Dataset e Análise de Sensibilidade de *Trojans* em Circuitos Quânticos

Wildisley José de Souza Filho [1], **Victor Takashi Hayashi** [2], **Bryan Kano Ferreira** [1]

[1]Instituto de Tecnologia e Liderança
São Paulo, São Paulo – Brasil

[2]Universidade de São Paulo
São Paulo, São Paulo – Brasil

wildisley.filho@sou.inteli.edu.br, victor.hayashi@usp.br

bryan.ferreira@prof.inteli.edu.br

Abstract. *This work presents a dataset and sensitivity analysis of quantum trojans inserted during quantum circuit compilation. Using circuits from RevLib, QASMBench, and VeriQBench, the range of circuits explored in the literature was extended, producing a dataset with 535 golden circuits and 5,350 infected QASM variants. The impact of the insertions is evaluated through simulation-based distribution metrics, while text-based classifiers and SHAP analysis are used to study detection and relevant trojan attributes. The results show measurable behavioral changes and indicate that textual detection approaches can distinguish legitimate and infected circuits, reaching 66.0% accuracy and 73.3% balanced accuracy in the most consistent setting.*

Resumo. *Este trabalho apresenta um dataset e análise de sensibilidade de quantum trojans inseridos durante a compilação de circuitos quânticos. A partir de circuitos do RevLib, QASMBench e VeriQBench, estendeu-se a gama de circuitos explorados pela literatura gerando um dataset com 535 circuitos golden e 5.350 variantes QASM infectadas. O impacto das inserções é avaliado por métricas de distribuição via simulação, enquanto classificadores textuais e análise SHAP são usados para estudar a detecção e os atributos relevantes dos trojans. Os resultados mostram alterações comportamentais mensuráveis e indicam que abordagens textuais de detecção conseguem distinguir circuitos legítimos e infectados, alcançando 66,0% de acurácia e 73,3% de acurácia balanceada no cenário mais consistente.*

1. Introdução

Avanços recentes de empresas como Google e Microsoft [Neven 2024, Bolgar 2025], somados à designação de 2025 como o Ano Internacional da Ciência e Tecnologia Quântica [UNESCO 2025], ampliaram o interesse pela computação quântica. A área explora superposição, emaranhamento e interferência para processar informação em *qubits*, com aplicações potenciais em otimização, simulação, cibersegurança e aprendizado de máquina [Nielsen and Chuang 2001].

Apesar desse potencial, o uso prático de computadores quânticos depende de uma cadeia de ferramentas complexa. Em geral, o usuário descreve um algoritmo como circuito quântico (análogo a circuitos booleanos clássicos), que é processado por compiladores e transpiladores responsáveis por otimizar, reescrever e adaptar o circuito ao hardware ou simulador de destino; ao final, os resultados são obtidos como distribuições de probabilidade [Das and Ghosh 2024, Cicero et al. 2025]. Essa complexidade, combinada à falta de práticas consolidadas de verificação, amplia a superfície de ataque e permite que compiladores, passes de otimização ou pacotes auxiliares sejam explorados como vetores para a inserção furtiva de *trojans*.

Quando considerados no contexto de compiladores quânticos, *trojans* podem modificar circuitos em diferentes etapas do processo de transformação, incluindo passes de otimização, inserção de operações intermediárias e alterações no circuito final. Esses ataques são plausíveis porque o comportamento probabilístico dos algoritmos quânticos e as mudanças estruturais naturalmente realizadas durante otimização e mapeamento para hardware podem mascarar modificações maliciosas como se fossem transformações legítimas [John et al. 2025, Das and Ghosh 2023].

Ainda assim, a detecção de *Quantum Trojans* permanece limitada pela ausência de datasets públicos com pares bem definidos de circuitos legítimos e infectados, abrangendo diferentes algoritmos, estratégias de inserção e cenários experimentais. Embora trabalhos como [Das and Ghosh 2023] e [John et al. 2025] apresentem propostas iniciais de trojans e métodos de detecção, os recursos disponíveis ainda são restritos. Diante dessa lacuna, este trabalho propõe a construção de um dataset público de *Quantum Trojans* e a análise de sensibilidade das inserções, permitindo avaliar seu impacto comportamental e apoiar o desenvolvimento de métodos automáticos de detecção.

Nesse contexto, a Seção 2 apresenta o modelo adversarial, o processo de geração do dataset e a metodologia de detecção. Em seguida, a Seção 3 traz a caracterização do conjunto gerado, a análise de sensibilidade das inserções e o desempenho dos classificadores avaliados. Por fim, a Seção 4 sintetiza os principais achados e aponta direções para trabalhos futuros.

