos de remediação e não foi validado empiricamente.

O ROI estimado baseia-se na redução de custos de remediação de vulnerabilidades (60% de redução, conforme Tabela 10), na redução de perdas por incidentes cibernéticos e na melhoria da eficiência das equipas de segurança. Para bancos pequenos e microfinanças em economias emergentes, o modelo de *cloud* computing (*pay-as-you-go*) pode reduzir significativamente a barreira de entrada.

Evolução das Ameaças: O panorama das ameaças cibernéticas está em constante evolução. Embora o *framework* seja projectado para ser adaptável, a sua eficácia a longo prazo dependerá da capacidade de ser continuamente actualizado e retreinado com novos dados de vulnerabilidades.

10. *ROADMAP* DE IMPLEMENTAÇÃO EM BANCOS

> ***Nota metodológica:*** *O roadmap apresentado nesta secção constitui uma proposta conceptual baseada em boas práticas de implementação de projectos de IA e cibersegurança. Não foi validado empiricamente em nenhuma instituição financeira real e deve ser adaptado ao contexto específico de cada organização.*

Para instituições financeiras que considerem adoptar o *framework* AI-VULN-FIN-LEGACY, propõe-se o seguinte *roadmap* de implementação faseado:

Fase 1: Avaliação e Preparação (Meses 1-3)

- **Inventário de Ativos Legados:** Identificação e catalogação de todos os sistemas legados críticos, suas linguagens, dependências e interfaces.
- **Higienização de Dados Históricos:** Coleta, anonimização e estruturação de dados históricos de vulnerabilidades, incidentes de segurança e *commits* de código para o treino inicial do modelo.
- **Definição de Métricas e KPIs:** Estabelecimento de métricas de sucesso (e.g., redução de MTTD, MTTR, FPR) e Key Performance Indicators (KPIs) para monitorizar a implementação.
- **Avaliação de Custos e Recursos:** Estimativa detalhada dos custos de implementação (conforme Tabela 12) e alocação de recursos humanos e computacionais.

Fase 2: Implementação Piloto e Validação (Meses 4-8)

- **Implementação em Modo "Shadow":** O *framework* é implementado em paralelo com as ferramentas de segurança existentes, sem impactar as operações. Os resultados da IA são comparados com os das ferramentas atuais para calibração e validação interna.
- **Treino *e Fine-tuning* Inicial:** O modelo de IA é treinado com os dados históricos e *fine-tuned* com dados específicos do ambiente do banco.
- **Formação de Equipas:** Capacitação das equipas de segurança, desenvolvimento e risco na utilização do *framework* e na interpretação dos seus resultados.
- **Validação com Regulador:** Apresentação dos resultados preliminares ao regulador (e.g., BdM) para obter *feedback* e garantir o alinhamento com os requisitos de supervisão.

Fase 3: Integração e Operacionalização (Meses 9+)

- **Integração no *Pipeline de DevSecOps:* O** *framework* é totalmente integrado nos *pipeline*s de CI/CD, automatizando a detecção de vulnerabilidades em cada *commit* de código.

- **Reporte Regulatório Automatizado:** Desenvolvimento de dashboards e relatórios automatizados para a supervisão prudencial, fornecendo uma visão contínua da postura de segurança.
- **Monitorização Contínua e Retreino:** Implementação de um ciclo contínuo de monitorização da performance do modelo e retreino com novos dados para garantir a sua adaptabilidade às ameaças emergentes.

10.1 Implicações para Economias Emergentes

Para economias emergentes, a adopção de IA na segurança cibernética de sistemas financeiros legados poderá representar uma oportunidade de "leapfrogging" tecnológico, embora esta hipótese dependa de condições prévias que não estão garantidas em todos os contextos. Dada a escassez de talentos altamente especializados em cibersegurança e a prevalência de infra-estruturas legadas, a IA poderia actuar como um multiplicador de força, permitindo que equipas menores protejam infra-estruturas complexas. Contudo, esta visão optimista deve ser temperada pela realidade de que a própria implementação e manutenção de sistemas de IA requer competências técnicas avançadas que podem ser igualmente escassas.

