Universidad Internacional de La Rioja
Escuela Superior de Ingeniería y Tecnología

Máster Universitario en Ciberseguridad

Piloto Experimental del Algoritmo Xander para la Detección de Intrusiones en red utilizando la Computación Cuántica Híbrida

Trabajo fin de estudio presentado por:	Oscar Javier Peñaloza Araque
Tipo de trabajo:	Piloto Experimental
Director/a:	Sergio Mauricio Martínez Monterrubio. PhD
Fecha:	11 de febrero del 2026

Agradecimientos

Quiero expresar mi más profundo agradecimiento a la Universidad Internacional de La Rioja UNIR y al profesorado del Máster en Ciberseguridad; por su guía, sus enseñanzas y el rigor académico que han acompañado en todo lo largo de este proceso. Asimismo, al Doctor; Sergio Mauricio Martinez Monterrubio, por su orientación, observaciones y constante disposición para encauzar este trabajo hacia un aporte significativo en el campo de la ciberseguridad y la computación cuántica. Finalmente, extiendo un especial reconocimiento a mi esposa e hijos, cuyo apoyo incondicional, paciencia y motivación permanente hicieron posible la culminación de este proyecto.

Resumen

Este Trabajo Fin de Máster presenta un piloto experimental para la detección de intrusiones en red mediante el algoritmo Xander, el cual representa una arquitectura híbrida cuántico-clásica, que integra modelos tales como; la Máquina de Vectores de Soporte Cuántico QSVM, el Circuito Cuántico Variacional VQC, y la Red Neuronal Convolucional Cuántica QCNN, en un ensamble con clasificadores tradicionales. El estudio utiliza el conjunto de datos CIC-IDS-2017 y desarrolla una metodología reproducible que comprende preprocesamiento, reducción de dimensionalidad y evaluación del sistema en tres niveles: Estimator local sin ruido, IBM Quantum Runtime Aer Simulator con aceleración clásica y ejecución en hardware cuántico real en entorno NISQ. Los resultados muestran métricas superiores al 99% en accuracy y F1-score, demostrando la viabilidad operativa del enfoque híbrido y la contribución complementaria del componente cuántico dentro del ensamble. Se concluye que Xander es una alternativa prometedora para mejorar la detección de intrusiones, especialmente en escenarios donde la robustez y la trazabilidad son críticas.

Palabras clave: Computación cuántica híbrida; Detección de intrusiones; IDS; Quantum Machine Learning; Ensamble cuántico-clásico.

Abstract

This Master's Thesis presents an experimental pilot for network intrusion detection using the Xander algorithm, which represents a hybrid quantum-classical architecture that integrates models such as the Quantum Support Vector Machine QSVM, the Variational Quantum Circuit VQC, and the Quantum Convolutional Neural Network QCNN in an ensemble with traditional classifiers. The study uses the CIC-IDS-2017 dataset and develops a reproducible methodology that includes preprocessing, dimensionality reduction, and system evaluation at three levels: local noise-free estimator, IBM Quantum Runtime Aer Simulator with classical acceleration, and execution on real quantum hardware in a NISQ environment. The results show metrics above 99% in accuracy and F1-score, demonstrating the operational viability of the hybrid approach and the complementary contribution of the quantum component within the ensemble. It is concluded that Xander is a promising alternative for improving intrusion detection, especially in scenarios where robustness and traceability are critical.

Keywords: Hybrid quantum computing; Intrusion detection; IDS; Quantum Machine Learning; Quantum-classical ensemble.

Índice de contenidos

Capítulo 1: Fundamentos de la Investigación 1

1. Introducción 1

1.1. Motivación 2

1.1.1. Problema 2

1.1.2. Causa principal 3

1.1.3. Importancia 3

1.2. Planteamiento del problema 4

1.3. Estructura del trabajo 5

1.4. Hipótesis 7

1.4.1. Hipótesis nula (H_o) 7

1.4.2. Hipótesis alternativa (H_a) 7

1.5. Preguntas de Investigación 7

1.6. Objetivos 8

1.6.1. Objetivo general 8

1.6.2. Objetivos específicos 8

Capítulo 2: Estado del arte 10

2. Introducción 10

2.1. Ciberseguridad y detección de intrusos 11

2.2. Aprendizaje automático en detección de intrusos 16

2.3. Computación cuántica y aprendizaje automático cuántico 19

2.4. Modelos híbridos cuántico-clásicos 22

2.5. Estudios previos en IDS cuánticos 23

2.5.1. Ensamble y Votación en Modelos Híbridos QSVM, VQC, QCNN 25

2.6. Síntesis crítica y posicionamiento de la propuesta 29

2.6.1. Divergencia cuántica y matemática 31

2.7. Síntesis del estado del arte 32

Capítulo 3: El Algoritmo Xander 34

3. Introducción 34

3.1. Descripción detallada del **Algoritmo Xander** 35

Capítulo 4: Metodología del Algoritmo Xander 39

4. Introducción 39

4.1. Metodología de ejecución del piloto experimental del Algoritmo Xander 40

4.1.1. Fase 1 - Carga de datos 42

4.1.2. Fase 2 - Preprocesamiento de datos 42

4.1.3. Fase 3 - Reducción de dimensionalidad 45

4.1.4. Fase 4 - Entrenamiento cuántico 46

4.1.5. Fase 5 - Entrenamiento clásico 50

4.1.6. Fase 6 - Ensamblado híbrido cuántico–clásico 52

4.1.7. Fase 7 - Evaluación, resultados e informes 53

Capítulo 5: Experimentación 57

5. Descripción de los resultados 57

5.1. Como transcurrió el piloto experimental 57

5.2. Instrumentos de seguimiento y evaluación 57

5.3. Análisis estadístico aplicado 58

5.4. Descripción de los resultados con la ejecución en estimator local de qiskit GPU 59

5.4.1. Resultados obtenidos en la ejecución con Estimator local de Qiskit GPU 59

5.5. Descripción de los resultados obtenidos en la ejecución con IBM Quantum 64

5.5.1. Resultados obtenidos en IBM Quantum Runtime Aer simulator con la activación del disparador de los recursos alternativos clásicos 64

5.5.2. Resultados obtenidos en la ejecución con el procesador IBM Quantum Heron en el sistema cuántico ibm_torino 67

5.5.3. Comprobaciones finales 73

Capítulo 6: Análisis de la experimentación 74

6. Análisis de los resultados 74

6.1. Relevancia de los resultados 74

6.2. Resultados relevantes con qiskit estimator Local. **GPU** 74

6.3. Resultados con ibm quantum runtime aer simulator con retorno a recursos clásicos acelerados. **IBM** 75

6.4. Resultados en hardware cuántico real y su papel dentro del ensamble 77

6.5. Datos anómalos y su explicación 79

6.6. Resultado final del análisis 79

Capítulo 7: Consideraciones Finales 80

7. Conclusiones y Trabajo futuro 80

7.1. Conclusiones 80

7.2. Trabajo futuro 83

Referencias bibliográficas 85

Anexo A. 91

Flujo - híbrido_heron_pipeline_v2 91

Anexo B. 94

Artículo científico 94

Índice de figuras

Figura 1. *IDS categóricos basados en firmas.* 12

Figura 2. *IDS categóricos basados en anomalías.* 12

Figura 3. *Comparación de métricas específicas de porcentajes de error y perdida de rendimiento.* 15

Figura 4. *Diagrama de flujo para la metodología de aplicabilidad del aprendizaje automático en la detección de intrusos.* 19

Figura 5. *Fases de ejecución del piloto experimental del Algoritmo Xander.* 40

Figura 6. *Flujograma de ejecución del piloto experimental del Algoritmo Xander 1.* 40

Figura 7. *Flujograma de ejecución del piloto experimental del Algoritmo Xander 2.* 40

Figura 8. *Flujograma de ejecución del piloto experimental del Algoritmo Xander 3.* 41

Figura 9. *Bloque para carga y procesamiento de datos.* 43

Figura 10. *Bloque de balanceo.* 43

Figura 11. *Bloque PCA + escalado.* 44

Figura 12. *Bloque para codificación de etiquetas.* 44

Figura 13. *Bloque para reducción de dimensionalidad.* 45

Figura 14. *Constructor de mapas de características.* 47

Figura 15. *Bloque constructor de conexión para el backend.* 47

Figura 16. *Bloque de entrenamiento Máquina de Vectores de Soporte Cuántico QSVM.* 48

Figura 17. *Bloque constructor del Circuito de la Red Neuronal Cuántica QNN.* 48

Figura 18. *Bloque de entrenamiento Circuito Cuántico Variacional VQC.* 49

Figura 19. *Bloque de entrenamiento Red Neuronal Convolucional Cuántica QCNN.* 49

Figura 20. *Bloque de entrenamiento para Clasificador de Vectores de Soporte SVC.* 50

Figura 21. *Bloque de entrenamiento para el Clasificador de bosque aleatorio RFC.* 51

Figura 22. *Bloque de entrenamiento para el Clasificador Perceptrón Multicapa MLPClassifier.* 51

Figura 23. *Bloque de entrenamiento de la biblioteca optimizada de impulso de árboles de decisión Xgboost.* 51

Figura 24. *Bloque para la exploración y evaluación de pesos.* 52

Figura 25. *Bloque estratégico del ensamblado.* 52

Figura 26. *Bloque para la aplicación de pesos.* 53

Figura 27. *Bloque para ajuste del Umbral.* 53

Figura 28. *Bloques de evaluación y diagramas.* 54

Figura 29. *Bloques para el almacenamiento de información.* 55

Figura 30. *Bloque para la monitorización.* 55

Figura 31. *Comparativa de métricas por modelo 1.* 60

Figura 32. *Comparativa de métricas por modelo 2.* 61

Figura 33. *Matriz de confusión del ensamble.* 61

Figura 34. *Curvas ROC – características operativas* 62

Figura 35. *Gráfica de robustez al ruido.* 63

Figura 36. *Gráfica de tiempos de entrenamiento.* 63

Figura 37. *Registro de seguimiento con la activación del disparador de recursos alternativos clásicos.* 65

Figura 38. *Tiempos de ejecución durante las etapas del proceso con la activación del disparador de recursos alternativos clásicos.* 65

Figura 39. *Comparativa de métricas por modelo con la activación del disparador de recursos alternativos clásicos.* 66

Figura 40. *Recursos computacionales Plan Open IBM.* 67

Figura 41. *Cargas de trabajo ejecutadas Plan open IBM.* 67

Figura 42. *Circuito cuántico.* 69

Figura 43. *Ruido del backend IBM sistema cuántico ibm_torino.* 69

Figura 44. *Curva ROC.* 70

Figura 45. *Matriz de confusión.* 70

Figura 46. *Probabilidades del ensamble (SVM, RF, QSVM).* 71

Figura 47. *Carga cuántica ejecutada.* 72

Figura 48. *Comparativa de métricas por modelo 1.* 75

Figura 49. *Comparativa de métricas por modelo con la activación del disparador de recursos alternativos clásicos.* 76

Figura 50. *Ruido del backend IBM sistema cuántico ibm_torino.* 77

Figura 51. *Curva ROC.* 78

Figura 52. *Registro de ejecución del pipeline hybrid_heron_pipeline_v2 .* 92

Figura 53. *Tiempos de ejecución Plan Open .* 92

Índice de tablas

Tabla 1. *Estadísticas de porcentajes de error y pérdida de rendimiento.* 15

Tabla 2. *Comparativa de Modelos de Aprendizaje Automático Cuántico QML.* 21

Tabla 3. *Comparativa de métricas relevantes.* 22

Tabla 4. *Comparativa de estudios previos.* 24

Tabla 5. *Ventajas y desventajas de los competidores cuánticos híbridos.* 26

Tabla 6. *Comparativa de los conjuntos de datos.* 27

Tabla 7. *Vacíos metodológicos en la literatura.* 30

Tabla 8. *Conjunto tecnológico de recursos utilizados en el piloto experimental del Algoritmo Xander.* 35

Tabla 9. *Ficha técnica del piloto experimental del Algoritmo Xander.* 37

Tabla 10. *Descripción de las fases de ejecución del piloto experimental del Algoritmo Xander.* 41

Tabla 11. *Mapeo de fases del piloto experimental del Algoritmo Xander versus ubicación en el código.* 55

Tabla 12. *Instrumentos de seguimiento y evaluación.* 58

Tabla 13. *Resumen del desempeño en Estimator local de Qiskit.* 59

Tabla 14. *Tiempos de ejecución con la activación del disparador de recursos alternativos clásicos.* 65

Tabla 15. *Métricas obtenidas del lanzamiento con la activación del disparador de recursos alternativos clásicos.* 66

Tabla 16. *Comparativa de ajustes para el lanzamiento en ibm_torino.* 68

Tabla 17. *Ajustes para el lanzamiento en el sistema cuántico ibm_torino.* 68

Tabla 18. *Datos por clase verdaderos vs falsos.* 71

Tabla 19. *Reporte de clasificación del ensamble.* 72

Capítulo 1: Fundamentos de la Investigación

1. Introducción

El enriquecimiento apalancado que ha estado recibiendo todo el tópico en el enfoque de la transformación digital con el pasar de los años, es el que ha generado ese desencadenamiento hacia la brecha de la superficie de exposición a las ciber-amenazas, en las redes empresariales y en todas aquellas infraestructuras de carácter crítico y en todos aquellos entornos con base en la nube, a lo que se estima, se hallan ampliado aún más esta exposición de lo que se tenía pronosticado. Es por ello, que en todos estos escenarios se refleja la posible clave permeable, y hago alusión a los *Sistemas de Detección de Intrusos IDS*, los cuales se han vuelto mucho más impredecibles, pero por ende, es de aclarar que aquellos enfoques tradicionales los cuales se encuentran basados en el aprendizaje automático o firmas, son los que a menudo degradan su respuesta al rendimiento, esto debido a ciber-ataques inimaginables y desconocidos, al tráfico de red y a la distribución de datos en estados oscilantes.

De frente a este panorama que es el que nos aqueja, los IDS, se proyectan como ese punto esencial en esa conformación de estrategias, para la puesta en marcha de la defensa de la información. Es así, como, el enfoque de la computación cuántica, dentro de un campo de estudio híbrido cuántico-clásico, sobresale dentro de las alternativas con las cuales podremos experimentar con el fin de dar solución, o allegar a una solución para todos aquellos problemas que se encuentran por encima de aquellas capacidades que conviven dentro de la computación tradicional, como los son: la búsqueda, la clasificación y la optimización. Es por este esclarecimiento, que se merece su profunda exploración, la cual se encuentra totalmente representada en las fronteras de aplicación al área de la ciberseguridad.