2. Metodologia

A metodologia deste trabalho divide-se em quatro etapas principais. Primeiro, há a definição de um modelo adversarial voltado à inserção de *trojans* durante a etapa de compilação/transpilação de circuitos quânticos. Em seguida, foi desenvolvido um pipeline para coletar circuitos de diferentes benchmarks, filtrar amostras elegíveis e gerar pares legítimo/infectado com metadados estruturados. Posteriormente, os circuitos foram avaliados por simulação, comparando suas distribuições de saída por métricas de sensibilidade. Por fim, foram conduzidos experimentos de detecção supervisionada a partir de representações textuais dos arquivos QASM, complementados por análise de interpretabilidade.[1]

[1]Durante a redação deste trabalho, utilizou-se inteligência artificial generativa como apoio textual. Em particular, foi empregado o modelo ChatGPT, da plataforma OpenAI, para revisão linguística, organização de trechos e melhoria de clareza, sem substituir a definição metodológica, a análise dos resultados ou a responsabilidade autoral.

2.1. Modelo Adversarial

O modelo adversarial considerado neste trabalho assume um cenário de ataque à cadeia de ferramentas de desenvolvimento quântico. Conforme sintetizado na Figura 1, ataques podem ocorrer em diferentes pontos da *stack* quântica, desde camadas de software e compilação até interfaces com provedores e dispositivos físicos [Das and Ghosh 2024]. Este trabalho concentra-se especificamente na camada de compilação/transpilação. O cenário deve ser entendido como uma hipótese de ataque à cadeia de suprimentos de software (*software supply chain*), e não como o relato de um incidente real já observado: assume-se que uma dependência, extensão ou componente de compilação inicialmente considerado confiável seja comprometido ou substituído por uma implementação maliciosa.

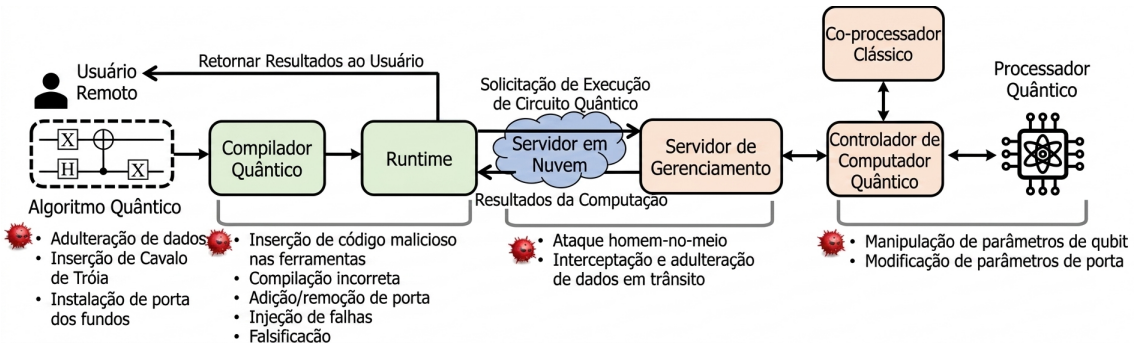

Figura 1. Pontos de ataque na *stack* quântica adaptado de [Das and Ghosh 2024].

Essa hipótese é motivada pelo uso de compiladores quânticos de código aberto, extensíveis e modulares, amplamente adotados pela comunidade, cujos componentes realizam síntese, roteamento, otimizações e adaptação a diferentes *backends* [Kharkov et al. 2022]. No Qiskit, a transpilação é composta por passes configuráveis [Javadi-Abhari et al. 2024]; essa flexibilidade facilita a incorporação de novas técnicas, mas cria uma fronteira de confiança ao incluir pacotes de terceiros. Nesse contexto, [Das and Ghosh 2023] considera motivação semelhante ao discutir compiladores não verificados que oferecem otimizações recentes, mas podem adulterar circuitos.

Nesse cenário, o atacante distribui um pacote que se apresenta como otimizador legítimo e implementa a interface `qiskit.transpiler.passes`. Incorporado a um *PassManager*, o componente malicioso atua durante a compilação junto aos passes de análise, otimização e mapeamento, sem exigir controle sobre o código-fonte do algoritmo, o provedor ou o hardware quântico.

Como o processo de transpilação pode adicionar operações de roteamento, substituir e decompor portas, remapear qubits e fundir ou cancelar operações, a representação textual original não é preservada [Das and Ghosh 2023]. Sendo assim, passes legítimos executados após a inserção podem reescrever a porta maliciosa e incorporá-la a uma estrutura distinta. Um analisador sintático ainda poderia detectar inserções diretas mediante uma versão *golden* confiável ou regras estritas, mas a comparação após a compilação é dificultada por transformações legítimas e pela ausência de uma representação canônica. Além disso, verificações anteriores ao passe comprometido não impedem alterações pos-

teriores. Assim, a furtividade considerada decorre da integração ao compilador, não da suposição de que as inserções simples do dataset sejam indetectáveis.

Por fim, considera-se que o atacante conhece a estrutura geral dos circuitos e a API do compilador, mas não controla a avaliação do pesquisador. Seu objetivo é inserir portas em posições aleatórias ou espaços ociosos, além de operações condicionais ou padrões que afetem apenas determinados estados, entradas ou configurações. O circuito pode, assim, manter aparência funcional em testes simples, mas apresentar degradação de desempenho, redução de fidelidade ou alterações probabilísticas específicas. O defensor tem acesso as versões anterior e posterior ao otimizador e suas medições, permitindo construir pares legítimo/infectado para analisar os efeitos das inserções e investigar técnicas de detecção.