No contexto de Moçambique, o sistema financeiro enfrenta o duplo desafio de modernizar as suas infra-estruturas tecnológicas enquanto mantém a operacionalidade dos sistemas existentes. A eventual adopção do *framework* AI-VULN-FIN-LEGACY poderia, em teoria, permitir que as instituições financeiras moçambicanas:

- Reduzam a sua exposição a riscos cibernéticos sem a necessidade imediata de substituição total dos sistemas legados.
- Desenvolvam competências internas em IA e cibersegurança, contribuindo para o ecossistema tecnológico nacional.
- Explorem modelos de cooperação regional em matéria de supervisão cibernética.

Contudo, a concretização destas oportunidades depende de factores como: disponibilidade de financiamento, capacidade de retenção de talentos em data science, qualidade da infra-estrutura de telecomunicações e vontade política para a criação de um quadro regulatório específico para IA no sector financeiro.

11. CONCLUSÕES E CONTRIBUIÇÃO CIENTÍFICA

11.1 Conclusões

O presente estudo investigou a aplicação de técnicas de Inteligência Artificial especificamente *Graph Neural Networks* (GNN) e *Large Language Models* (LLM) para a detecção automatizada de vulnerabilidades em código bancário legado. Com base em validação experimental realizada em ambiente simulado e restrita a Java 8 e PL/SQL, o *framework* AI-VULN-FIN-LEGACY apresentou resultados que sugerem uma superioridade estatística em relação aos métodos tradicionais de SAST, com um F1-Score de 0,90 (vs. 0,42 do *baseline*) e uma redução estimada do MTTD de 72h para 32,4h e do MTTR de 48h para 28,8h.

Contudo, é fundamental contextualizar estes resultados com as seguintes ressalvas:

1. **Validade externa limitada:** Os resultados foram obtidos em ambiente simulado com datasets públicos e sintéticos. A generalização para ambientes de produção reais com código proprietário, dependências complexas e restrições operacionais não está comprovada e requer validação piloto.

2. **Escopo linguístico restrito:** A ausência de COBOL, linguagem ainda prevalente em muitos sistemas bancários legados, limita o alcance das conclusões sobre "sistemas financeiros legados" em sentido amplo.

3. **Pressupostos do *baseline*:** Os valores de MTTD e MTTR *baseline* (72h e 48h) são estimativas baseadas em pressupostos, não medições em bancos reais. As reduções percentuais devem ser interpretadas neste contexto.

4. **Propostas conceptuais vs. resultados empíricos:** As secções sobre SupTech, integração com o BdM, *roadmap* de implementação e modelo de accountability constituem propostas conceptuais não validadas empiricamente.

A análise por subgrupos revelou que o *framework* apresenta performance particularmente forte na detecção de vulnerabilidades de alta severidade (F1 = 0,93 para CVSS Crítica) e em categorias CWE bem representadas nos dados de treino (e.g., SQL Injection com F1 = 0,95). Contudo, a análise de falsos negativos identificou áreas onde a intervenção humana permanece indispensável, nomeadamente na detecção de vulnerabilidades de lógica de negócio e condições de corrida.

O trabalho sublinha a importância de uma governação algorítmica rigorosa e de um modelo de supervisão prudencial adaptado (SupTech) para mitigar os riscos inerentes à IA, incluindo viés, opacidade, ataques adversariais, riscos de privacidade, dependência tecnológica e accountability. A

articulação proposta com os princípios de Basileia III e o NIST AI Risk Management *Framework* oferece um enquadramento teórico para a adopção segura e auditável, cuja operacionalização requer validação prática.