Este Trabajo Fin de Máster. Propone y valida un piloto experimental de detección de intrusiones basadas en red y sustentado en la computación cuántica híbrida; un flujo, donde los módulos tradicionales como lo son: el preprocesamiento, la reducción de dimensionalidad y la orquestación, se acoplan de lleno con los siguientes bloques cuánticos: la codificación, la representación vectorial de datos y los modelos variacionales cuánticos asistidos, todo esto,

con el único fin de mejorar la capacidad de detección de intrusos frente a los estándares tradicionales.

En esta memoria, se presenta como primera medida, el estado del arte en el aprendizaje automático aplicado a los sistemas de detección de intrusos IDS y en la proyección del Aprendizaje Automático Cuántico QML que al inglés se traduce *Quantum Machine Learning QML*; después, una metodología reproducible sobre conjuntos de datos de tráfico de red basados en el conjunto de datos: *CIC IDS-2017*, el cual incluye la limpieza, el balanceo, el particionado y las métricas estándar como: la exactitud, la precisión y el *recall*; este último como encargado de medir la capacidad que puede llegar a tener el modelo en la identificación adecuada de todos los objetos relevantes que pueden haber dentro de un conjunto de datos y por último la F1 score; la encargada de medir el rendimiento de los modelos de clasificación, a continuación, se describe la arquitectura híbrida propuesta y el protocolo experimental comparativo con los modelos clásicos, y finalmente, se discuten las posibles ventajas y limitaciones del enfoque en la era *NISQ*, Cuántico Intermedio a Escala Ruidosa que sus siglas traducidas al inglés corresponderían a *Noisy Intermediate Scale Quantum* y su aplicabilidad potencial en escenarios operativos.

Con ello, se busca ofrecer una contribución práctica; dentro de un piloto experimental y una contribución analítica; y una comparación crítica entre enfoques clásicos y cuántico-híbridos para IDS, sentando bases para nuevas investigaciones y despliegues futuros.

1.1. Motivación

1.1.1. Problema

Los sistemas que realizan medidas de detección de intrusiones con un enfoque tradicional tienden a presentar dificultades para adaptarse a los nuevos tipos de ataques o a ataques emergentes, al tráfico cifrado y a los patrones comportamentales que de por sí mismos, no siguen una línea clara y especifica. En sí, esto puede llegar a resultar en un alto número de falsos positivos y falsos negativos, además de una reducción en la visibilidad de los patrones. Es de aclarar, que estos problemas no solo comprometen y exponen a la seguridad técnica, sino que de una u otra manera también generan pérdidas económicas, que definitivamente

conllevan a la afectación de la confianza por parte de los clientes y lo que, por otra parte, dificulta el cumplimiento de las normativas tales como: el GDPR, el NIST CSF y la ISO/IEC 27001.

En el caso de los estudios de Kalinin & Krundyshev. (2023), se hace un señalamiento hacia los modelos tradicionales, donde se infiere que estos modelos clásicos generan un deterioro altamente significativo en cuanto a métricas de precisión y a la velocidad cuando enfrentan un tráfico masivo o cifrado, lo que por ende los vuelve ineficaces ante los ataques distribuidos o los ataques de baja frecuencia.

1.1.2. Causa principal

Por lo que respecta a la principal causa, se puede decir que esta, se encuentra ligada a los avances tecnológicos, al desarrollo disruptivo, y a las actividades que reflejan el interior del tráfico de red, así mismo la representación del alto costo computacional, lo que, en sí, significa la representación en los espacios de alta envergadura, dando uso únicamente a los métodos clásicos-tradicionales; de la misma manera, Abreu et al. (2024), hacen énfasis en que estos modelos clásicos-tradicionales necesitan de recursos computacionales muy elevados para tratar de lograr mantener métricas como la precisión en aquellos entornos multicapa, mientras que, por su parte, los modelos cuánticos pudiesen llegar a representar a aquellos patrones mucho más complejos y con una menor profundidad del circuito.

1.1.3. Importancia

De igual manera, podemos observar que esta problemática refleja su impacto de una forma clara, directa y dirigida hacia el sustento de las empresas y organizaciones, en el cumplimiento de las regulaciones, en el salvaguardar la información y en la capacidad de recuperar los servicios críticos. Dentro de este marco, resulta claro y preciso dar pie a la investigación de aquellos modelos híbridos que logran combinar lo cuántico con lo clásico-tradicional; dado que, la superposición y el entrelazado hablando en términos cuánticos, podrían llegar a permitir descripciones más detalladas de esos flujos dentro del conjunto de datos a estudiar, mejorando así la capacidad de moldeamiento y la identificación de irregularidades.

Es por ello, que, Abreu et al. (2024), en el artículo titulado *Quantum Machine Learning Intrusión Detection System* que en español significa Sistema de Detección de Intrusiones con

Aprendizaje Automático Cuántico, logra demostrar; que, los modelos híbridos cuántico-clásicos pueden llegar a superar en gran manera a los clasificadores clásicos-tradicionales en aquellas tareas de detección de multi-categoría, con mejoras en métricas como la precisión y la reducción de falsos positivos, incluso dentro de los entornos NISQ.

1.2. Planteamiento del problema

Dentro de este planteamiento, el proceso para la identificación de intrusiones es de suma importancia, para la contención y el aseguramiento de las infraestructuras críticas y los sistemas de información; los métodos tradicionales de detección, que utilizan las técnicas de aprendizaje automático y el aprendizaje profundo, han conseguido avances significativos en cuanto a la precisión y a la capacidad de adaptación. No obstante, tienen debilidades notables en los escenarios dinámicos como el Internet de las Cosas, las redes 5G o las redes definidas por software, donde el tráfico es variado, no constante y muy abundante. Estas debilidades incluyen: las altas tasas de falsos positivos, el requerimiento de considerable poder de procesamiento y los problemas para generalizar ante los ataques nuevos o del tipo de ataque Zero-day.

Es por eso, que, de manera conjunta, los modelos que aplican a la computación cuántica surgen como esa opción novedosa e interesante para darle un mejoramiento al tratamiento de datos en las magnitudes superiores, de tal forma, ofreciendo potenciales beneficios en lo que compete a la clasificación y la optimización. Modelos tales como: la Máquina de Vectores de Soporte Cuántico en sus siglas en ingles QSVM, el Circuito Cuántico Variacional en sus siglas en ingles VQC y la Red Neuronal Convolucional Cuántica en sus siglas en ingles QCNN, han sido probados en el campo de la detección de intrusos, mostrando grandes mejoras en precisión y la disminución de falsos positivos en las pruebas simuladas. Sin embargo, la mayoría de estos estudios se han realizado en los entornos controlados, sin establecer un marco reproducible y comparativo con respecto a los clasificadores tradicionales utilizando los conjuntos de datos de referencia como el CIC-IDS-2017.

Además, como mencionan Abreu et al. (2024) y Gong et al. (2022), aquellos modelos los cuales se encuentran basados en principios cuánticos han demostrado tener enormes ventajas

teóricas en cuanto a lo que compete a la recuperación de la información y al rendimiento computacional, sin embargo, no han sido validados en hardware real, ni han sido integrados funcionalmente en sistemas híbridos. Esta falta de metodología dificulta la evaluación de si las ventajas de la computación cuántica se mantienen en situaciones técnicas reales, donde son cruciales la trazabilidad, el registro de datos y la capacidad de adaptación incremental para una detección eficaz.

Por tanto, el principal desafío radica en la ausencia de un estudio comparativo exhaustivo y sistemático entre los modelos de detección de intrusiones tradicionales y los cuánticos, lo cual permitiría establecer, con evidencia observacional, si los métodos híbridos que combinan lo cuántico y lo clásico ofrecen verdaderas ventajas en cuanto a exactitud, eficacia y flexibilidad, tomando en cuenta las limitaciones de los sistemas actuales. Esta necesidad se vuelve aún más apremiante ante la falta de proyectos piloto que integren la ejecución en hardware cuántico, un conjunto de clasificadores y la trazabilidad visual, como el que se plantea en esta investigación.

1.3. Estructura del trabajo

- **Capítulo 1 - Introducción**

  Presenta el problema, la motivación, los objetivos y el panorama del trabajo.

- **Capítulo 2 - Estado del arte**

  En esta sección se examina primero el panorama actual de los sistemas de detección de intrusos IDS, diferenciando entre los métodos que se basan en firmas y aquellos que se fundamentan en anomalías. También se abordan las limitaciones tanto prácticas como metodológicas de los enfoques tradicionales de aprendizaje automático cuando se aplican a grandes volúmenes de datos y a entornos variables. Luego, se presenta la computación cuántica y sus aplicaciones emergentes en el aprendizaje automático QML, con un enfoque particular en los modelos relevantes para IDS como lo son: QSVM, VQC y QCNN, además de las restricciones que plantea la era NISQ.

Para concluir, se resumen investigaciones experimentales que implementan QML en problemas de detección de intrusos y se destacan las principales lagunas en el conocimiento que este Trabajo Fin de Máster busca resolver: elección y evaluación de métodos de codificación, reproducibilidad entre simuladores y hardware real, así como el desarrollo de protocolos de comparación sólidos sobre el conjunto de datos CIC-IDS 2017. Se incluye también una revisión crítica de artículos.

- **Capítulo 3 - Desarrollo específico de la contribución**

  Piloto experimental del Algoritmo Xander: Descripción y detalles de toda la caracterización del piloto experimental, la configuración, tecnologías, técnicas utilizadas, desarrollo del entrenamiento, instrumentos de seguimiento y de la validación y evaluación.

- **Capítulo 4 - Metodología**
  a. Piloto experimental del Algoritmo Xander: Descripción del flujo sobre el conjunto de datos CIC-IDS-2017, acerca del preprocesamiento, la ingeniería de características, la reducción de dimensionalidad, la arquitectura híbrida, el lanzamiento del entrenamiento y la validación y análisis de los resultados.
  b. Descripción de los Resultados: Comparativa cuantitativa con los estándares clásicos y cuánticos híbridos, Exposición objetiva de resultados.
  c. Discusión: Análisis de métricas, la robustez y las consideraciones prácticas y la relevancia de los resultados.

- **Capítulo - 5 y 6 experimentaciones y análisis de resultados**
- **Capitulo - 7 Consideraciones finales - Conclusiones y trabajos futuros**

Conclusiones y trabajos futuros: Síntesis de hallazgos, limitaciones, líneas de investigación y mejora de técnicas y de despliegue.

1.4. Hipótesis

1.4.1. Hipótesis nula (H_0)

Los enfoques híbridos que combinan lo cuántico y lo clásico como: QSVM y VQC reflejan algunas mejoras en términos estadísticos, cuando se comparan con los métodos tradicionales de aprendizaje automático como: Random Forest, XGBoost, SVM, Redes Neuronales en la identificación de intrusiones, en cuanto a métricas de exactitud, recall y reducción en la cantidad de falsos positivos al ser probados con el conjunto de datos CIC-IDS-2017. Toda vez, que han sido entrenados y evaluados de forma individual.

1.4.2. Hipótesis alternativa (H_a)

El algoritmo Xander, basado en computación cuántica híbrida QSVM, VQC, QCNN en VOTING y ajuste dinámico, como aplicación novedosa y combinación de técnicas existentes en ensamble, demuestra un rendimiento superior en la tarea de detección de intrusos, igualando e incluso superando tanto a algunos de sus rivales clásicos-tradicionales, como a otros de enfoques híbridos cuántico-clásicos como: QSVM y VQC, específicamente en las métricas de precisión, recall y accuracy. Esta ventaja se evidencia al ser evaluado con el conjunto de datos CIC-IDS-2017, donde se logra identificar amenazas en red con mayor exactitud y menor margen de error.

1.5. Preguntas de Investigación

Surge la necesidad de investigar el verdadero potencial que pueden llegar a ofrecer los algoritmos computacionales cuánticos dentro de un campo híbrido, proyectados hacia las detecciones intrusivas en la red. Asimismo, se generan interrogantes de lo que se pretende evaluar, la viabilidad técnica de los enfoques, y la contribución al avance dentro del conocimiento clásico y cuántico, vistos desde el entrelazamiento de la ciberseguridad y la cuántica computacional.

Siendo así, y estando dentro de un laberinto inmerso de espejismos me cuestiono y doy cuestionamiento a los requerimientos que surgen de dicha necesidad tales como: ¿Cuáles son las principales limitaciones de los modelos clásicos de aprendizaje automático aplicados a la

detección de intrusiones en la era NISQ al implementar IDS híbridos?, ¿En qué medida los modelos cuánticos e híbridos QSVM, VQC mejoran las métricas de detección (precisión, recall, F1-score, tasa de falsos positivos) frente a modelos clásicos, cuando se aplican a conjuntos de datos de referencia como CIC-IDS-2017?, ¿Qué ventajas o desventajas prácticas presentan los enfoques cuántico–clásicos respecto a su implementación en hardware cuántico de la era NISQ?, ¿Es posible diseñar un flujo reproducible de detección de intrusiones que integre reducción de dimensionalidad PCA, clasificación cuántica y evaluación estandarizada para validar experimentalmente la viabilidad de modelos híbridos en ciberseguridad?, Y finalmente, ¿Qué mecanismos de trazabilidad visual y validación cruzada pueden integrarse en modelos híbridos cuántico-clásicos para lanzar la auditoria?.

1.6. Objetivos

1.6.1. Objetivo general

Diseñar y evaluar un piloto experimental de detección de intrusiones basadas en red mediante el uso de computación cuántica híbrida, integrando la ejecución en simuladores cuánticos de alto rendimiento y complementando en hardware real cuántico, con una trazabilidad visual y lógica de ensamble, y su posterior comparación frente a enfoques clásicos-tradicionales y otros de enfoques cuántico hibrido, en términos de rendimiento, eficiencia y viabilidad operativa en entornos NISQ.

1.6.2. Objetivos específicos

1. Revisar el estado del arte sobre los sistemas de detección de intrusos clásicos y los modelos de quantum machine learning aplicados a la ciberseguridad, identificando brechas metodológicas como la falta de ejecución en hardware cuántico, ausencia de trazabilidad visual y escasa reproducibilidad operativa.
2. Diseñar un flujo reproducible de preprocesamiento en cuanto a la limpieza, la normalización, el balanceo, la reducción de dimensionalidad PCA y el particionado sobre el conjunto de datos CIC-IDS-2017, con trazabilidad técnica y codificación visual jerárquica.