2.2. Geração do Dataset

Um script automatiza a descoberta, filtragem, infecção e avaliação dos circuitos. As inserções são determinadas por uma semente controlada, garantindo reprodutibilidade e permitindo variar circuitos, portas, qubits e posições. O pipeline responde às limitações de estudos anteriores: [John et al. 2025] propõe inserções em espaços ociosos da DAG, mas não disponibiliza um dataset amplo, enquanto [Das and Ghosh 2023] restringe a detecção baseada em matrizes unitárias e CNNs a circuitos QAOA e poucos *trojans*. Este trabalho amplia essa cobertura com fontes heterogêneas, variantes sistemáticas e metadados rastreáveis.

O pipeline descobre arquivos `.qasm` e `.real` provenientes do RevLib [Wille et al. 2008], QASMBench [Li et al. 2022] e VeriQBench [Chen et al. 2022]. Circuitos QASM são normalizados e hasheados para remoção de duplicatas; em seguida, todas as amostras recebem metadados de origem e estrutura. Arquivos `.qasm` são importados como OpenQASM 2, enquanto arquivos `.real` são convertidos por portas da família de Toffoli. A filtragem considera número de qubits, profundidade e erros de *parsing*, mantendo no inventário os circuitos descartados e seus motivos de exclusão.

Para cada circuito elegível, exporta-se uma versão *golden* e geram-se variantes infectadas pela inserção de uma porta `H`, `T`, `X`, `Y` ou `Z`. A inserção ocorre em uma posição de um qubit selecionado ou em uma camada ociosa da DAG, aproximando-se da estratégia de [John et al. 2025]. Registram-se porta, modo, qubit, posição e alterações de profundidade e quantidade de operações. As portas simples constituem uma abstração controlada para medir impacto; o experimento não avalia ofuscação adicional nem quantifica a furtividade após a reaplicação completa dos passes de otimização.

Por fim, os pares legítimo/infectado são simulados no `AerSimulator`, comparando-se suas distribuições por distância de variação total (TVD), coeficiente de Bhattacharyya (BC) e distância de Bhattacharyya (BD), com repetições para estimar médias e desvios-padrão. O dataset é organizado em diretórios de circuitos e arquivos `CSV` de metadados, sensibilidade, inventário e erros, garantindo rastreabilidade e suporte às análises de detecção.

2.3. Detecção dos Trojans

A detecção dos *trojans* foi formulada como um problema de classificação binária entre circuitos quânticos *golden* e circuitos infectados. Para isso, os arquivos QASM foram

tratados como dados textuais e representados por Bag-of-Words e TF-IDF, considerando tokens e bigramas extraídos do código. Essas representações foram avaliadas com classificadores supervisionados, incluindo Regressão Logística, Multinomial e Complement Naive Bayes e Linear SVC.

Para reduzir o efeito do desbalanceamento, aplicou-se subamostragem aleatória da classe majoritária somente ao conjunto de treino de cada *fold*. Os circuitos infectados do treino foram sorteados, com semente distinta por *fold*, até igualarem a quantidade de circuitos *golden*; as amostras excedentes foram descartadas. O conjunto de teste não foi alterado, preservando a proporção real de aproximadamente um circuito *golden* para dez infectados. Complementarmente, para a Regressão Logística e Linear SVC as classes foram também ponderadas na função de perda; embora parcialmente redundante após a subamostragem, esse ajuste não modifica a composição dos dados de teste.

A validação foi conduzida com StratifiedKFold e, principalmente, com GroupKFold. Este último foi adotado por representar um cenário de avaliação mais realista, pois mantém todas as variantes de um mesmo circuito base no mesmo *fold*, evitando que versões *golden* e infectadas muito semelhantes apareçam simultaneamente nos conjuntos de treino e teste. As métricas analisadas foram acurácia, acurácia balanceada, precisão, *recall*, F1-score e ROC-AUC, além de matrizes de confusão.

3. Resultados

O pipeline de geração resultou em um dataset composto por 535 circuitos *golden*, cada um associado a 10 variantes infectadas, totalizando 5.350 circuitos com inserção de *trojans*. Essa organização preserva a relação entre cada circuito base e suas respectivas versões modificadas, permitindo tanto análises pareadas quanto avaliações supervisionadas de classificação entre amostras legítimas e infectadas. O repositório com os artefatos do trabalho está disponível em https://github.com/apenas-will/IC-QT-2026[2].

Para além disso, foi realizada uma análise de sensibilidade para medir o impacto físico e comportamental das inserções maliciosas. Para tal, cada par *golden* / *infected* foi simulado com 512 *shots* e 15 repetições, comparando-se as distribuições de saída por meio da distância de variação total (TVD), do coeficiente de Bhattacharyya (BC) e da distância de Bhattacharyya (BD). Essas métricas permitiram caracterizar a intensidade da alteração produzida pelos *trojans* nas distribuições de medição, fornecendo uma métrica quantitativa do desvio entre o comportamento esperado e o comportamento observado após a infecção.