No contexto de Moçambique, o *framework* poderá representar uma oportunidade para o Banco de Moçambique explorar capacidades de SupTech, embora a sua implementação dependa de condições prévias como a disponibilidade de recursos humanos qualificados, infra-estrutura computacional adequada e um quadro regulatório específico para IA no sector financeiro.

11.2 Contribuição Científica

Este trabalho procura contribuir para a intersecção entre a ciência da computação (*Deep Learning* em Grafos e Processamento de Linguagem Natural para Código) e a economia macroprudencial. As principais contribuições científicas propostas incluem:

- **Formalização Matemática do Risco Sistémico:** A proposição de um modelo matemático formal para quantificar o risco sistémico induzido por vulnerabilidades de software em infra-estruturas financeiras interconectadas. Este modelo constitui uma proposta teórica cuja calibração empírica requer dados de produção reais.
- ***Framework* Híbrido para Legados (Java 8 e PL/SQL):** O desenvolvimento e a validação empírica (em ambiente simulado) de um *framework* híbrido (GNN + LLM) adaptado para código bancário legado em Java 8 e PL/SQL. A contribuição reside na engenharia de integração e na validação experimental, não na criação de novos algoritmos. A extensão para outras linguagens legadas (COBOL, Natural, RPG) permanece como trabalho futuro.
- **Modelo de Governação Proposto para *Supervisory Technology*:** A proposição de um modelo de governação algorítmica para a adopção de IA em funções de supervisão bancária, com foco em explicabilidade, rastreabilidade e controlo humano. Este modelo é conceptual e requer validação junto de reguladores.
- **Discussão de Implicações para Economias Emergentes:** A exploração das implicações e do potencial de "leapfrogging" tecnológico para economias emergentes, com foco no contexto moçambicano. Esta discussão é de natureza propositiva e depende de condições locais específicas.
- **Análise de Riscos:** A discussão sistemática de riscos de privacidade, data residency, vendor lock-in e accountability, com propostas de mitigação, contribuindo para o debate sobre a adopção responsável de IA no sector financeiro.

A replicabilidade deste modelo para outras economias emergentes com infra-estruturas financeiras semelhantes depende de validação adicional em ambientes de produção reais. O presente estudo oferece evidência preliminar e um enquadramento teórico para a integração da IA na defesa cibernética do sector financeiro, cujo sucesso prático dependerá de factores contextuais, institucionais e regulatórios específicos de cada jurisdição.

11.3 Direcções para Investigação Futura

Com base nas limitações identificadas, propõem-se as seguintes direcções prioritárias para investigação futura:

1. **Validação em ambiente de produção real:** Realização de estudos piloto em instituições financeiras reais, com código proprietário anonimizado, para avaliar a validade externa dos resultados.

2. **Extensão para COBOL:** Desenvolvimento de *parsers* COBOL-to-CPG e *fine-tuning de LLMs* em código COBOL, permitindo a cobertura de uma parcela significativa dos sistemas legados globais.

3. **Validação do modelo de risco sistémico:** Calibração empírica do modelo matemático proposto com dados de incidentes reais e topologias de interconexão bancária.

4. **Estudo longitudinal de concept *drift*:** Avaliação da degradação da performance do modelo ao longo do tempo e definição de critérios óptimos de retreino.

5. **Validação do modelo de governação:** Consulta a reguladores e instituições financeiras para validar a exequibilidade e adequação do modelo de accountability proposto.

6. **Análise comparativa com ferramentas comerciais:** Comparação directa com ferramentas SAST comerciais de última geração (e.g., Checkmarx, Snyk) em datasets idênticos.

12. REFERÊNCIAS BIBLIOGRÁFICAS

[1] P. Kruchten, R. L. Nord, and I. Ozkaya, “Technical Debt: From Metaphor to Theory and Practice,” *IEEE Software*, vol. 29, no. 6, pp. 18–21, 2012. doi: 10.1109/MS.2012.167.

[2] R. C. Seacord, D. Plakosh, and G. A. Lewis, *Modernizing Legacy Systems: Software Technologies, Engineering Processes, and Business Practices*. Boston: Addison-Wesley, 2003.