3. Implementar una arquitectura híbrida cuántico-clásica en ensamble, con la integración de bloques de representación vectorial, circuitos variacionales como: los QCNN, los QSVM y los VQC asistidos, los clasificadores y optimizadores. Todos estos acoplados a componentes clásicos mediante lógica modular y decoradores por entorno.
4. Comparar el rendimiento de los modelos híbridos cuántico-clásicos de frente a los clasificadores clásicos tales como: Clasificador de bosque aleatorio ó Random Forest, la biblioteca optimizada de impulso de árboles de decisión ó XGBoost, el Clasificador Perceptrón Multicapa ó SVM y Redes Neuronales, y por el otro lado, aquellos modelos de enfoque cuánticos híbridos tales como: la Máquina de Vectores de Soporte Cuántico ó QSVM y el Circuito Cuántico Variacional ó VQC, finalmente, haciendo uso de las métricas estándar tales como: la exactitud-accuarcy, la precisión, el recall, el F1-score y la matriz de confusión.
5. Examinar los pros y los contras del método híbrido, teniendo en cuenta el tiempo necesario para el entrenamiento, la consistencia en las operaciones, la respuesta a los hiperparámetros y la disminución de falsos positivos.
6. Analizar la viabilidad práctica del piloto experimental en situaciones reales, como en la supervisión de la red, la clasificación de las alertas y el diagnóstico progresivo, examinando su habilidad para fusionarse en procesos operativos con registros y verificación cruzada.

Capítulo 2: Estado del arte

2. Introducción

El presente capítulo reviste los principales antecedentes teóricos y empíricos que sustentan esta investigación sobre la detección de intrusiones mediante técnicas híbridas cuántico-clásicas. Su éxito se enfoca en la expresión de la percepción diversificada de ópticas materiales orientadas y enlazadas a correlacionales tales como: sistemas de detección de intrusos IDS, el aprendizaje automático tradicional ML, la computación cuántica y las aproximaciones híbridas QML, con la determinación de situar la propuesta dentro del contexto de la investigación actual que repercute e impacta directamente en la ciberseguridad.

En primer lugar, se examina el panorama actual de los sistemas de detección de intrusos IDS, diferenciando entre los métodos que se basan en firmas y aquellos que se fundamentan en anomalías. También se abordan las limitaciones tanto prácticas como metodológicas de los enfoques tradicionales de aprendizaje automático cuando se aplican a grandes volúmenes de datos y a entornos variables.

A continuación, exploramos y nos adentramos en la computación cuántica y sus utilidades en el aprendizaje automático QML, con un enfoque singular hacia los modelos más característicos dirigidos hacia los IDS, como son las *Quantum Support Vector Machines* QSVM que en español; significa Máquinas de Soporte Vectorial Cuánticas, el Clasificador Cuántico Variacional VQC y la *Quantum Convolutional Neural Network*, que en español significa; Red Neuronal Convolucional Cuántica QCNN. De igual forma, a todas aquellas limitaciones determinística del *Noisy Intermediate-Scale Quantum*, o Cuántico Ruidoso de Escala Intermedia NISQ.

Por último, se realiza una recopilación de investigaciones empíricas y actividades científicas las cuales han aplicado QML a la determinación para enfrentar dificultades contraídas en la detección de intrusiones. Aquí es donde se puede hacer un reconocimiento de las brechas caóticas que abordan el conocimiento que este Trabajo Fin de Máster busca aproximar: la selección y evaluación de esquemas de codificación, la reproducibilidad entre simuladores y

hardware real, así como el desarrollo de protocolos comparativos robustos y sólidos en torno al conjunto de datos CIC-IDS-2017.

2.1. Ciberseguridad y detección de intrusos

La ciberseguridad se constituye como un campo, el cual se encuentra en constantes cambios evolutivos, impulsados y motivados por el incremento en la sofisticación y complejidad de las amenazas y la criticidad e importancia de los sistemas de información en sectores estratégicos clave. Dentro de este ámbito, los Sistemas de Detección de Intrusos IDS se han consolidado como una herramienta esencial y parte fundamental para la identificación de conductas inusuales y/o comportamientos anómalos y actividades maliciosas perjudiciales para las redes y sistemas.

Se pueden distinguir dos grandes tipos o categorías de IDS:

Los **basados en firmas**, los cuales contrastan el tráfico contra patrones previamente ya conocidos, tal como se referencia en el *Handbook of Research on Machine and Deep Learning Applications for Cyber Security* que traducido al español significa; Manual de investigación sobre aplicaciones de aprendizaje automático y profundo para la ciberseguridad, en el capítulo titulado; "*Big Data Analytics for Intrusión Detection: An Overview*" que traducido al español significa; Análisis de Big Data para la Detección de Intrusiones: Una Visión General, donde se menciona claramente que el IDS utiliza las firmas para identificar ataques que ya son conocidos y las anomalías para detectar amenazas que no se han visto antes, aunque esto conlleva un costo más elevado en términos de falsas alarmas (Ganapathi & Shanmugapriya, 2019).

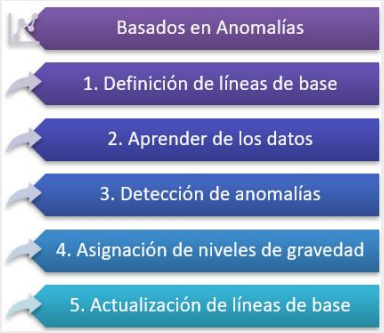
Figura 1. *IDS categóricos basados en firmas.*

Fuente: Elaboración Propia Basada en (Ganapathi & Shanmugapriya, 2019).

Las firmas de los ataques se reflejan en patrones distintivos asociados con comportamientos perjudiciales y los cuales son registrados en un repositorio. El sistema de monitoreo de intrusiones analiza de manera constante el tráfico de red que entra. Este sistema contrasta la actividad de red con las firmas de ataque que tiene almacenadas. Si hay una coincidencia, se emite una alerta para informar a los administradores sobre un potencial riesgo. Las herramientas de ciberseguridad implementan normas concretas que establecen cómo actuar ante una coincidencia de firmas detectadas. Las actualizaciones regulares aseguran la inclusión de nuevos métodos de ataque, lo que mantiene la efectividad del sistema.

De igual manera, encontramos a aquellos que se **fundamentan en anomalías**, los cuales identifican bifurcaciones y posteriormente detectan algunas discrepancias con respecto a las conductas inusuales y/o comportamientos que son y podría considerarse como habituales.

Figura 2. *IDS categóricos basados en anomalías.*



Fuente: Elaboración Propia Basada en (Ganapathi & Shanmugapriya, 2019).

En su función, podemos ver que establece referencias para el comportamiento habitual del sistema usando datos del pasado, Históricos, empleando estadísticas como el promedio y la variabilidad para definir lo que es normal, para que así los modelos de aprendizaje automático

examinen los datos históricos con el fin de crear y ajustar las referencias con los métodos de agrupamiento, como K medias que organizan patrones normales e identifican elementos fuera de lo común. Los datos nuevos se comparan con la referencia previamente establecida. Las variaciones de los patrones normales se destacan como posibles anomalías. La magnitud de la variación indica la gravedad de las alertas de anomalías. Y desviaciones mayores sugieren niveles de riesgo más elevados. Por lo cual, se deben realizar actualizaciones periódicas de las referencias con datos recientes para ajustarse a las nuevas tendencias. Los ajustes constantes aseguran que la detección de anomalías siga siendo efectiva.

Estos últimos han cobrado mayor relevancia en escenarios donde las amenazas emergentes no poseen o cuentan con firmas predefinidas. Es, por tanto, que los IDS denotan su evolución característica de enfoques basados en firmas hacia sistemas basados en anomalías, usando aprendizaje automático ML. Por ello, podemos inferir que el aprendizaje automático detecta e identifican desviaciones y diferencias respecto al comportamiento habitual, aunque enfrentan desafíos y presentan retos relacionados en cuanto a falsos positivos y escalabilidad (Pinto et al., 2023).

Dentro de este ámbito, los Sistemas de Detección de Intrusos IDS, se han consolidado como una herramienta esencial para la identificación de conductas anómalas. Revisiones exhaustivas como las de Hozouri et al. (2025) y Alotaibi & Rassam, (2023), quienes destacan que la efectividad de estos sistemas radica en su capacidad para adaptarse a una superficie de ataque en constante expansión, categorizando las técnicas actuales según su metodología de detección y despliegue.

De acuerdo con Vidal et al. (2020), los sistemas de detección de intrusiones que se fundamentan en la detección de anomalías se enfrentan a desafíos tales como el alto uso de energía, la antigüedad de los conjuntos de datos y la susceptibilidad a ataques maliciosos, lo que hace necesario desarrollar modelos que sean más flexibles y resistentes. En este sentido, Thakkar & Lohiya, (2020), enfatizan que la investigación detallada de los datasets es vital, ya que la calidad de la detección está intrínsecamente ligada y/o relacionada con la representatividad y actualidad de las muestras de tráfico utilizadas, ya que incluyen categorías desconocidas.

Por otra parte, en comparación con Noor et al. (2025), detallan que el aprendizaje automático clásico-tradicional es efectivo en situaciones constantes, aún en más, su rendimiento disminuye con el cambio de conceptos y la variabilidad de los entornos. Es así, que se sugieren la implementación del aprendizaje por transferencia y los enfoques híbridos como aquellas alternativas para potenciar la generalización y la capacidad de adaptación de los sistemas de detección de intrusiones en redes 5G y posteriores.

Asimismo, Agnew et al. (2024), muestran que los IDS convencionales carecen de la adaptabilidad requerida en redes inteligentes que utilizan *Software-Defined Networking* o Redes Definidas por Software, en sus siglas *SDN*. Por tanto, sugieren mezclar los métodos de aprendizaje automático y de seguridad anticipada para abordar riesgos tales como: el DoS, la suplantación de identidad y la alteración de los datos en sistemas fundamentales.

Cabe mencionar también que el Manual de Investigaciones sobre Aplicaciones de Aprendizaje Automático y Profundo para la Ciberseguridad. Ganapathi & Shanmugapriya (2019), proporciona un marco teórico sobre el desarrollo de los sistemas de detección de intrusos tradicionales, sus desventajas frente al tráfico encriptado y los ataques Zero-day, así como la urgencia de investigar enfoques diferentes como el aprendizaje automático cuántico.

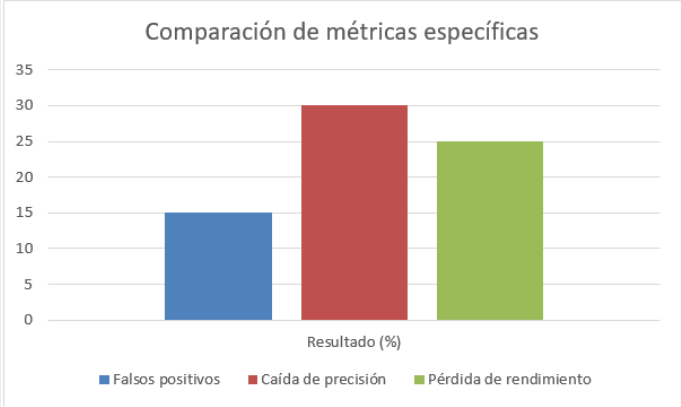
Antes de concluir este apartado se da soporte al porque estadísticamente hablando aún no hay una solución clara a la problemática y el por qué no ha sido solucionado el problema, y es que ahondando más en las investigaciones y tal como lo plantea el estudio de Pinto et al. (2023), donde más del 40% de los sistemas que se encuentran basados en Machine Learning presentan tasas de falsos positivos superiores al 15%, lo que de base compromete su aplicabilidad en entornos críticos y además, Rawat et al. (2022), han conseguido demostrar que los modelos tradicionales de por sí, suelen tener algunas dificultades para identificar ataques que aún no se conocen, mostrando así disminuciones significativas en cuanto a métricas como la precisión de hasta un 30% en situaciones innovadoras. Se puede deducir que estas restricciones se han intensificado en redes de 5G y en el internet de las cosas, donde Noor et al. (2025), informaron de una reducción en el rendimiento del 25% en contextos de borde. Ver tabla 1. Estadísticas de porcentajes de error y pérdida de rendimiento y grafica 3. Comparación de métricas específicas.

Tabla 1. *Estadísticas de porcentajes de error y pérdida de rendimiento.*

Estudio / Fuente	Año	Contexto / Entorno	Métrica evaluada	Resultado reportado	Implicación principal
Pinto et al.	2023	Sistemas basados en *Machine Learning*	Tasa de falsos positivos	> 15% en más del 40% de los sistemas	Alta tasa de falsos positivos compromete la aplicabilidad en entornos críticos.
Rawat et al.	2022	Modelos clásicos de detección	Precisión en detección de ataques nuevos	Caída de hasta el 30% en escenarios no vistos	Los modelos tradicionales no generalizan bien ante ataques inéditos.
Noor et al.	2025	Redes 5G e IoT (entornos de borde)	Rendimiento general del sistema	Pérdida de rendimiento del 25%	Las limitaciones se agravan en entornos distribuidos y de baja latencia.

Fuente: Elaboración Propia Basada en la RSL.

Figura 3. *Comparación de métricas específicas de porcentajes de error y perdida de rendimiento.*



Fuente: Elaboración Propia Basada en la RSL.

Como podemos observar en la tabla 1 y en la figura 3, al comparar la relevancia de dichas métricas específicas, podemos dar conclusión a que estas investigaciones han pretendido revelar y despejar la valiosa necesidad de pasar de técnicas estrictas a enfoques mucho más flexibles, mucho más adaptativos y escalables. Esta investigación se encuentra alinea con esta tendencia, donde se propone un sistema híbrido que permita integrar aspectos tanto cuánticos como clásicos, y el cual aproveche las ventajas del Aprendizaje Automático Cuántico QML para mejorar la detección de intrusos en situaciones complejas, tal como lo evidencian los datos del conjunto CIC-IDS-2017.

Y ya para concluir este apartado se denota que, en las estadísticas revisadas, se evidencian que los enfoques clásicos y basados únicamente en aprendizaje automático ML y aprendizaje profundo DL, no resuelven de forma efectiva los desafíos actuales en detección de intrusiones. Toda vez que, las tasas de falsos positivos, la escasa adaptabilidad y la dependencia de datos desactualizados limitan su aplicabilidad en entornos modernos. Por ello, se justifica la exploración de modelos híbridos cuántico-clásicos, que han demostrado mejoras sustanciales en precisión, eficiencia y robustez, especialmente en entornos NISQ y redes emergentes.