Buscando explorar como diferentes atributos da inserção impactam nos valores da TVD diversas visualizações foram geradas, trazendo maior compreensão do comportamento do sistema. Nesse sentido, na Figura 2 tem-se a distribuição dos valores de TVD agrupado por porta inserida. Pode-se notar que as portas X e Y produziram as maiores variações de TVD, com grande concentração de valores acima de 0,8. Esse comportamento é plausível, pois ambas realizam inversão de bit, alterando diretamente a base computacional do qubit; no caso da porta Y, há ainda a presença de inversão de fase. A porta Hadamard também se destaca, com concentração média em torno de 0,5, o que pode ser parcialmente explicado por sua ação de superposição: considerando a TVD como

[2]`https://github.com/apenas-will/IC-QT-2026`

métrica, parte das inserções tende a produzir alteração perceptível, enquanto outra parte pode ter impacto reduzido dependendo do estado e do contexto do circuito.

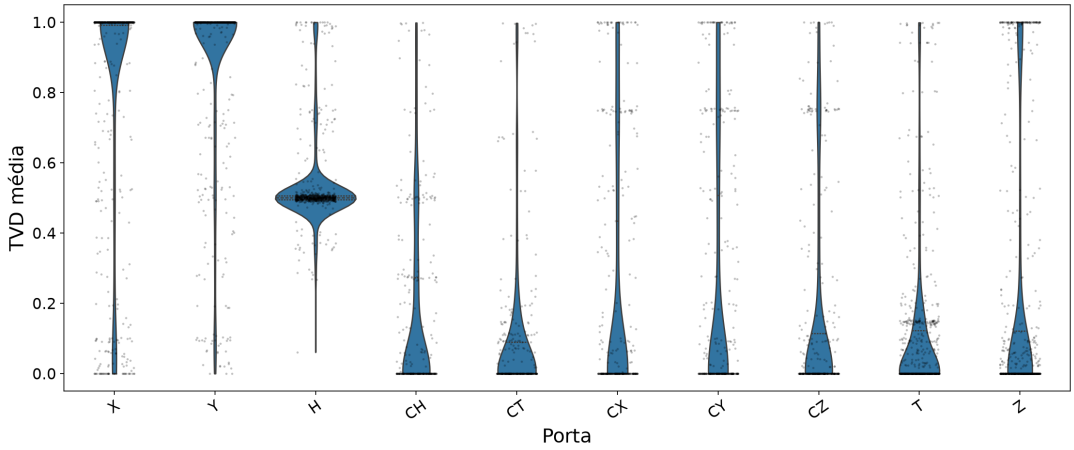

Figura 2. Distribuição da TVD média por porta inserida. As portas X e Y produzem os maiores impactos comportamentais, enquanto portas de fase e controladas tendem a apresentar valores menores.

As demais operações, como T, Z e portas controladas, apresentaram valores de TVD média consistentemente menores, embora sem impacto uniforme. No caso de portas que afetam apenas a fase, como Z, o menor impacto é coerente com o fato de que alterações puramente de fase nem sempre se manifestam diretamente nas distribuições de medição na base computacional. Para portas controladas, parte dos resultados reduzidos pode estar associada à configuração experimental, na qual as simulações partem do estado inicial $|0\rangle$. Assim, trojans baseados em portas controladas podem permanecer pouco perceptíveis em um cenário padrão, mas causar impacto maior sob estados iniciais específicos. Além disso, efeitos de interferência, como *phase kickback*, podem influenciar os resultados de forma dependente da estrutura do circuito.

Na Figura 3, por sua vez, as distribuições por modo de inserção mostram que inserções aleatórias e inserções em espaços vazios apresentam padrões de impacto bastante semelhantes, com concentração de valores nas extremidades e também próximos de 0,5. Por outro lado, as inserções com portas controladas concentram resultados próximos de zero, reforçando a hipótese de que a configuração com estado inicial $|0\rangle$ reduz a ativação ou o efeito observável desse tipo de operação em parte dos circuitos avaliados.

Já na Figura 4, observa-se que a relação entre impacto e características estruturais do circuito não é direta. Em relação à quantidade de qubits, há uma tendência de aumento da TVD média conforme cresce o número de qubits do circuito. Entretanto, ao relacionar a TVD com a profundidade ou com o tamanho total do circuito, não se verifica um padrão bem definido, havendo alta variabilidade dos valores médios para circuitos com dimensões semelhantes.

Além da análise agregada das métricas de sensibilidade, foi utilizado o método SHAP para investigar a relevância das características associadas ao processo de inserção. Nessa etapa, empregou-se um modelo Random Forest combinado ao *TreeExplainer*, considerando atributos como porta inserida, posição, modo de inserção, qubit afetado, quan-