[3] OWASP Foundation, *OWASP Top 10: The Ten Most Critical Web Application Security Risks*, 2021. Available: https://owasp.org/www-project-top-ten/

[4] Financial Stability Board, *The Financial Stability Implications of Artificial Intelligence*. Basel: FSB, 2024. Available: https://www.fsb.org/uploads/P14112024.pdf

[5] D. K. G. de Araujo, S. Doerr, L. Gambacorta, and B. Tissot, “Artificial intelligence in central banking,” *BIS Bulletin*, no. 84, Bank for International Settlements, 2024. Available: https://www.bis.org/publ/bisbull84.htm

[6] J. C. Crisanto, J. Ehrentraud, J. Prenio, and J. Yong, *Smart Supervision: Sound Capacity Development Approaches for Tech-Savvy Supervisors*, FSI Insights on Policy Implementation, no. 70. Basel: Bank for International Settlements, 2025. Available: https://www.bis.org/fsi/publ/insights70.pdf

[7] International Monetary Fund, *AI Projects in Financial Supervisory Authorities*. Washington, DC: IMF, 2025. Available: https://www.elibrary.imf.org/view/journals/001/2025/199/article-A001-en.xml

[8] I. Aldasoro, L. Gambacorta, A. Korinek, V. Shreeti, and M. Stein, “Intelligent financial *System*: How AI is transforming finance,” *BIS Working Papers*, no. 1194, Bank for International Settlements, 2024. Available: https://www.bis.org/publ/work1194.htm

[9] International Organization for Standardization, *ISO/IEC 27001:2022 — Information Security, Cybersecurity and Privacy Protection — Information Security Management Systems — Requirements*. Geneva: ISO, 2022.

[10] S. Shimmi, H. Okhravi, and M. Rahimi, “AI-Based Software Vulnerability Detection: A *Syste*matic Literature Review,” arXiv preprint arXiv:2506.10280, 2025. Available: https://arxiv.org/abs/2506.10280

[11] Y. Zhou, S. Liu, J. Siow, X. Du, and Y. Liu, “Devign: Effective Vulnerability Identification by Learning Comprehensive Program Semantics via *Graph Neural Networks*,” in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[12] B. Wu, “Code Vulnerability Detection Based on Deep Sequence Modeling and Graph Modeling Technologies,” *Journal of Electrical and Computer Engineering*, vol. 2022, 2022. doi: 10.1155/2022/6929015.

[13] X. Cheng, H. Wang, J. Hua, G. Xu, and Y. Sui, “DeepWukong: Statically Detecting Software Vulnerabilities Using Deep *Graph Neural Network*,” *ACM Transactions on Software Engineering and Methodology*, vol. 30, no. 4, pp. 1–33, 2021. doi: 10.1145/3436877.

[14] Z. Feng et al., “CodeBERT : A Pre-Trained Model for Programming and Natural Languages,” in *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1536–1547, 2020. doi: 10.18653/v1/2020.findings-emnlp.139.

[15] Board of Governors of the Federal Reserve *System* and Office of the Comptroller of the Currency, *SR 11-7: Guidance on Model Risk Management*. Washington, DC: Federal Reserve Board, 2011. Available: https://www.federalreserve.gov/supervisionreg/srletters/sr1107.htm

[16] S. M. Lundberg and S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[17] C. O’Neil, *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. New York: Crown, 2016.

[18] N. Papernot et al., “The Limitations of *Deep Learning* in Adversarial Settings,” in *Proceedings of the IEEE European Symposium on Security and Privacy*, pp. 372–387, 2016. doi: 10.1109/EuroSP.2016.36.

[19] National Institute of Standards and Technology, *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*, NIST AI 100-1. Gaithersburg, MD: NIST, 2023. doi: 10.6028/NIST.AI.100-1.