2.2. Aprendizaje automático en detección de intrusos

De la misma manera, el empleo de las técnicas y los métodos de aprendizaje automático ML, han logrado transformar y de una u otra forma revolucionar significativamente los sistemas de detección de intrusos que se encuentran basados en anomalías. Al igual que Almseidin et al. (2018) y Sarker (2021) señalan la importancia de algoritmos clásicos como SVM y Random Forest en IDS basados en anomalías, la literatura posterior al año 2020 confirma esta base tradicional, pero también evidencia una clara tendencia hacia enfoques más automatizados y complejos. Revisiones sistemáticas recientes como, por ejemplo; Maseer et al. (2023) y Rehman et al. (2025) y estudios especializados tales como, Alakhras et al. (2025) muestran cómo la comunidad de investigación ha evolucionado desde simples modelos supervisados hacia modelos híbridos, ensamblados y basados en aprendizaje profundo diseñados para resolver problemas del mundo real con mayor automatización y robustez frente a amenazas dinámicas.

Por tal razón, algoritmos tales como: las máquinas de soporte vectorial SVM, los bosques aleatorios Random Forest y las redes neuronales han demostrado grandes niveles de precisión en la clasificación de tráfico tanto legítimo como dañino. No obstante, estas técnicas y métodos presentan limitaciones y dificultades de frente a la escalabilidad y el manejo de grandes cantidades de datos y a su capacidad de generalización ante ataques inéditos.

Es así, que, el uso de técnicas más complejas como redes neuronales profundas DL ha mejorado la identificación de patrones. Modelos de vanguardia como: "*MAD-GAN*", que utilizan redes generativas adversarias han mostrado una fuerte influencia en la identificación

de intrusiones, particularmente en datos de series temporales y en tráfico de red complejo (Li et al. 2019); Por ejemplo, algunos estudios han combinado *GAN* con redes neuronales profundas para potenciar la eficacia de detección y crear muestras sintéticas que son valiosas para el entrenamiento, logrando así mejoras significativas en la precisión y disminuyendo las tasas de falsos positivos y negativos en datasets estándar como NSL-KDD y CICIDS2017 (Ajdani, 2025).

Igualmente, esfuerzos recientes han abordado el uso de arquitecturas GAN para proteger las comunicaciones en vehículos eléctricos y otros ambientes de IoT, logrando niveles elevados de precisión y rendimiento en tiempo real (Kalyan et al., 2025). Investigaciones más avanzadas también han creado marcos que integran GAN con redes profundas en formatos de ensamble, lo que aumenta la resistencia y la habilidad de generalizar ante ataques complejos en áreas de IoT y de la nube (Nazim et al., 2025).

También por su parte, Musa et al. (2020) y Khan et al. (2019), han demostrado superar las limitaciones de los métodos tradicionales en la detección de ataques complejos. Asimismo, el uso de modelos de ensamble se posiciona como una estrategia robusta para mejorar la precisión y reducir errores en entornos heterogéneos (Premalatha, 2025; Danso et al., 2022). Si consideramos esto, los métodos de ensamble junto con el aprendizaje profundo se establecen como alternativas sólidas ante las restricciones de los métodos tradicionales. Ciertas revisiones de técnicas de ensamble evidencian que la fusión de varios clasificadores, que incluyen estructuras profundas como CNN, LSTM y GRU, logra niveles de detección más altos que los modelos por separado, mostrando métricas de precisión y F1 mucho más elevadas en conjuntos de datos de intrusión actuales (Odeh y Abu Taleb, 2023).

Cabe mencionar a; Khan et al. (2024), que, aunque su trabajo fue retirado en febrero del 2025, podemos destacar que sugirió un modelo simple que combina RNN-RF, en el cual la red neuronal es responsable de extraer características temporales y el Random Forest funciona como un clasificador eficaz. A pesar de que el análisis incluye advertencias editoriales, su diseño híbrido es conceptualmente significativo.

Por otra parte, tenemos a Noor et al. (2025), quienes dirigieron y realizaron un estudio de forma sistemática, sobre los métodos de aprendizaje automático y la transferencia del conocimiento en los sistemas de detección de intrusiones para redes 5G y la computación en

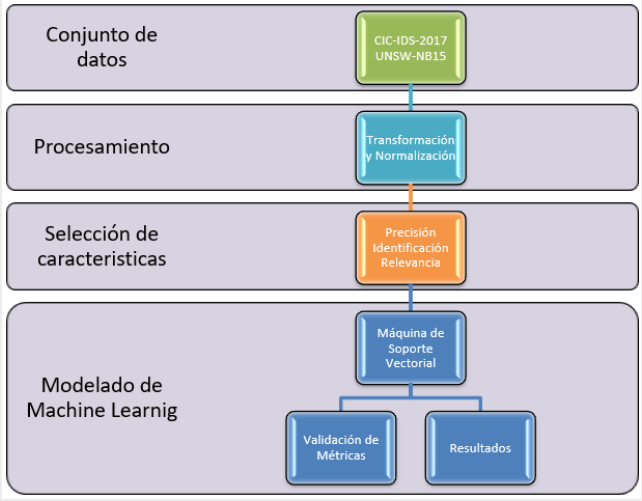
la nube. Ellos concluyen que el aprendizaje transferido y los sistemas híbridos son de base, fundamentales para aumentar la flexibilidad en contextos cambiantes.

De la misma forma, se han estudiado a aquellos modelos híbridos que han logrado combinar el aprendizaje automático tradicional con las técnicas profundas y los algoritmos de optimización, obteniendo algunos resultados sorprendentes. Por ejemplo, Talukder et al. (2023), lograron alcanzaron un 99.99% de precisión en el conjunto de datos KDDCup'99 y un 100% en el conjunto de datos CIC-MalMem-2022 utilizando SMOTE y XGBoost junto a redes profundas.

En contextos de IoT, Akif et al. (2025), quienes lograron mejoras notables con clasificadores de votación que integran RF, XGBoost, KNN y AdaBoost en el conjunto de datos IoT-23. En esta misma línea, Danso et al. (2022), demuestran que el uso de aprendizaje automático híbrido es fundamental para abordar la heterogeneidad de los dispositivos en redes IoT, permitiendo una detección de intrusiones más robusta frente a patrones de tráfico altamente variables. Estos análisis demuestran que el aprendizaje automático tradicional ha llegado a un nivel avanzado, pero necesita combinarse con métodos híbridos.

Es importante de igual manera, mencionar los conjuntos de datos de referencia como KDDCup'99, NSL KDD y el más reciente CIC-IDS-2017 han sido empleados ampliamente como bancos de prueba y plataformas de evaluación para poder comparar y validar distintos modelos de detección. De igual forma, el uso de técnicas más complejas y sofisticadas tales como; redes neuronales profundas CNN, RNN, LSTM, han mejorado la captación e identificación de patrones tanto temporales como espaciales. Estudios investigativos han reportado un alto rendimiento en conjunto de datos como CIC-IDS-2017 y UNSW-NB15 mediante el uso de métodos híbridos como CNN-LSTM y CNN-SVM (Rawat et al., 2022).

Figura 4. *Diagrama de flujo para la metodología de aplicabilidad del aprendizaje automático en la detección de intrusos.*



Fuente: Elaboración Propia Basada en la Interpretación Metodológica de (Noor et al., 2025).

En la **figura 4**. Podemos ver que se ha tomado como primera medida un conjunto de datos el cual es procesado y caracterizado para luego aplicar el modelado vectorial y por último analizar su posterior validación y resultados.

Estos análisis demuestran que el aprendizaje automático tradicional ha llegado a un nivel avanzado, pero necesita combinarse con métodos híbridos, optimización metaheurística y enfoques de adaptación para abordar los retos actuales. Este cambio en la metodología respalda la investigación de modelos cuántico-clásicos como los que se presentan en este documento, los cuales intentan superar las restricciones de escalabilidad, exactitud y capacidad de generalización en la identificación de intrusiones.

2.3. Computación cuántica y aprendizaje automático cuántico

De la computación cuántica podríamos decir que surge y se presenta como un paradigma alternativo, capaz de proporcionar beneficios en problemas complejos relacionados con la optimización y clasificación. Su base se fundamenta en conceptos y principios como la superposición, el entrelazamiento y la interferencia cuántica. En este contexto, se han desarrollado algoritmos de QML cuyo objetivo busca explotar el espacio de Hilbert para representar datos de manera eficiente. Este fundamento teórico fue definido hace algunos

años en investigaciones importantes, como las realizadas por Schuld et al. (2015) y Biamonte et al. (2017), que explican de qué manera estos principios físicos pueden convertirse en posibles beneficios computacionales para los algoritmos de aprendizaje cuántico (Devadas y T, 2025).

A pesar de los avances, la mayoría de los dispositivos todavía se encuentran dentro de la etapa NISQ como se expresa en Solar et al. (2024), caracterizada por un número limitado de qubits y una alta susceptibilidad al ruido, lo que limita la profundidad de los circuitos y la escala de los modelos entrenables sin una corrección de errores robusta. Pero, para superar estas barreras, se han propuesto métodos como las Máquinas de Soporte Vectorial Cuánticas (Camargo et al., 2024). Redes Neuronales Cuánticas (Farhi & Neven, 2018). Y el uso de espacios de características mejorados cuánticamente, los cuales permiten clasificar datos en dimensiones inalcanzables para la computación clásica (Orazi et al., 2024; Havlíček et al., 2019).

Algunos de los modelos destacados o más notables son:

QSVM: *Máquina de Vectores de Soporte Cuántica*, que aprovecha y utiliza kernels y/o núcleos cuánticos para realizar la clasificación.

VQC: *Clasificador Cuántico Variacional*, que integra y combina circuitos parametrizados con técnicas de optimización tradicional.

QCNN: *Red Neuronal Cuántica Convolucional*, aplica el principio de las redes convolucionales en el ámbito cuántico, lo que facilita la captura en niveles de características.

QAOA: es un *Algoritmo Cuántico de Optimización Aproximada*, el cual es aplicado y utilizado en problemáticas de optimización combinatoria.

Si bien las plataformas actuales presentan limitaciones y restricciones como la cantidad de qubits, el ruido y la decoherencia, los progresos en hardware y software han impulsado un aumento en la cantidad de investigaciones sobre la aplicación de QML en la ciberseguridad.

Estas limitaciones y entorno NISQ, nos permiten inferir que, a pesar de los avances en la tecnología del hardware cuántico, la mayoría de los dispositivos presentes todavía se encuentran dentro de la etapa NISQ. Que de acuerdo con Oda et al. (2023), el ruido y la decoherencia son los desafíos técnicos predominantes en esta era, afectando la estabilidad

de los qubits y limitando la profundidad de los circuitos que pueden ejecutarse con fidelidad. Escala Intermedia Ruidosa, que se caracteriza por: Un número limitado de qubits, una alta susceptibilidad al ruido y la decoherencia y la necesidad de simuladores para realizar validaciones experimentales. Por tanto, las limitaciones han llevado al desarrollo de modelos híbridos que combinan lo cuántico y lo clásico, donde el procesamiento previo y la extracción de características se llevan a cabo en sistemas convencionales, y las tareas de clasificación u optimización se asignan a circuitos cuánticos.

Tabla 2. *Comparativa de Modelos de Aprendizaje Automático Cuántico QML.*

Modelo	Descripción	Ventajas	Aplicaciones	Limitaciones en NISQ
QSVM (Quantum Support Vector Machine**)**	Implementa SVM usando kernels cuánticos para proyectar datos en espacios de alta dimensión (espacio de Hilbert) y facilitar la clasificación.	- Aprovecha kernels no triviales - Potencial mejora en separabilidad de clases - Eficiencia en espacios complejos	- Clasificación supervisada - Detección de intrusos - Reconocimiento de patrones	- Requiere circuitos profundos - Sensible al ruido - Escalabilidad limitada por qubits
VQC (Variational Quantum Classifier**)**	Usa circuitos cuánticos parametrizados y técnicas clásicas de optimización para entrenar el modelo.	- Flexibilidad de diseño - Compatible con hardware NISQ - Permite modelos híbridos	- Clasificación de datos estructurados - Seguridad en redes - Análisis de amenazas	- Entrenamiento lento - Dependencia de optimizadores clásicos - Ruido afecta el gradiente
QCNN (Quantum Convolutional Neural Network**)**	Extiende las CNN al ámbito cuántico mediante capas de convolución cuántica, ideal para detectar patrones locales.	- Reducción de parámetros - Detecta características jerárquicas - Potencial para datos de alta dimensión	- Análisis de imágenes cuánticas - Procesamiento de señales - Detección de anomalías	- Difícil implementación física - Circuitos aún experimentales - Necesita más validación práctica
QAOA (Quantum Approximate Optimization Algorithm**)**	Diseñado para problemas de optimización combinatoria mediante una secuencia parametrizada de puertas cuánticas.	- Bueno para problemas NP-duros - Adapta algoritmos clásicos (como Max-Cut) - Funciona en modelos híbridos	- Optimización de rutas - Detección de configuraciones de ataque - Planeación de seguridad	- Complejidad de entrenamiento - Alta sensibilidad a ruido - Requiere qubits bien calibrados

Fuente: Elaboración Propia Basada en la RSL.

Es por ello, que encontramos que algunas de las investigaciones han demostrado que los modelos de Aprendizaje Automático Cuántico QML pueden llegar a superar a sus contrapartes clásicas en aquellas tareas de identificación de anomalías, como, por ejemplo:

En el estudio de Kalinin & Krundyshev (2023), dan cuenta de una mejora del 98% en la precisión al usar QSVM y QCNN en conjuntos de datos de tráfico de IoT, y por su parte,

Abdulrazzaq Altarraji & Mokdad (2024), mostraron que la implementación de ZZ Feature Maps en QSVC disminuye notablemente los falsos positivos en la detección de fraudes. Y Gong et al. (2022), confirmaron la efectividad de un modelo VQNN en dispositivos NISQ, logrando un 97. 21% de precisión en el conjunto de datos CIC-IDS-2017, tal como se puede observar en la tabla comparativa de métricas relevantes para estos casos.

Tabla 3. *Comparativa de métricas relevantes.*

Enfoque	Conjunto de datos	Métricas relevantes
QSVM + QCNN	Conjunto de datos propio (10M)	Accuracy 98%, tiempo de entrenamiento reducido
VQNN	CIC-IDS-2017, NSL-KDD	Accuracy 97.21%
QSVC con ZZ Feature Map	Conjunto de datos financiero	Precisión 1.00, recall 0.87

Fuente: Elaboración Propia Basada en la RSL.