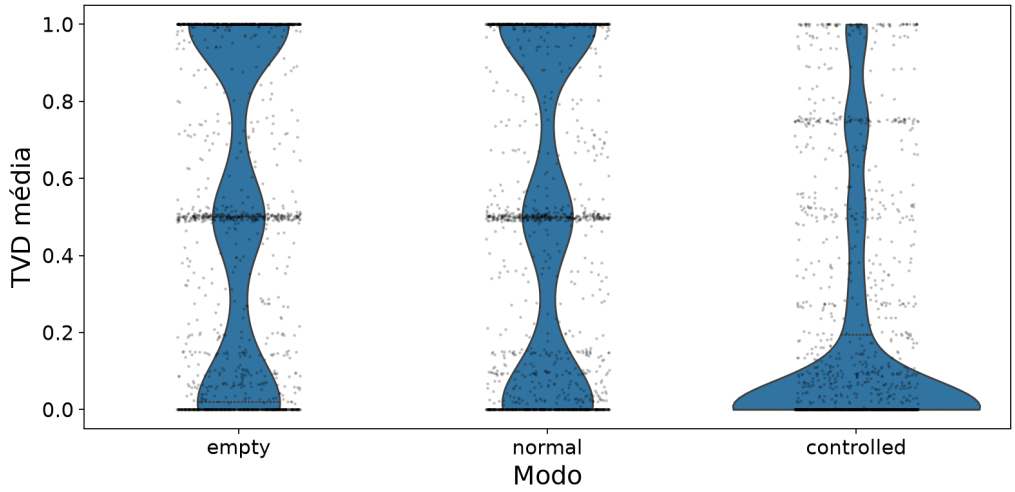

Figura 3. Distribuição da TVD média por modo de inserção. Os modos aleatório e em espaços vazios apresentam impactos semelhantes, enquanto inserções controladas se concentram próximas de zero no cenário iniciado em $|0\rangle$.

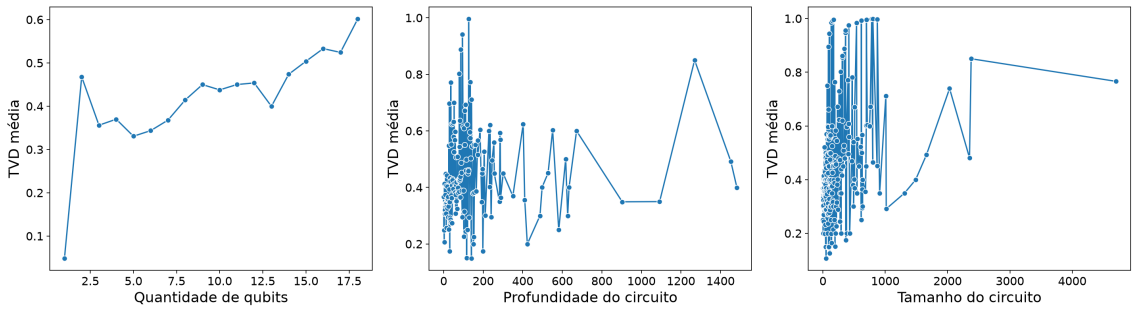

Figura 4. Relação entre a TVD média e as características estruturais dos circuitos. Observa-se tendência de aumento da TVD com o número de qubits, mas não há padrão bem definido em relação à profundidade ou à quantidade de operações.

tidade de bits clássicos e características estruturais do circuito. Essa análise teve como objetivo identificar quais fatores mais contribuíram para explicar a variação no impacto observado, oferecendo maior interpretabilidade sobre os padrões de sensibilidade dos circuitos infectados.

A Figura 5 mostra que a característica de maior impacto para explicar a TVD média foi a porta inserida, com diferença expressiva em relação às demais variáveis. Em seguida, aparecem o número de bits clássicos, a família do circuito e a posição de inserção. Esse resultado sugere que o impacto do trojan não depende apenas da natureza da operação maliciosa, mas também de características do próprio circuito, uma vez que atributos como quantidade de bits clássicos e família aparecem em posições elevadas no ranking. Uma análise detalhada das famílias de circuitos e de suas propriedades específicas, contudo, foge do escopo deste trabalho.

Ressalta-se que as métricas TVD, BC e BD foram utilizadas para caracterizar o efeito físico e comportamental dos *trojans*, mas não como entrada principal da etapa