[20] Basel *Commit*tee on Banking Supervision, *Basel III: Finalising Post-Crisis Reforms*. Basel: Bank for International Settlements, 2017. Available: https://www.bis.org/bcbs/publ/d424.htm

[21] R. K. Yin, *Case Study Research and Applications: Design and Methods*, 6th ed. Thousand Oaks, CA: SAGE Publications, 2018.

[22] National Institute of Standards and Technology, *Juliet Test Suite for C/C++ and Java*. Gaithersburg, MD: NIST, 2017. Available: https://samate.nist.gov/SARD/test-suites/112

[23] J. Fan et al., “A C/C++ Code Vulnerability Dataset with Code Changes and CVE Summaries,” in *Proceedings of the 17th International Conference on Mining Software Repositories*, pp. 508–512, 2020. doi: 10.1145/3379597.3387501.

[24] MITRE, *Common Weakness Enumeration (CWE)*, 2024. Available: https://cwe.mitre.org/

[25] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic Minority Over-Sampling Technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002. doi: 10.1613/jair.953.

[26] I. Žliobaitė, “A *Survey* on Concept *Drift* Adaptation,” *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–37, 2014. doi: 10.1145/2523813.

[27] Banco de Moçambique, *Aviso n.º 02/GBM/2024: Aprova as Directrizes de Gestão do Risco e Resiliência Cibernética*. Maputo: Banco de Moçambique, 2024. Available: https://bancomoc.mz/media/53rfngce/aviso-n-%C2%BA-02-gbm-2024-directrizes-de-gest%C3%A3o-do-risco-e-resili%C3%AAncia-cibern%C3%A9tica.pdf

[28] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and Harnessing Adversarial Examples,” in *Proceedings of the International Conference on Learning Representations*, 2015. Available: https://arxiv.org/abs/1412.6572

[29] Z. Li et al., “Vulnerability Detection with Fine-Grained Interpretations,” in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 292–303, 2021. doi: 10.1145/3468264.3468597.

[30] J. Prenio, “Peering Through the Hype: Assessing SupTech Tools’ Transition from Experimentation to Supervision,” *FSI Insights on Policy Implementation*, no. 58. Basel: Bank for International Settlements, 2024. Available: https://www.bis.org/fsi/publ/insights58.html

[31] A. Singh, S. Sanyal, e S. Sanyal, “FinBERT: Financial Sentiment Analysis with Pre-trained Language Models,” *arXiv preprint arXiv:2006.08097*, 2021. doi: 10.48550/arXiv.2006.08097.

GLOSSÁRIO DE TERMOS TÉCNICOS

Termo	Definição
Sistema Legado	Plataforma computacional desenvolvida há mais de uma década, assente em linguagens de programação de gerações anteriores (e.g., COBOL, PL/SQL, versões anteriores a Java 11), arquitecturas monolíticas, *middleware* proprietário e bases de dados sem suporte activo do fabricante.
Dívida Técnica	Consequências a longo prazo de decisões de desenvolvimento de software que priorizam a velocidade em detrimento da qualidade ou da arquitectura ideal, resultando em custos acumulados de manutenção e risco operacional.
SAST (Static Application Security Testing)	Análise Estática de Segurança de Aplicações — técnica de teste de segurança que analisa o código-fonte sem o executar, utilizando regras predefinidas para identificar vulnerabilidades.
GNN (Graph Neural Network)	Rede Neural de Grafos — classe de redes neurais profundas que operam sobre estruturas de dados em forma de grafo, capazes de aprender representações de nós e arestas.
GraphSAGE	Algoritmo específico de GNN que utiliza amostragem e agregação de vizinhos para gerar *embeddings* de nós de forma indutiva.
LLM (Large Language Model)	Modelo de Linguagem de Grande Porte — modelo de inteligência artificial treinado em vastos corpora de texto, capaz de compreender e gerar linguagem natural e código.
CodeBERT	Modelo de linguagem pré-treinado desenvolvido pela Microsoft, optimizado para compreensão e geração de código-fonte em múltiplas linguagens de programação.
CPG (Code Property Graph)	Grafo de Atributos de Código — representação unificada que combina *Abstract Syntax Tree* (AST), *Control Flow Graph* (CFG) e *Data Flow Graph* (DFG) de um programa.
AST (Abstract Syntax Tree)	Árvore Sintáctica Abstracta — representação hierárquica da estrutura sintáctica do código-fonte.
CFG (*Control Flow Graph*)	Grafo de Fluxo de Controlo — representação dos caminhos de execução possíveis num programa.
DFG (*Data Flow Graph*)	Grafo de Fluxo de Dados — representação das dependências de dados entre instruções de um programa.
MTTD (Mean Time To Detect)	Tempo Médio de Detecção — métrica que mede o tempo médio entre a introdução de uma vulnerabilidade e a sua identificação.
MTTR (Mean Time To Respond/Repair)	Tempo Médio de Resposta/Reparação — métrica que mede o tempo médio entre a detecção de uma vulnerabilidade e a sua correção.
CVSS (Common Vulnerability Scoring *System*)	Sistema Comum de Pontuação de Vulnerabilidades — *framework* padronizado para classificar a severidade de vulnerabilidades de segurança.