Estos hallazgos refuerzan la claridad de emplear un aprendizaje automático cuántico QML en sistemas de detección de intrusos, especialmente cuando se les logra dar uso junto a los métodos clásicos de preprocesamiento y codificación.

2.4. Modelos híbridos cuántico-clásicos

En este sentir y dado que, los ordenadores cuánticos aún no han logrado alcanzar la madurez necesaria y tampoco están lo suficientemente desarrollados para procesar grandes conjuntos de datos de manera autónoma e independiente, se han desarrollado enfoques híbridos, que combinan los elementos tanto cuánticos como clásicos. Estos enfoques unen la capacidad de procesamiento previo y la reducción de dimensiones de los modelos tradicionales con la etapa de clasificación o aprendizaje que se lleva a cabo en un procesador cuántico. Es de mencionar, que, la literatura en investigaciones recientes demuestra que este tipo de estructuras pueden mejorar tanto la precisión como la eficiencia en los cálculos, al mismo tiempo que facilitan una transición gradual hacia un entorno de computación cuántica más confiable.

Por ejemplo, Metawei et al. (2020) y Cerezo et al. (2021) sostienen que los algoritmos variacionales y los modelos híbridos representan la ruta más viable para obtener beneficios prácticos en el corto plazo. Esta convergencia es particularmente relevante en la ciberanalítica, donde la integración de aprendizaje híbrido cuántico-clásico está abriendo

nuevas fronteras para la detección de amenazas avanzadas en infraestructuras críticas (Fan et al., 2024; Noor et al., 2018).

Asimismo, la literatura reciente ha fomentado el uso de modelos híbridos, particularmente en situaciones NISQ, que presentan desafíos como ruido y un número limitado de qubits. Estos modelos integran el preprocesamiento clásico con la clasificación cuántica, como, por ejemplo: según Nicesio et al. (2023), realizaron un análisis exhaustivo que reveló una mejora en la precisión y una disminución del tiempo en el entrenamiento de modelos híbridos VQC y QSVM, por otra parte, Abreu et al. (2024), destacaron la solución QML-IDS, que emplea VQC, QSVM y QCNN, donde informaron que alcanzaron un F1-score de hasta 95.6% en el conjunto de datos CIC-IDS-2017, destacando que QCNN ofreció una mejor clasificación en comparación con los modelos tradicionales en simuladores cuánticos.

Adicionalmente, un estudio reciente titulado "Un nuevo sistema de detección de intrusiones basado en una máquina de soporte vectorial cuántica híbrida y un optimizador de lobo gris mejorado" desarrolló un modelo QSVM optimizado a través de un lobo gris mejorado IGWO para el Internet de las Cosas, logrando altos niveles de precisión y una baja tasa de falsos positivos (Elsedimy et al., 2024).

Estas investigaciones concuerdan en que la incorporación de técnicas cuánticas no solo potencia métricas convencionales, sino que también facilita una disminución considerable en el tiempo de entrenamiento, un factor crucial en situaciones con recursos restringidos. Sin embargo, la gran parte de los ensayos se ha llevado a cabo en simuladores, lo que enfatiza la importancia de comprobar estos modelos en hardware cuántico auténtico.

2.5. Estudios previos en IDS cuánticos

Investigaciones recientes han indagado en la utilización de modelos QSVM y VQC en conjuntos de datos de referencia como CIC-IDS-2017, logrando resultados alentadores en la identificación de tráfico normal y malicioso. Se ha notado que, aunque las métricas de precisión y recall todavía no superan en gran medida a los modelos tradicionales, la tendencia indica que el avance evolutivo del hardware cuántico permitirá obtener ventajas competitivas en un futuro cercano.

Además, algunas investigaciones han comenzado a sugerir ensambles híbridos cuánticos, integrando y combinando varios clasificadores para reforzar la detección de intrusiones y reducir los problemas causados por el ruido cuántico.

Tabla 4. *Comparativa de estudios previos.*

Estudio	Enfoque	Conjunto de datos	Métricas clave	Ventaja principal	Limitación
Elsedimy et al. (2024)	QSVM + IGWO	CIC-IDS-2017	F1-score, precisión	Optimización de hiperparámetros, reducción de falsos positivos	Simulación
Abreu et al. (2024)	QSVM + VQC + QCNN	CIC-IDS-2017	Accuracy 97.8%, F1 95.6%	QCNN mejora recall en ataques raros	Dependencia de reducción de dimensionalidad
Kalinin & Krundyshev (2023)	QSVM + QCNN	Dataset propio (10M)	Accuracy 98%, tiempo de entrenamiento reducido	Escalabilidad en Big Data	Simulación
Gong et al. (2022)	VQNN	CIC-IDS-2017, NSL-KDD	Accuracy 97.21%	Viabilidad en NISQ, ventaja frente a ANN/SVM	Requiere optimización de circuitos
Abdulrazzaq Altarraji & Mokdad (2024)	QSVC con ZZ Feature Map	Dataset financiero	Precision 1.00, recall 0.87	Reducción de falsos positivos	Dataset no de red, extrapolación indirecta
Kadry et al. (2023)	QNN + WOA-CSO + ECC	NSL-KDD, KDD99	F1-score alto, cifrado integrado	Seguridad integral (detección + protección)	Complejidad computacional elevada
Alluhaibi, R. (2024)	QML-SVM, QML-NN, QML-LR	CIC-IDS-2017	Accuracy 91–92.5%	Mejora en APTs y ataques zero-day	Costo computacional alto
Kim & Madhavi (2024)	QSVM + Quantum Clustering	NSL-KDD, UNSW-NB15	Recall, precisión	IDS integral supervisado/no supervisado	Simulación

Fuente: Elaboración Propia Basada en la RSL.

La **tabla 4**. Comparativa de estudios previos, nos evidencia como el trabajo de Elsedimy et al. (2024), es el que mejor se ajusta al piloto: emplea QSVM optimizado, utiliza el conjunto de datos CIC-IDS-2017, y reporta mejoras claras en precisión y tasa de falsos positivos. La incorporación de IGWO como técnica metaheurística lo hace especialmente beneficioso para el módulo de ajuste.

Por su parte Abreu et al. (2024), presentan un sistema integral QML-IDS que incluye varios clasificadores cuánticos. Resalta la implementación de QCNN, que incrementa el recall en ataques poco frecuentes, un aspecto clave en la detección de intrusos. A su vez, Kalinin y Krundyshev (2023), proporcionan pruebas de escalabilidad en grandes volúmenes de datos (>10^6 registros), logrando disminuir el tiempo de entrenamiento. Aunque utilizan un conjunto

de datos propio, su arquitectura podría servir para el diseño modular del trabajo propuesto. Asimismo, Gong et al. (2022), confirman la efectividad de un esquema VQNN en hardware NISQ, lo que refuerza la factibilidad práctica del enfoque híbrido. Su comparación con varios algoritmos tradicionales lo establece como un referente en la metodología.

También se destaca a Abdulrazzaq Altarraji y Mokdad (2024), que, aunque centrados en el fraude financiero, muestran que el ZZ Feature Map en QSVC puede superar a los modelos tradicionales en términos de precisión. Su flujo híbrido que incluye PCA y normalización es aplicable al tráfico de red. Por otro lado, Kadry et al. (2023), presentan una arquitectura híbrida que fusiona la detección con el cifrado ECC, lo cual es útil si se pretende expandir en algún momento, un proyecto enfocado hacia la seguridad integral. Ya que la implementación de dos técnicas metaheurísticas WOA + CSO le proporciona robustez metodológica.

Otras investigaciones han examinado la comparación directa entre métodos clásicos y cuánticos. Por ejemplo, Alluhaibi (2024), realiza una comparación directa entre QML y ML convencional, evidenciando mejoras en la detección de amenazas avanzadas. Y Kim y Madhavi (2024), añaden un enfoque con clustering cuántico, lo que facilita la utilización de técnicas no supervisadas en IDS. Aunque no trabajan con CIC-IDS-2017, su estructura integral tiene un valor conceptual significativo.

2.5.1. Ensamble y Votación en Modelos Híbridos QSVM, VQC, QCNN

Dentro de los textos académicos también se descubrieron estudios que presentan técnicas de ensamble y comparación entre clasificadores, donde Dong et al. (2022), sugieren el Algoritmo de Enjambre de Escarabajos Cuánticos QBSA para mejorar un Máquina de Aprendizaje Extremo ELM en el contexto de IDS. Aunque no emplean clasificadores cuánticos de manera directa, su método híbrido de optimización proporciona inspiración para mejorar la selección de hiperparámetros en el proceso que se sugiere en este piloto experimental.

De igual manera, Shen et al. (2024), ofrecen una propuesta de computación cuántica basada en la inspiración QIC que incluye un sistema de votación entre modelos clásicos y cuánticos simulados. A pesar de que no se ejecutan en un hardware real ni incorporan QSVM, VQC o QCNN, su diseño de conjunto parcial respalda la metodología del modelo VOTING desarrollado, donde se pone en práctica funcionalmente.

Por otro lado, la investigación de Payares & Martínez-Santos (2021), que comparan QSVM y QNN en la identificación de ataques DDoS, subraya avances en precisión y recuperación de información. Su metodología es tanto experimental como simulada, pero enfatiza la utilidad de modelos cuánticos en situaciones de gran volumen de tráfico.

Es indispensable mencionar la investigación de Kadi et al. (2025), quienes ofrecen un análisis detallado, cotejando diversas estrategias de codificación entre lo cuántico y lo clásico, enfocadas en identificar posibles intrusiones. Sus estudios exploran métodos como Angle Encoding, Amplitude Encoding y Pauli Feature Maps. Evalúan cómo estas técnicas influyen en la capacidad de distinguir diferentes tipos de actividad y en el rendimiento de los cálculos. Esto es clave para seleccionar las codificaciones correctas y elegir representaciones que funcionen mejor en los sistemas cuánticos ruidosos de escala intermedia NISQ.

Tabla 5. *Ventajas y desventajas de los competidores cuánticos híbridos.*

Referencia / Pipeline	Ventajas principales	Desventajas / Limitaciones
Dong et al. (2022)	Optimización metaheurística (QBSA) para ELM, Mejora en selección de hiperparámetros	No usa clasificadores cuánticos, No ejecuta en hardware real, Sin trazabilidad ni ensamble funcional
Shen et al. (2024)	Voting parcial entre modelos simulados, Enfoque cuántico-inspirado (QIC)	No incluye QSVM, VQC ni QCNN, No ejecuta en hardware real, Ensamble no funcional ni trazable
Payares & Martínez (2021)	Comparación entre QSVM y QNN, Aplicación a tráfico DDoS	Simulado, no operativo, Sin integración en ensamble, Sin optimización adaptativa
Kadi et al. (2025)	Comparativa profunda de codificaciones cuánticas, Análisis técnico útil para diseño de circuitos	No implementa modelos de clasificación, No aplica ensamble ni votación, No ejecuta en hardware real
Kadry et al. (2023)	Integración de QNN con ECC para seguridad, Mejora en precisión y cifrado	Simulado, no ejecutado en hardware real, sin ensamble ni lógica adaptativa, No aplica minilotes
Gong et al. (2022)	Uso de VQNN para detección de ataques, Arquitectura cuántica especializada	Simulado, no operativo, Sin ensamble ni trazabilidad, No incluye optimización incremental
Elsedimy et al. (2024)	QSVM + Grey Wolf Optimizer, Mejora en recall y precisión	Simulado, no ejecutado en hardware real, Sin ensamble funcional, Sin trazabilidad visual

Fuente: Elaboración Propia Basada en la RSL.

Tabla 6. *Comparativa de los conjuntos de datos.*

Conjunto de datos	Año	Nº registros (aprox)	Tipos de ataque	Uso típico / limitación
KDDCup'99	1999	~5Millones	DoS, Probe, U2R, R2L	Desbalance y redundancia de artefactos
NSL-KDD	2009	~150Mil	DoS, Probe, U2R, R2L	Versión corregida de KDD – Ataques modernos
CIC-IDS-2017	2017	Millones	DDoS, BruteForce, Bot, Infiltration.	Dataset moderno, Alto volumen
UNSW-NB15	2015	~2.5Millones	Fuzzers, Shellcode, Worms, Backdoor	Menor uso en estudios cuánticos
IoT-23 / Bot-IoT	2019-2020	Variable	IoT attacks	Ideal para escenarios IoT – Requiere adaptación

Fuente: Elaboración Propia Basada en la RSL.

La **tabla 6**. Comparativa de los conjuntos de datos, presenta un panorama histórico y técnico de cinco conjuntos de datos fundamentales en la detección de intrusos IDS, cada uno resaltando las particularidades que definen su contexto, tipo de ataque y su importancia en investigaciones actuales.

KDDCup'99 – El precursor ruidoso

Introducido en 1999, este conjunto de datos fue uno de los primeros intentos significativos por simular el tráfico de red junto con ataques. Con más de cinco millones de registros, incluye categorías tradicionales como: Denegación de Servicio DoS, Escaneo Probe, De Usuario a Raíz U2R, y De Remoto a Local R2L.

Sin embargo, su utilidad está afectada por un importante desbalance entre las clases y la repetición de elementos, lo que provoca un sobreajuste en los modelos y limita su capacidad de generalización.

NSL-KDD – Optimización y ajuste

En 2009, se introduce NSL-KDD como una versión mejorada del anterior. La cantidad de registros se reduce a alrededor de 150 mil y se eliminan duplicados, optimizando así la distribución de las clases. Aunque mantiene los mismos tipos de ataques, su objetivo es ofrecer una base más realista y equilibrada para la formación de modelos. Sin embargo, no incluye ataques recientes, lo que restringe su relevancia.

UNSW-NB15 – Diversidad técnica

Desarrollado en 2015 por la Universidad de Nueva Gales del Sur, este conjunto de datos contiene alrededor de 2. 5 millones de registros y abarca ataques menos comunes como: fuzzers, código shell *Shellcode*, gusanos *Worms* y puertas traseras *Backdoors*. Su rica complejidad técnica lo convierte en un recurso valioso para investigaciones en detección avanzada, aunque su aplicación en entornos cuánticos o simulaciones de QML ha sido limitada, probablemente debido a su estructura o falta de flexibilidad.

IoT-23 / Bot-IoT – Enfoque en dispositivos inteligentes

Estos conjuntos de datos más recientes 2019-2020 se centran en ataques dirigidos a dispositivos IoT, un área crucial y en crecimiento. Su tamaño es variable y, aunque son ideales para recrear situaciones en redes inteligentes, necesitan ajustes para adaptarse a sistemas más tradicionales o investigaciones cuánticas, dada su estructura y clasificación particular.