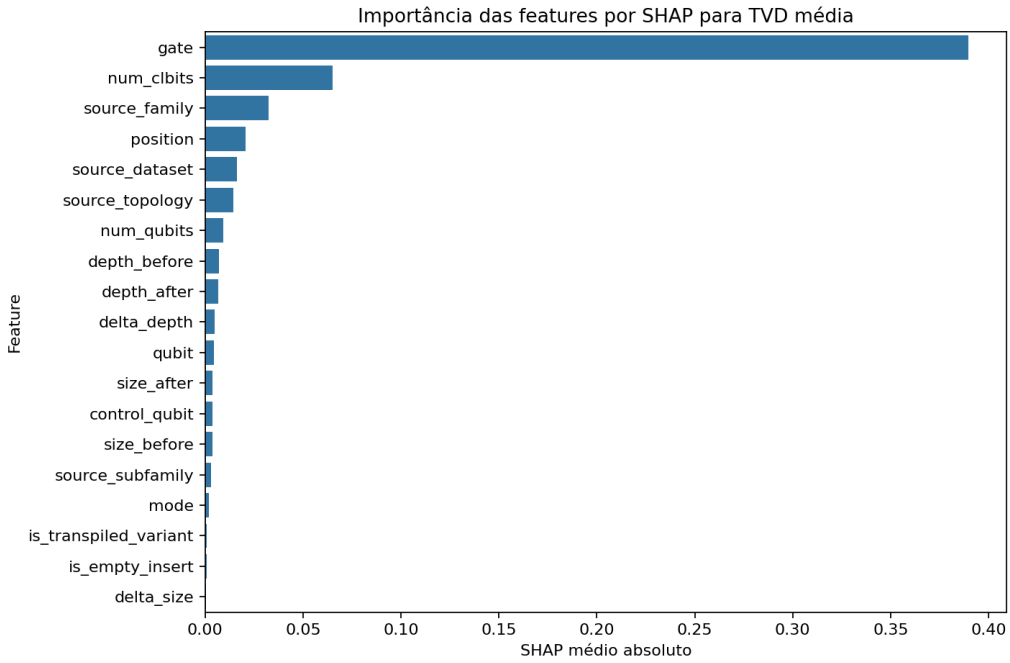

Figura 5. Importância média das características segundo o método SHAP aplicado ao modelo Random Forest.

de detecção supervisionada. Na classificação binária, a entrada dos modelos foi composta pelo texto QASM dos circuitos, representado por Bag-of-Words e TF-IDF. Assim, a análise de sensibilidade e a análise de detecção cumprem papéis complementares: a primeira quantifica o impacto das inserções nas distribuições de saída, enquanto a segunda avalia a possibilidade de distinguir circuitos legítimos e infectados a partir de seus padrões textuais.

Na Tabela 1, são apresentadas as métricas de classificação obtidas pelos diferentes métodos de detecção. Para o cálculo de precisão, *recall* e F1, a classe positiva corresponde aos circuitos infectados por *trojans*; portanto, essas métricas expressam a capacidade dos modelos de detectar circuitos infectados. No contexto de *trojans*, falsos negativos são particularmente críticos, pois representam circuitos infectados classificados como legítimos, que poderiam ser utilizados pelo usuário sem conhecimento da presença de trechos maliciosos capazes de afetar sua funcionalidade. Assim, diferentemente de cenários nos quais erros de falso positivo podem ser o principal custo, neste problema a falha em sinalizar um circuito infectado tende a ser mais grave do ponto de vista de segurança.

Por esse motivo, o *recall* é prioritário, pois mede a capacidade de identificar amostras infectadas e reduzir falsos negativos. Entretanto, tal medida deve ser analisada em conjunto com métricas sensíveis ao desempenho nas duas classes, como F1-score, F1 macro e acurácia balanceada. Um detector que classifique todas as amostras como infectadas pode alcançar *recall* igual a 1, mas é inútil na prática por não distinguir circuitos legítimos de maliciosos.

Com as técnicas de balanceamento aplicadas no treinamento, os classificadores se mostram capazes de produzir predições para ambas as classes, em vez de simplesmente atribuir todas as amostras à classe majoritária. O desempenho, contudo, se mos-

tra variável. No cenário *GroupKFold*, *BoW + LinearSVC* apresentou o resultado mais equilibrado, com acurácia balanceada de $0{,}733 \pm 0{,}02$, F1 de $0{,}773 \pm 0{,}05$ e precisão de $0{,}974 \pm 0{,}01$. Já *TF-IDF + LinearSVC* obteve a maior ROC-AUC ($0{,}723 \pm 0{,}02$). As combinações *BoW + ComplementNB* e *BoW + MultinomialNB* alcançaram maior acurácia e *recall* médios, mas apresentaram desvios-padrão elevados e acurácia balanceada próxima de $0{,}5$, indicando menor estabilidade e discriminação entre as classes.

Tabela 1. Desempenho dos classificadores supervisionados aplicados ao texto QASM (média ± desvio-padrão).