CWE (Common Weakness Enumeration)	Enumeração de Fraquezas Comuns — catálogo de tipos de fraquezas de software mantido pelo MITRE.

Termo	Definição
CVE (Common Vulnerabilities and Exposures)	Vulnerabilidades e Exposições Comuns — lista de vulnerabilidades de segurança publicamente conhecidas.
XAI (Explainable AI)	Inteligência Artificial Explicável — conjunto de técnicas que permitem compreender e interpretar as decisões de modelos de IA.
SHAP (*SHapley Additive exPlanations*)	Técnica de XAI baseada na teoria dos jogos que atribui um valor de importância a cada feature para uma previsão específica.
LIME (Local Interpretable Model-agnostic Explanations)	Técnica de XAI que gera explicações locais para previsões individuais de qualquer modelo de classificação.
MRM (Model Risk Management)	Gestão de Risco de Modelo — processo de identificação, avaliação e mitigação dos riscos associados à utilização de modelos quantitativos.
SupTech (*Supervisory Technology*)	Tecnologia de Supervisão — uso de tecnologias inovadoras pelas autoridades de supervisão para apoiar e aprimorar a supervisão regulatória.
DevSecOps	Abordagem que integra práticas de segurança no ciclo de vida de desenvolvimento de software (DevOps).
FPR (*False positive* Rate)	Taxa de Falsos Positivos — proporção de exemplos não vulneráveis incorrectamente classificados como vulneráveis.
ROC-AUC	Área Sob a Curva da Característica Operacional do Receptor — métrica que mede a capacidade discriminatória de um classificador binário.
SMOTE	Synthetic Minority Over-sampling Technique — técnica de balanceamento de classes que gera exemplos sintéticos da classe minoritária.
Embedding	Representação vetorial de alta dimensão que captura as propriedades semânticas e sintácticas de um elemento (e.g., token de código, nó de grafo).
Fine-tuning	Processo de ajuste fino de um modelo pré-treinado para uma tarefa específica, utilizando um dataset especializado.
Data drift	Mudança na distribuição dos dados de entrada ao longo do tempo, que pode degradar a performance de um modelo de IA.
Vendor Lock-in	Dependência excessiva de um fornecedor específico de tecnologia, que dificulta a migração para soluções alternativas.
Data Residency	Requisito de que os dados sejam armazenados e processados dentro de uma jurisdição geográfica específica.

BdM	Banco de Moçambique — banco central e autoridade de supervisão financeira de Moçambique.
HITL (*Human-in-the-Loop*)	Controlo Humano no Ciclo — mecanismo que exige validação humana em decisões críticas de sistemas automatizados.