CIC-IDS-2017 – Amplitud y modificaciones

Este conjunto de datos originario de Canadá representa un avance notable: abarca millones de registros con tráfico realista y ataques actuales como: DDoS, Fuerza Bruta *BruteForce*, redes de *botnets* e infiltraciones. Su amplio volumen y variedad lo hacen muy popular, aunque su tamaño puede requerir una infraestructura sólida para un correcto procesamiento y entrenamiento.

En resumen, cada conjunto de datos representa una etapa en la evolución de la ciberseguridad:

- KDD y NSL-KDD son aptos para evaluaciones elementales.
- **CIC-IDS-2017** define el estándar actual por su tamaño y realismo.
- UNSW-NB15 ofrece variedad técnica.
- IoT-23/Bot-IoT apoyan estudios en entornos distribuidos e inteligentes.

Finalmente se trabajará con el conjunto de datos **CIC-IDS-2017** por su tamaño y diversidad dentro del realismo de tráfico en la red. Asimismo, este equilibrio lo establece como el estándar contemporáneo para analizar sistemas de detección de intrusos.

2.6. Síntesis crítica y posicionamiento de la propuesta

El estudio que se presenta expone un foco de formas detalladas acerca de la información, la cual se encuentra especializada sobre la detección de intrusos, el aprendizaje automático y la computación cuántica, donde se muestra un cambio en los vicios metodológicos; primero, pasando de métodos los cuales se encuentran basados en firmas a aquellos sistemas que presentan adaptación al cambio como los híbridos o cuánticos-clásicos; aunque los modelos tradicionales han demostrado ser efectivos en los entornos controlados, su rendimiento se ve afectado ante el tráfico encriptado, los ataques Zero-day y a todas aquellas situaciones cambiantes como las observadas en las redes 5G o el Internet de las Cosas IoT.

Asimismo, encontramos que la computación cuántica, da cuenta especialmente, dentro del Aprendizaje Automático Cuántico QML, el cual, está surgiendo como una opción alentadora en modelos como: la Máquina de Vectores de Soporte Cuántico QSVM, el Circuito Cuántico Variacional VQC y la Red Neuronal Convolucional Cuántica QCNN, donde estos evidencian avances en términos de precisión, eficacia y capacidad de generalización. Sin embargo, la mayor parte de las investigaciones que han sido revisadas dentro de este estudio, se concentran en simulaciones, más no se implementan pruebas en hardware real y no aplican estrategias de conjunto funcional ni trazabilidad operativa.

Por tanto, dentro de este marco explicativo, el piloto experimental propone un flujo que combina lo cuántico y lo clásico, de tal forma que integra los modelos de aprendizaje como: la Máquina de Vectores de Soporte Cuántico QSVM, el Circuito Cuántico Variacional VQC y la Red Neuronal Convolucional Cuántica QCNN, dentro de un esquema de votación funcional, y realizando inicialmente la ejecución en Qiskit Aer aer_simulator_statevector, un simulador de circuitos cuánticos de alto nivel que ofrece IBM y que permite simular el estado cuántico del sistema con precisión. Como segunda fase, se plantea complementar el entrenamiento con pruebas en una computadora cuántica real; en un hardware cuántico físico IBM, que es proporcionado por IBM a través de su plataforma en la nube, con el fin de lograr obtener una comprensión mucho más profunda, donde se pueda incorporar codificación avanzada, lograr una optimización adaptable, donde se pueda observar una trazabilidad visual adecuada y clara y una lógica de activación condicional. Para asegurar la viabilidad de estas pruebas, se empleará el modelo de ejecución de Qiskit Runtime, el cual permite optimizar el rendimiento

y minimizar la latencia en el intercambio de datos entre procesadores clásicos y cuánticos, facilitando una ejecución eficiente en entornos de nube (IBM, s. f.).

De base, esta arquitectura no solo permite abordar las deficiencias metodológicas que han sido encontradas, sino que también establece lo que al parecer puede llegar a ser un nuevo estándar de implementación mucho más reproducible, verificable y porque no, científicamente sólido para la detección de intrusos en entornos NISQ.

Tabla 7. *Vacíos metodológicos en la literatura.*

Dimensión metodológica	¿Presente en estudios previos?	¿Presente en el Piloto Experimental?	Observaciones
Aer_simulator_statevector como fallback	Solo statevector	Sí (IBM Qiskit Aer) Fallback	Simulador de circuitos cuánticos de alto rendimiento que ofrece IBM
Ejecución en hardware cuántico real	No	Sí (IBM Q RuntimeEstimator)	La mayoría usa simuladores
Ensamble funcional entre clasificadores cuánticos	No	Sí (QSVM + VQC + QCNN-like)	Solo proponen ensambles parciales o simulados sin integración operativa
Codificación cuántica avanzada (ZZ, Pauli)	Parcial	Sí	Algunos estudios comparan codificaciones, pero no las aplican en ejecución real
Optimización adaptativa	No	Sí	El piloto experimental usa DataLoader con lógica de parada temprana y ajuste dinámico
Fallbacks clásicos integrados	No	Sí (MLP, RandomForest, XGBoost)	La arquitectura del piloto experimental es resiliente
Evaluación robusta (ROC, matriz de confusión, métricas ponderadas)	Parcial	Sí	Incluye classification_report, confusion_matrix, ROC y métricas por clase
Simulación de ruido y curva de robustez	No	Sí	El modelo evalúa rendimiento ante ruido gaussiano en los datos de entrada
Monitoreo de recursos (CPU, RAM)	No	Sí	Uso de psutil para capturar uso de CPU y memoria durante la ejecución
Aplicación en dataset multiclase moderno (CIC-IDS-2017)	Sí	Sí	Se entrena y evalúa en uno de los datasets más exigentes y actuales
Trazabilidad visual y logging técnico estructurado	No	Sí	Logging con logging, decoradores @trace, visualización con matplotlib y seaborn

Fuente: Elaboración Propia Basada en la RSL.

2.6.1. Divergencia cuántica y matemática

Con el fin de ahondar cada vez más en el posicionamiento critico de la propuesta, se da contextualización al siguiente interrogante ¿Por qué hacerlo cuántico y no solo matemático?, la razón válida la fijación de un enfoque cuántico, que, en sí, no responde únicamente a una preferencia tecnológica, sino más bien a una necesidad metodológica que se encuentra derivada de todas aquellas limitaciones persistentes halladas en los modelos clásicos y matemáticos tradicionales.

A continuación, se detallan las razones clave:

En primera oportunidad observamos a aquellas limitaciones de los modelos clásicos-tradicionales y matemáticos en cuanto a la escasa generalización ante ataques nuevos. Según Pinto et al. (2023), más del 40% de los IDS basados en Machine Learning presentan tasas de falsos positivos superiores al 15%, con una alta especialidad en ataques nunca vistos. En segundo punto observamos la dependencia de la ingeniería de las características ya que los modelos clásicos requieren una selección manual de atributos, lo que, en sí, limita su adaptabilidad. Por otra parte, encontramos que, en el rendimiento decreciente en entornos dinámicos, donde Noor et al. (2025), allegan reportes de una pérdida en el rendimiento del 25% en las redes 5G y los entornos de borde. Y por último la sobrecarga computacional, donde Talukder et al. (2023), evidencian que los modelos matemáticos complejos como CNN-SVM requieren tiempos de entrenamiento mucho más elevados y recursos intensivos.

Ahora bien, las ventajas del enfoque cuántico e híbrido cuántico-clásico enmarcan la división superior en aquellos espacios de alta dimensión como: la Máquina de Vectores de Soporte Cuántico y el Circuito Cuántico Variacional, los cuales proyectan los datos en el espacio de Hilbert, lo que mejora la clasificación de patrones complejos (Kalinin & Krundyshev, 2023). Asimismo, la codificación eficiente de datos: Feature Maps como ZZ y Pauli permiten representar múltiples atributos en estados cuánticos entrelazados, reduciendo la dimensionalidad efectiva (Kadi et al., 2025). Por otra parte, la robustez ante el ruido y variabilidad como: Gong et al. (2022), demuestran; que la Red Neuronal Convolucional Cuántica mantiene precisión superior al 97% en entornos NISQ, incluso con ruido cuántico. También encontramos reducción de tiempo de entrenamiento como en Abreu et al. (2024), donde reportan mejoras en eficiencia computacional al combinar VQC, QSVM y QCNN en

simuladores cuánticos. Y por último tenemos la adaptabilidad estructural, donde los modelos híbridos permiten delegar el preprocesamiento a sistemas clásicos y la clasificación a circuitos cuánticos, optimizando recursos (Elsedimy et al., 2024).

La justificación científica y estratégica enmarca el uso de modelos cuánticos que no reemplazan la matemática, sino más bien, la extiende a un dominio mucho más expresivo y eficiente. La computación cuántica también permite representar, transformar y clasificar datos en formas que los modelos clásicos no pueden replicar, y eso, sin incurrir en costos exponenciales, en contextos como la detección de intrusiones, donde los patrones son sutiles, multiclase y dinámicos, y se conoce que esta capacidad es altamente crítica.

Por tanto, este estudio no busca abandonar los fundamentos matemáticos, sino más bien trascenderlos mediante el uso de una arquitectura híbrida cuántico-clásica que aprovecha la potencia visionaria de la computación cuántica con el fin de dar alcance y/o superar las limitaciones de la escalabilidad, la generalización y la eficiencia que presentan los modelos tradicionales. Es así, que, esta decisión metodológica se sustenta en evidencia empírica, revisiones sistemáticas y resultados experimentales los cuales dan validez al enfoque propuesto como una contribución original y necesaria en el campo de la ciberseguridad.

2.7. Síntesis del estado del arte

El análisis del estado del arte indica que:

1. Los sistemas tradicionales de detección de intrusos basados en aprendizaje automático son útiles, pero enfrentan limitaciones en situaciones complicadas y cambiantes rápidamente. La computación cuántica ofrece un enfoque novedoso para manejar datos, aunque todavía presenta retos técnicos.
2. Las soluciones híbridas representan una alternativa prometedora y han mostrado resultados favorables en la práctica.
3. La aplicación específica de modelos híbridos que combinan métodos cuánticos y clásicos en el conjunto de datos CIC-IDS-2017 se constituye como un área de investigación con un creciente interés y gran potencial.

4. Como resultado del estudio de la literatura revisada, se observa un panorama alentador, aunque todavía en progreso: las metodologías tradicionales que se basan en aprendizaje automático brindan resultados consistentes para casos particulares y conjuntos de datos predefinidos, no obstante, tienen limitaciones en su escalabilidad, adaptabilidad a nuevas amenazas y efectividad en contextos de recursos restringidos.

Por otra parte, la investigación en aprendizaje automático cuántico ha mostrado avances positivos en simulaciones, como la mejora de precisión y la reducción de tiempos de entrenamiento para ciertos procesos, sin embargo, gran parte de la información está limitada a entornos simulados o aquellos que emplean GPU; las pruebas con hardware NISQ y la posibilidad de aplicar estas técnicas a grandes volúmenes de datos siguen siendo obstáculos por superar. Por lo tanto, las técnicas híbridas que integran elementos cuánticos y clásicos se establecen como la alternativa más factible a corto y medio plazo: permiten explotar beneficios cuánticos específicos, como kernels cuánticos o la reducción de dimensiones en el espacio de Hilbert, mientras que utilizan infraestructuras clásicas ya establecidas para el preprocesamiento y almacenamiento.

El Trabajo Fin de Máster se orienta en esta dirección: su objetivo será diseñar y examinar un proceso reproducible que utilice el conjunto de datos CIC-IDS-2017 para comparar diferentes métodos de codificación, clasificadores cuánticos QSVM-VQC-QCNN y modelos tradicionales, con protocolos definidos para evaluar la robustez en las predicciones, así como el costo computacional y la posibilidad de repetir los experimentos.

Capítulo 3: El Algoritmo Xander

3. Introducción

El estudio actual presenta el Piloto Experimental del Algoritmo Xander para la Detección de Intrusiones en red utilizando la Computación Cuántica Híbrida con una metodología enfocada en la identificación de intrusiones en red, utilizando un enfoque híbrido que combina la computación cuántica y clásica. En sí mismo, este sistema fusiona modelos de Aprendizaje Automático Cuántico ***QML*** y Aprendizaje Automático Clásico ***CML*** en un solo flujo que permite analizar, entrenar y evaluar diferentes paradigmas de aprendizaje de manera simultánea.

Por tanto, la principal aportación se encuentra en la integración de tres modelos cuánticos; Máquina de Vectores de Soporte Cuántica ***QSVM***, Clasificador Cuántico Variacional ***VQC*** y una arquitectura similar a Red Neuronal Convolucional Cuántica ***QCNN***, junto con tres modelos clásicos *Random Forest* - Bosque Aleatorio ***RF***, *XGBoost* ***XGB*** y *Multi-Layer Perceptron* - Perceptrón Multicapa ***MLP***.

Este planteamiento tiene como objetivo examinar la viabilidad y el rendimiento comparativo de los modelos cuánticos en relación con sus contrapartes clásicas, dentro de un mismo entorno experimental y utilizando métricas uniformes. Por otro lado, el sistema se elabora en un entorno que se puede controlar y reproducir, empleando IBM Quantum Runtime como el backend cuántico y Qiskit Machine Learning. El flujo de trabajo también incluye un mecanismo de recuperación inteligente, el cual redirige el proceso hacia versiones clásicas de los modelos si los recursos cuánticos no están disponibles, asegurando así la continuidad del funcionamiento y la completitud del piloto experimental.

Además, la arquitectura modular del sistema facilita la medición y comparación sistemática del rendimiento, el ruido, la carga computacional y la eficiencia del sistema híbrido. En conjunto, esta contribución sugiere una arquitectura para detectar intrusiones basada en computación cuántica híbrida, con el propósito de demostrar la viabilidad técnica y el potencial de optimización de este método en el ámbito de la ciberseguridad aplicada.

3.1. Descripción detallada del Algoritmo Xander

El proyecto se clasifica como una contribución de tipo "Piloto Experimental", con un enfoque cuantitativo y aplicado. El objetivo de la investigación es comprobar de manera empírica cómo se combinan los modelos de aprendizaje cuántico en sistemas que detectan intrusiones, evaluando su eficacia a través de métricas de rendimiento como: Accuracy, Precision, Recall, F1-score, considerando diferentes configuraciones de entrenamiento y ambientes de ejecución simulados y reales.