Pipeline	Acurácia	Acc. bal.	F1 macro	Precisão	Recall	F1	ROC-AUC
GroupKFold							
BoW + LinearSVC	0.660 ± 0.06	$\mathbf{0.733 \pm 0.02}$	$\mathbf{0.540 \pm 0.04}$	$\mathbf{0.974 \pm 0.01}$	0.643 ± 0.07	$\mathbf{0.773 \pm 0.05}$	0.711 ± 0.02
BoW + LogisticRegression	0.489 ± 0.05	0.558 ± 0.03	0.406 ± 0.03	0.931 ± 0.01	0.474 ± 0.06	0.626 ± 0.05	0.558 ± 0.02
BoW + ComplementNB	$\mathbf{0.707 \pm 0.31}$	0.506 ± 0.01	0.432 ± 0.16	0.918 ± 0.02	$\mathbf{0.752 \pm 0.38}$	0.759 ± 0.35	0.534 ± 0.02
BoW + MultinomialNB	$\mathbf{0.707 \pm 0.31}$	0.506 ± 0.01	0.432 ± 0.16	0.918 ± 0.02	$\mathbf{0.752 \pm 0.38}$	0.759 ± 0.35	0.534 ± 0.02
TF-IDF + LinearSVC	0.578 ± 0.04	0.662 ± 0.02	0.477 ± 0.03	0.960 ± 0.00	0.559 ± 0.05	0.706 ± 0.04	$\mathbf{0.723 \pm 0.02}$
TF-IDF + LogisticRegression	0.536 ± 0.04	0.608 ± 0.01	0.442 ± 0.03	0.945 ± 0.00	0.520 ± 0.05	0.670 ± 0.04	0.654 ± 0.02
TF-IDF + ComplementNB	0.568 ± 0.05	0.605 ± 0.02	0.458 ± 0.03	0.942 ± 0.01	0.560 ± 0.06	0.701 ± 0.04	0.629 ± 0.03
TF-IDF + MultinomialNB	0.568 ± 0.05	0.605 ± 0.02	0.458 ± 0.03	0.942 ± 0.01	0.560 ± 0.06	0.701 ± 0.04	0.629 ± 0.03
StratifiedKFold							
BoW + LinearSVC	$\mathbf{0.613 \pm 0.12}$	$\mathbf{0.699 \pm 0.05}$	$\mathbf{0.504 \pm 0.08}$	$\mathbf{0.969 \pm 0.01}$	0.594 ± 0.15	$\mathbf{0.727 \pm 0.12}$	$\mathbf{0.692 \pm 0.05}$
BoW + LogisticRegression	0.526 ± 0.05	0.496 ± 0.04	0.409 ± 0.03	0.908 ± 0.01	0.532 ± 0.06	0.669 ± 0.05	0.472 ± 0.03
BoW + ComplementNB	0.584 ± 0.33	0.488 ± 0.02	0.375 ± 0.15	0.898 ± 0.03	$\mathbf{0.605 \pm 0.41}$	0.648 ± 0.36	0.496 ± 0.03
BoW + MultinomialNB	0.584 ± 0.33	0.488 ± 0.02	0.375 ± 0.15	0.898 ± 0.03	$\mathbf{0.605 \pm 0.41}$	0.648 ± 0.36	0.496 ± 0.03
TF-IDF + LinearSVC	0.571 ± 0.03	0.589 ± 0.01	0.455 ± 0.01	0.936 ± 0.01	0.567 ± 0.04	0.705 ± 0.03	0.614 ± 0.01
TF-IDF + LogisticRegression	0.529 ± 0.05	0.527 ± 0.01	0.419 ± 0.02	0.918 ± 0.00	0.529 ± 0.06	0.670 ± 0.05	0.549 ± 0.02
TF-IDF + ComplementNB	0.547 ± 0.10	0.527 ± 0.03	0.423 ± 0.04	0.919 ± 0.01	0.552 ± 0.13	0.682 ± 0.10	0.524 ± 0.03
TF-IDF + MultinomialNB	0.547 ± 0.10	0.527 ± 0.03	0.423 ± 0.04	0.919 ± 0.01	0.552 ± 0.13	0.682 ± 0.10	0.524 ± 0.03

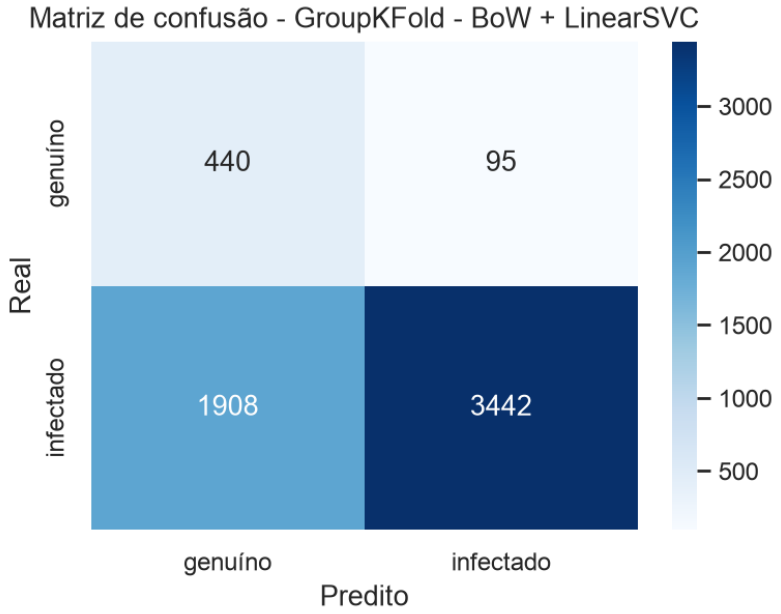

Figura 6. Matriz de confusão do modelo *BoW + MultinomialNB* com validação GroupKFold.

Conforme ilustra a matriz de confusão da Figura 6, o balanceamento permitiu que o modelo produzisse classificações em ambas as classes, evitando a solução trivial de rotular todas as amostras como infectadas (tomando proveito do natural desbalanceamento dessas categorias). Os separadores lineares, sobretudo o LinearSVC, apresentaram

o comportamento mais consistente quando consideradas conjuntamente acurácia balanceada, F1 e ROC-AUC. Os modelos Naive Bayes mantiveram *recall* elevado em alguns cenários, mas com maior variabilidade e menor capacidade de discriminação balanceada. Portanto, embora o *recall* permaneça prioritário devido ao custo dos falsos negativos, ele deve ser analisado com precisão, acurácia balanceada e ROC-AUC para evitar detectores que favoreçam excessivamente uma das classes.

Ademais, esses classificadores se baseiam em representações textuais simples, como Bag-of-Words e TF-IDF, que capturam tokens e bigramas do QASM, mas não modelam a semântica, as dependências do circuito ou o efeito físico das operações. Os resultados demonstram sinais textuais úteis no dataset avaliado, porém não há garantia de que as métricas permaneçam equivalentes caso passes adicionais de otimização sejam aplicados. Tais passes podem decompor, substituir, reordenar ou eliminar operações, modificando os padrões usados pelos modelos e exigindo nova avaliação ou treinamento. Isso reforça a necessidade de detectores que combinem informações sintáticas, estruturais e comportamentais dos circuitos.

4. Conclusão

Este trabalho apresentou um dataset de *Quantum Trojans* e análises de sensibilidade e detecção em circuitos quânticos. O modelo adversarial representa uma hipótese de ataque à cadeia de suprimentos, na qual um passe malicioso integrado ao Qiskit insere operações durante a compilação. Embora passes posteriores possam reescrever e mascarar essas inserções, o dataset utiliza portas simples como abstração controlada e não quantifica diretamente a furtividade após uma transpilação completa.

As inserções produziram impactos comportamentais distintos. Portas X e Y causaram as maiores variações de TVD, enquanto portas de fase tenderam a apresentar efeitos menores. A análise SHAP indicou que a porta inserida é o principal fator associado à TVD média, seguida por características do circuito, como quantidade de bits clássicos, família e posição da inserção.

Na detecção textual, o modelo *BoW + LinearSVC* apresentou o desempenho mais equilibrado no *GroupKFold*, com $66{,}0\%$ de acurácia e $73{,}3\%$ de acurácia balanceada, enquanto o *TF-IDF + LinearSVC* obteve a maior ROC-AUC. Os modelos Naive Bayes alcançaram *recall* elevado em alguns cenários, mas com maior variabilidade e menor discriminação balanceada. Assim, o *recall* deve ser considerado com F1-score, acurácia balanceada e ROC-AUC, pois seu valor isolado não caracteriza um detector útil.

Cabe ressaltar que os resultados se restringem às representações QASM avaliadas, visto que passes de otimização adicionais podem modificar os padrões textuais e alterar o desempenho dos classificadores. Como trabalhos futuros, pretende-se avaliar circuitos após diferentes fluxos de otimização e *backends*, ampliar estratégias de inserção e estados iniciais, e investigar representações estruturais e semânticas, como grafos e DAGs. A disponibilização do dataset busca apoiar essas comparações e o desenvolvimento de mecanismos de segurança mais robustos para compiladores quânticos.

Referências

Bolgar, C. (2025). Microsoft's majorana 1 chip carves new path for quantum computing. Acessado em: 28 de fevereiro de 2025.

Chen, K., Fang, W., Guan, J., Hong, X., Huang, M., Liu, J., Wang, Q., and Ying, M. (2022). VeriQBench: A benchmark for multiple types of quantum circuits.

Cicero, A., Maleki, M. A., Azhar, M. W., Kockum, A. F., and Trancoso, P. (2025). Simulation of quantum computers: Review and acceleration opportunities. *ACM Transactions on Quantum Computing*, 7(1):1–35.

Das, S. and Ghosh, S. (2023). Trojannet: Detecting trojans in quantum circuits using machine learning. *arXiv preprint arXiv:2306.16701*.

Das, S. and Ghosh, S. (2024). Trojan taxonomy in quantum computing. In *2024 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 644–649. IEEE.

Javadi-Abhari, A. et al. (2024). Quantum computing with qiskit. *arXiv preprint arXiv:2405.08810*.

John, J., Golla, L., and Wang, Q. (2025). Stealthy conditional trojans in quantum circuits. In *2025 26th International Symposium on Quality Electronic Design (ISQED)*, pages 1–7. IEEE.

Kharkov, Y., Ivanova, A., Mikhantiev, E., and Kotelnikov, A. (2022). Arline benchmarks: Automated benchmarking platform for quantum compilers. *arXiv preprint arXiv:2202.14025*.

Li, A., Stein, S., Krishnamoorthy, S., and Ang, J. (2022). QASMBench: A low-level quantum benchmark suite for nisq evaluation and simulation. *ACM Transactions on Quantum Computing*.

Neven, H. (2024). Meet willow, our state-of-the-art quantum chip. Acessado em: 28 de fevereiro de 2025.

Nielsen, M. A. and Chuang, I. L. (2001). *Quantum computation and quantum information*, volume 2. Cambridge university press Cambridge.

UNESCO (2025). International year of quantum science and technology 2025. Acessado em: 28 de fevereiro de 2025.

Wille, R., Große, D., Teuber, L., Dueck, G. W., and Drechsler, R. (2008). RevLib: An online resource for reversible functions and reversible circuits. In *Int'l Symp. on Multi-Valued Logic*, pages 220–225. RevLib is available at http://www.revlib.org.