El piloto experimental facilita la observación, comparación y cuantificación de los efectos del uso de circuitos cuánticos variacionales en contraste con algoritmos clásicos convencionales, todo dentro de un proceso unificado y automático. Así, se busca establecer la viabilidad técnica, la eficiencia en el cálculo y la precisión en las predicciones del enfoque cuántico, además de su futuro potencial para integrarse en plataformas de ciberseguridad a gran escala.

Para dar comienzo al desarrollo descriptivo del proyecto y como primera medida, el conjunto tecnológico para el desarrollo del piloto experimental une elementos cuánticos, tradicionales y de ingeniería de datos, organizados en un proceso automatizado. Cada tecnología se evalúo en función de su contribución funcional, su compatibilidad y su efecto en la capacidad de reproducir el experimento. Esta estructura demuestra un método modular, lo que posibilita la expansión futura del sistema hacia arquitecturas distribuidas o su implementación en hardware cuántico auténtico.

Tabla 8. *Conjunto tecnológico de recursos utilizados en el piloto experimental del Algoritmo Xander.*

Tecnología / Herramienta	Tipo / Categoría	Descripción técnica	Justificación de uso en el proyecto
Python 3.9	Lenguaje de programación	Lenguaje interpretado de alto nivel, multiparadigma, ampliamente usado en ciencia de datos, IA y computación cuántica.	Base del desarrollo del pipeline híbrido; permite integración fluida con bibliotecas cuánticas (Qiskit, PennyLane) y de ML clásico (Scikit-learn, PyTorch).
Qiskit (v0.39.2)	Framework de computación cuántica	Biblioteca de código abierto de IBM para construir, simular y ejecutar circuitos cuánticos en hardware real o simulado.	Utilizada para la implementación de los modelos **QSVM**, **VQC** y **QCNN-like**, permitiendo ejecutar experimentos tanto en simuladores locales como en el entorno **IBM Quantum Runtime**.

Qiskit Machine Learning (v0.5.0)	Extensión de Qiskit para ML	Módulo especializado que integra algoritmos de aprendizaje automático cuántico (QML) con flujos de trabajo de IA tradicionales.	Facilita el uso de **EstimatorQNN**, **QuantumKernel**, **TorchConnector** y estructuras de modelos híbridos cuántico-clásicos dentro del pipeline.
IBM Quantum Runtime	Entorno de ejecución cuántica	Plataforma en la nube de IBM que permite la ejecución optimizada de algoritmos cuánticos sobre hardware real con baja latencia y administración de recursos.	Permite validar el rendimiento real del modelo cuántico en hardware físico y realizar comparaciones entre resultados simulados y experimentales.
PennyLane (v0.28.0)	Framework híbrido cuántico–clásico	Librería de código abierto orientada a la diferenciación automática en sistemas cuánticos.	Referencia teórica y de validación cruzada del modelo VQC; complementa el diseño conceptual del enfoque híbrido.
PyTorch (v1.13)	Framework de aprendizaje profundo	Biblioteca de redes neuronales y tensor computing con soporte para GPU y compatibilidad con TorchConnector.	Utilizado para construir las capas híbridas de los modelos cuántico-clásicos (VQC y QCNN-like), optimizando pesos mediante descenso de gradiente y AdamW.
Scikit-learn (v1.1.3)	Framework de aprendizaje automático clásico	Conjunto de herramientas para clasificación, regresión, clustering y preprocesamiento de datos.	Base para el entrenamiento de modelos clásicos (RF, SVC, MLP) y para métricas de evaluación (accuracy, recall, F1, ROC).
XGBoost (v2.1.4)	Biblioteca de boosting	Implementación optimizada del algoritmo Gradient Boosted Trees, orientado al rendimiento y escalabilidad.	Se utiliza como modelo de referencia clásico para comparar la eficiencia del enfoque híbrido frente a técnicas de boosting tradicionales.
Imbalanced-learn (v0.12.0)	Biblioteca de balanceo de datos	Extensión de Scikit-learn para tratar desequilibrios de clase mediante técnicas como SMOTE y Tomek Links.	Aplicada en el preprocesamiento del dataset CIC-IDS-2017 para garantizar una representación equitativa entre tráfico normal y ataques.
NumPy (v1.23.5)	Biblioteca científica	Proporciona soporte para operaciones matemáticas y manejo eficiente de arreglos multidimensionales.	Base del cálculo vectorial en todas las etapas del pipeline, incluyendo la fusión ponderada de probabilidades en el ensamblado híbrido.
Pandas (v1.5.3)	Biblioteca de manipulación de datos	Herramienta para el manejo y análisis de estructuras tabulares.	Empleada para la lectura, consolidación y limpieza de los archivos CSV del dataset CIC-IDS-2017.
Matplotlib (v3.5.3)	Biblioteca de visualización	Permite la generación de gráficos 2D, histogramas, matrices de confusión y curvas ROC.	Utilizada para la representación visual de métricas comparativas, curvas ROC, y análisis de ruido.
Seaborn (v0.11.2)	Extensión de visualización estadística	Proporciona gráficos de alto nivel basados en Matplotlib con estilo mejorado.	Facilita la presentación visual profesional de resultados experimentales en el TFM.
psutil	Biblioteca de monitoreo de recursos del sistema	Permite medir el uso de CPU, memoria y procesos en ejecución en tiempo real.	Se emplea para la evaluación de la **carga computacional**, comparando el impacto entre modelos cuánticos y clásicos.

Dataset CIC-IDS-2017	Fuente de datos (dataset)	Conjunto de datos de tráfico de red que incluye ataques y tráfico legítimo, desarrollado por el Canadian Institute for Cybersecurity.	Dataset base del piloto experimental; reconocido por su realismo y diversidad de ataques. Permite evaluar la eficacia del enfoque híbrido en detección de intrusiones.

Fuente: Elaboración Propia.

Tabla 9. *Ficha técnica del piloto experimental del Algoritmo Xander.*

Campo	Descripción técnica
Nombre del sistema	**Piloto Experimental Xander (Sistema de Detección de Intrusos Híbrido Cuántico-Clásico)**
Versión	7.0 (Experimental Release)
Tipo de contribución	Piloto Experimental Cuantitativo Sistema híbrido cuántico–clásico para detección de intrusiones en red.
Objetivo principal	Implementar y validar un **pipeline híbrido cuántico–clásico** que combine modelos de *Quantum Machine Learning (QML)* y *Classical Machine Learning (CML)* para la detección de tráfico malicioso en red.
Descripción general	El pipeline Xander, integra modelos cuánticos (**QSVM, VQC, QCNN-like**) con modelos clásicos (**RF, XGB, MLP**) bajo una arquitectura de ensamblado ponderado (Weighted Soft Voting). El sistema incorpora control de ruido, medición de carga computacional, análisis comparativo de rendimiento y fallback automático ante fallos cuánticos.
Arquitectura metodológica	1) Carga y preprocesamiento del dataset (CIC-IDS-2017). 2) Reducción de dimensionalidad con PCA (6 componentes). 3) Entrenamiento cuántico (QSVM, VQC, QCNN-like). 4) Entrenamiento clásico (RF, XGB, MLP). 5) Ensamble híbrido cuántico–clásico. 6) Evaluación, métricas y análisis de ruido. 7) Exportación de resultados y generación de informes.
Lenguaje principal	Python 3.9
Frameworks cuánticos	Qiskit (v0.39.2), Qiskit Machine Learning (v0.5.0), IBM Quantum Runtime y PennyLane (referencial).
Frameworks clásicos	Scikit-learn (v1.1.3), PyTorch (v1.13), XGBoost (v2.1.4), Imbalanced-learn (v0.12.0).
Bibliotecas complementarias	NumPy (v1.23.5), Pandas (v1.5.3), Matplotlib (v3.5.3), Seaborn (v0.11.2), psutil
Dataset empleado	**CIC-IDS-2017** (Canadian Institute for Cybersecurity) Tráfico normal y malicioso (ataques SSH, FTP, DoS, DDoS, etc.).
Tamaño de muestra experimental	6.000 registros balanceados mediante SMOTE + Tomek Links (muestras representativas de tráfico normal y anómalo).
Preprocesamiento aplicado	Limpieza de datos, codificación de etiquetas, normalización (StandardScaler), reducción PCA (6 componentes) y balanceo de clases.
Modelos cuánticos	**QSVM:** Quantum Kernel con ZZFeatureMap (3 reps). **VQC:** EfficientSU2 Ansatz (6 qubits, 3 reps, Adam lr=0.001). **QCNN-like:** Convolución cuántica con entrelazamiento local y pooling. Entrenamiento con EstimatorQNN y TorchConnector.

Modelos clásicos (fallbacks)	Random Forest (200 estimadores) XGBoost (300 árboles, depth=6) MLP (2 capas ocultas: 64–32).
Estrategia de ensamblado	Weighted Soft Voting con **ponderación ajustada por Grid Search** y control de semillas (NumPy, Torch, Scikit-learn) para reproducibilidad.
Métricas de evaluación	Accuracy, Precision, Recall, F1-score, AUC-ROC, matriz de confusión, curva ROC, curva de ruido, carga computacional (CPU/Mem).
Visualizaciones generadas	Matrices de confusión (por modelo y global). Curvas ROC comparativas. Gráficas de ruido y CPU/memoria. Radar chart de métricas globales. Tiempos de entrenamiento por modelo.
Salida de resultados	Métricas en formato CSV y JSON; gráficos PNG; artefactos del modelo (. pkl y. pth).
Mecanismo de fallback	Activación automática de versiones clásicas (SVC, RF, MLP) en caso de fallo del backend cuántico o ausencia de Qiskit.
Control de reproducibilidad	Semillas globales (NumPy, Torch, Scikit-learn) fijadas en GLOBAL_SEED = 42.
Plataforma de ejecución	Windows 10 / Anaconda Environment con Python 3.9, soporte para IBM Quantum Runtime y simuladores locales de Qiskit.
Salida final esperada	Conjunto de métricas comparativas y modelo híbrido optimizado con precisión > 95 % en validación experimental.
Autor	**Oscar Javier Peñaloza Araque** Estudiante de Máster en Ciberseguridad Universidad Internacional de La Rioja (UNIR)
Año de desarrollo	2025
Estado del proyecto	En fase de evaluación experimental (versión piloto funcional).

Fuente: Elaboración Propia.

El Piloto Experimental Xander fue concebido en un ambiente regulado, empleando la ruta local del conjunto de datos CIC-IDS-2017 como la fuente básica de información. Asimismo, el procedimiento se realizó de manera sucesiva y reproducible, cumpliendo con las etapas establecidas previamente en la metodología:

Capítulo 4: Metodología del Algoritmo Xander

4. Introducción

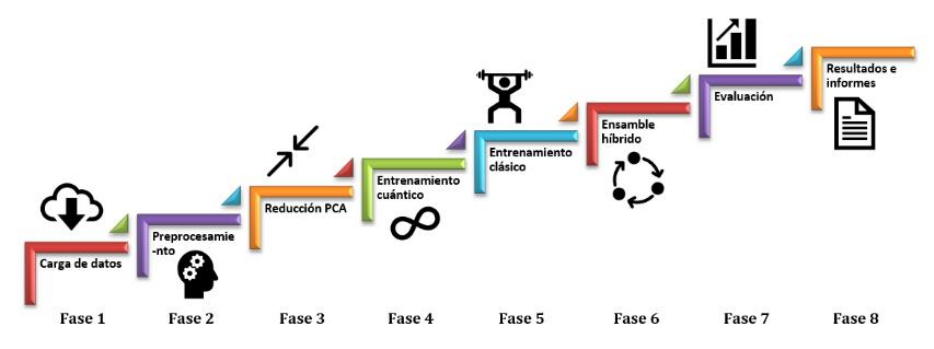
En este apartado se expone la metodología que respalda el desarrollo del Piloto Experimental del Algoritmo Xander para la Detección de Intrusiones en red utilizando la Computación Cuántica Híbrida. Cuyo objetivo es desarrollar un sistema híbrido cuántico-clásico orientado a la identificación de intrusiones en la red. El enfoque metodológico que he elegido es un piloto experimental con un alcance de carácter práctico, ya que pretendo confirmar una propuesta tecnológica a través de un entorno controlado para su implementación.

Asimismo, y tomando como base el análisis exhaustivo de la investigación que lleva por nombre Hacia la generación de un nuevo conjunto de datos de detección de intrusiones y la caracterización del tráfico de intrusiones, articulo que data del año 2018, señalo el conjunto de datos con el cual realizare las pruebas para el piloto experimental. Toda vez, que hacen énfasis en la base de datos que he escogido la cual es: CIC-IDS-2017, y donde se hace mención, de que se encuentra ampliamente valorada en la literatura por su variedad de ataques y su tráfico de red real según (Sharafaldin et al., 2018).

Por otra parte, la investigación integra elementos de Aprendizaje Automático Cuántico QML y Aprendizaje Automático Clásico CML bajo una estructura modular que facilita la fusión de modelos cuánticos y modelos clásicos mediante un ensamblado ponderado. El proceso metodológico sigue un esquema y un flujo secuencial que abarca desde la recolección y el preprocesamiento de los datos hasta la validación del rendimiento del ensamble híbrido.

4.1. Metodología de ejecución del piloto experimental del Algoritmo Xander

Figura 5. *Fases de ejecución del piloto experimental del Algoritmo Xander.*



Fuente: Elaboración Propia.

La estrategia para el desarrollo del proyecto se basa en un método experimental y secuencial, en el que cada etapa del flujo de trabajo ayuda a construir un sistema híbrido que combina tecnologías cuánticas y clásicas para la identificación de intrusiones. El proceso incluye actividades de recopilación y preparación de datos, reducción de la cantidad de dimensiones, entrenamiento tanto cuántico como clásico, ensamblaje con ponderaciones, y un análisis del rendimiento utilizando métricas específicas de eficiencia y efectividad computacional. Esta estructura modular promueve la posibilidad de repetir los experimentos, seguir el rastro de los resultados y hacer comparaciones imparciales entre los modelos cuánticos y sus contrapartes clásicas.

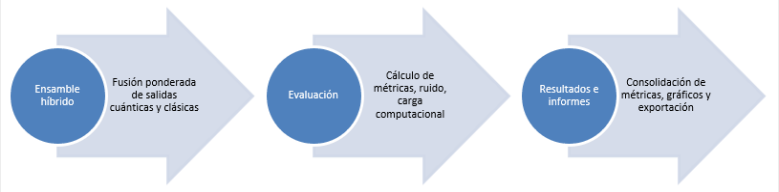
Figura 6. *Flujograma de ejecución del piloto experimental del Algoritmo Xander 1.*

Fuente: Elaboración Propia.

Figura 7. *Flujograma de ejecución del piloto experimental del Algoritmo Xander 2.*

Fuente: Elaboración Propia.

Figura 8. *Flujograma de ejecución del piloto experimental del Algoritmo Xander 3.*



Fuente: Elaboración Propia.

Tabla 10. *Descripción de las fases de ejecución del piloto experimental del Algoritmo Xander.*

Item	Fases	Descripción
1	**Carga de datos**	Integración de los archivos CSV del dataset **CIC-IDS-2017**, unificando el tráfico normal y malicioso para conformar un conjunto de datos consolidado.
2	**Preprocesamiento**	Limpieza de valores atípicos, balanceo de clases (undersampling/oversampling), codificación de etiquetas y escalado de características con **StandardScaler**.
3	**Reducción PCA**	Aplicación de **Análisis de Componentes Principales (PCA)** para reducir la dimensionalidad del conjunto de datos a seis componentes, preservando la variabilidad esencial.
4	**Entrenamiento cuántico**	Implementación de los modelos **QSVM**, **VQC** y **QCNN** mediante **Qiskit** e integración con **IBM Quantum Runtime**. Incluye mecanismos de fallback hacia equivalentes clásicos en caso de error.
5	**Entrenamiento clásico**	Entrenamiento de modelos **Random Forest (RF)**, **XGBoost (XGB)** y **MLPClassifier**, usando los datos reducidos por PCA como referencia base.
6	**Ensamble híbrido**	Fusión ponderada de las salidas cuánticas y clásicas mediante **Weighted Soft Voting**, ajustando los pesos con **Grid Search** para optimizar la precisión.
7	**Evaluación**	Cálculo de métricas de desempeño (**Accuracy, Precision, Recall, F1, AUC-ROC**) y análisis de ruido y carga computacional con **psutil**.
8	**Resultados e informes**	Consolidación de métricas, gráficos comparativos y exportación de resultados en formatos **CSV** y **JSON**, junto con reportes visuales del rendimiento.

Fuente: Elaboración Propia.

4.1.1. Fase 1 - Carga de datos

En esta fase se realizó la integración de los archivos CSV del dataset *CIC-IDS-2017*, unificando el tráfico normal y malicioso para conformar un conjunto de datos consolidado. En esta primera etapa lo que se busco es *reunir y preparar los archivos originales del dataset CIC-IDS-2017* para que se pudieran usar de forma uniforme en el pipeline.

4.1.1.1. Localización de archivos: se identifican las carpetas que contienen los CSV de tráfico normal y de cada tipo de ataque.

4.1.1.2. Lectura robusta: se cargan los .CSV en la memoria; como por dar un ejemplo: con *pandas.read_csv*, cuidando separadores y codificación de texto.

4.1.1.3. Estandarización de columnas: se renombran los encabezados para que todos los archivos tengan el mismo esquema como: IPs, puertos, protocolo, métricas de flujo, etiqueta.

4.1.1.4. Normalización de tipos: se convierten las columnas a sus tipos correctos enteros, flotantes, fechas y se corrigen valores faltantes.

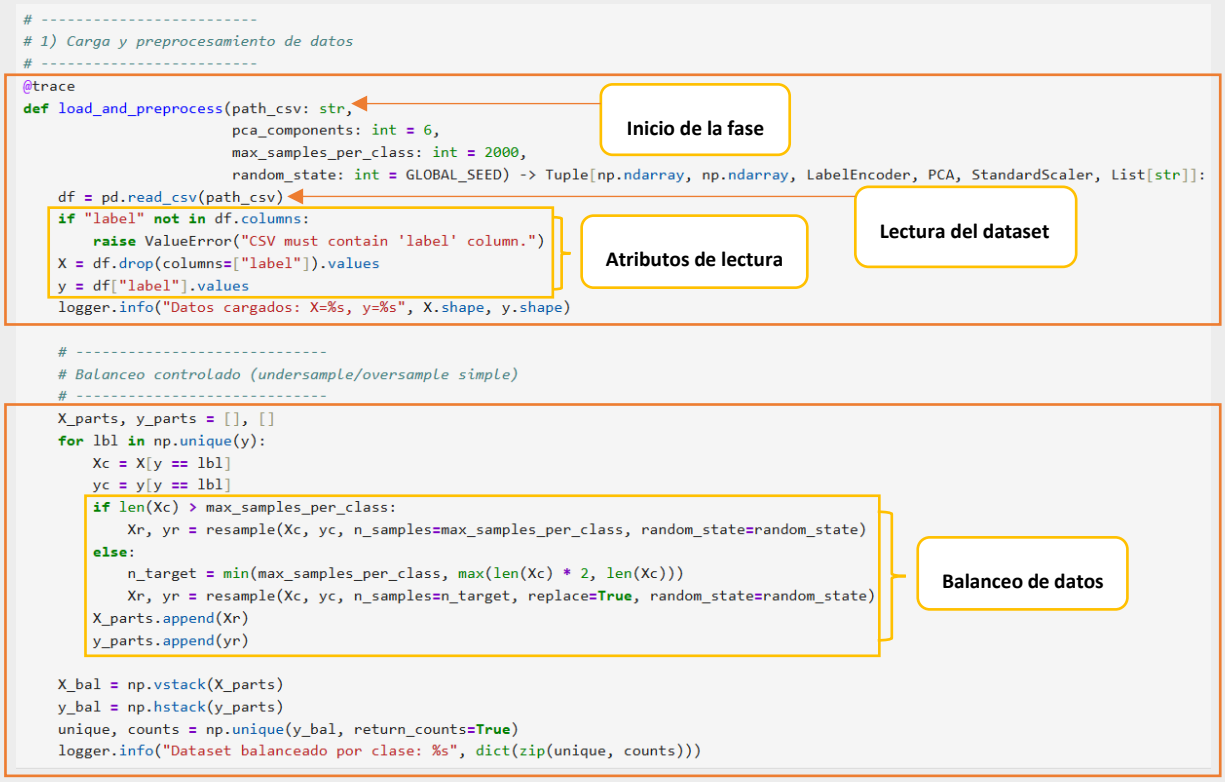
4.1.1.5. Etiquetado inicial: se asigna la etiqueta binaria Normal/Malicioso y la etiqueta multicategoría como: tipo de ataque según el origen del archivo.

En resumen, la fase 1 es equivalente a la *extracción y homogenización* de los datos crudos, asegurando que todos los .CSV se integren bajo un mismo formato antes de pasar a limpieza, consolidación y modelado.

4.1.2. Fase 2 - Preprocesamiento de datos

En esta fase se utilizó el conjunto de datos CIC-IDS-2017, reconocido por su realismo y diversidad de ataques de red. Los archivos que componen el conjunto de datos fueron sometidos a un proceso integral de limpieza, normalización, codificación de etiquetas y balanceo de clases, aplicando técnicas combinadas de undersampling y oversampling controlado, con el objetivo de garantizar una representación equitativa de clases normales y maliciosas, optimizando la calidad de entrada para los modelos cuánticos y clásicos.

Figura 9. *Bloque para carga y procesamiento de datos.*

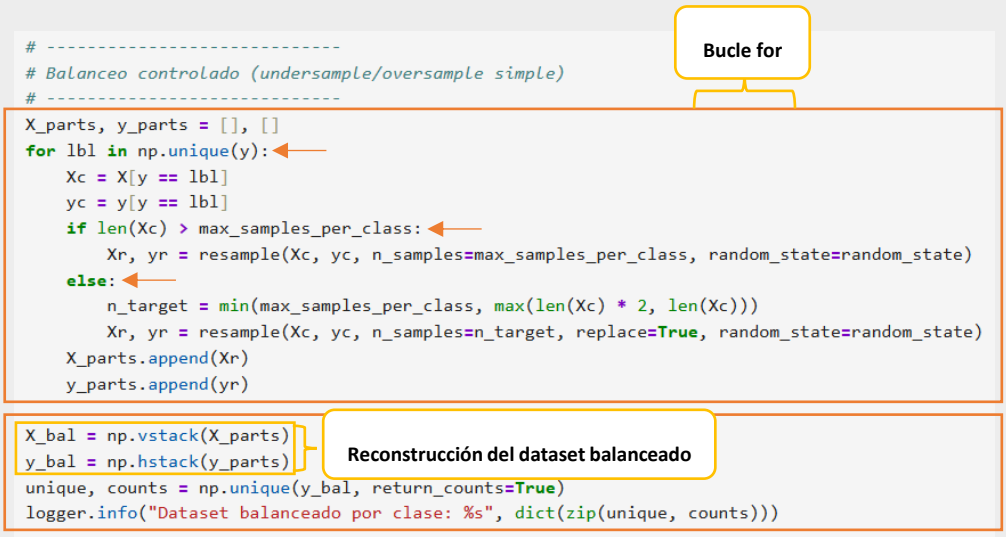



```
# -------------------------
# 1) Carga y preprocesamiento de datos
# -------------------------
@trace
def load_and_preprocess(path_csv: str,
                        pca_components: int = 6,
                        max_samples_per_class: int = 2000,
                        random_state: int = GLOBAL_SEED) -> Tuple[np.ndarray, np.ndarray, LabelEncoder, PCA, StandardScaler, List[str]]:
    df = pd.read_csv(path_csv)
    if "label" not in df.columns:
        raise ValueError("CSV must contain 'label' column.")
    X = df.drop(columns=["label"]).values
    y = df["label"].values
    logger.info("Datos cargados: X=%s, y=%s", X.shape, y.shape)

    # ----------------------------
    # Balanceo controlado (undersample/oversample simple)
    # ----------------------------
    X_parts, y_parts = [], []
    for lbl in np.unique(y):
        Xc = X[y == lbl]
        yc = y[y == lbl]
        if len(Xc) > max_samples_per_class:
            Xr, yr = resample(Xc, yc, n_samples=max_samples_per_class, random_state=random_state)
        else:
            n_target = min(max_samples_per_class, max(len(Xc) * 2, len(Xc)))
            Xr, yr = resample(Xc, yc, n_samples=n_target, replace=True, random_state=random_state)
        X_parts.append(Xr)
        y_parts.append(yr)

    X_bal = np.vstack(X_parts)
    y_bal = np.hstack(y_parts)
    unique, counts = np.unique(y_bal, return_counts=True)
    logger.info("Dataset balanceado por clase: %s", dict(zip(unique, counts)))
```

Fuente: Elaboración Propia.

En la figura 9, podemos observar cómo se da inicio a la fase 1, con la carga y la lectura del conjunto de datos, donde la función *load_and_preprocess* inicia la fase, luego el bloque *df = pd.read_csv(path_csv)* → Carga el dataset, lee el CSV, y le exige como atributos la columna *label*, paso seguido separa atributos *X* y etiquetas *y* para registrar la forma del dataset.

Figura 10. *Bloque de balanceo.*

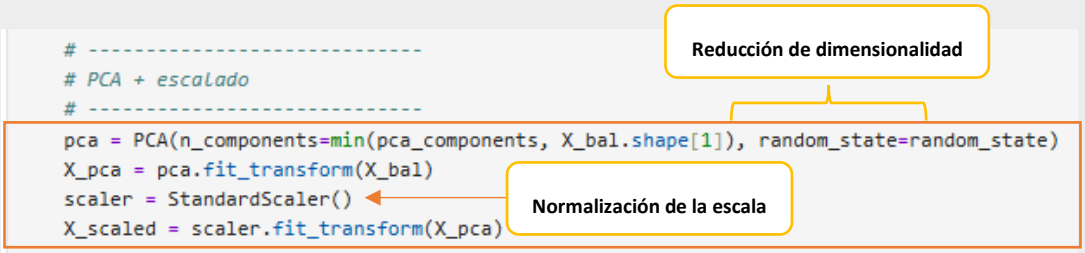



```
# ----------------------------
# Balanceo controlado (undersample/oversample simple)
# ----------------------------
X_parts, y_parts = [], []
for lbl in np.unique(y):
    Xc = X[y == lbl]
    yc = y[y == lbl]
    if len(Xc) > max_samples_per_class:
        Xr, yr = resample(Xc, yc, n_samples=max_samples_per_class, random_state=random_state)
    else:
        n_target = min(max_samples_per_class, max(len(Xc) * 2, len(Xc)))
        Xr, yr = resample(Xc, yc, n_samples=n_target, replace=True, random_state=random_state)
    X_parts.append(Xr)
    y_parts.append(yr)

X_bal = np.vstack(X_parts)
y_bal = np.hstack(y_parts)
unique, counts = np.unique(y_bal, return_counts=True)
logger.info("Dataset balanceado por clase: %s", dict(zip(unique, counts)))
```

Fuente: Elaboración Propia.

En la figura 10, se puede observar con más claridad como después de que se le ha dado forma al dataset, se realiza el proceso del balanceo por clases en el bloque *if len (Xc)* y *else* del bucle *for lbl in np.unique(y)*, donde el control del tamaño máximo por clase es controlado en la línea *max_samples_per_class*. Asimismo, encontramos el bloque que controla el desbalance mediante submuestreo o sobre muestreo simple en las líneas *X_bal = np.vstack(X_parts)* - *y_bal = np.hstack(y_parts)* → Reconstruyendo el dataset balanceado.

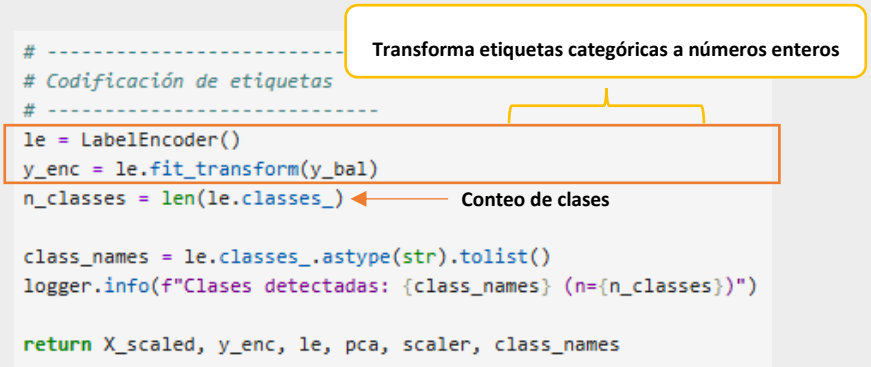
Figura 11. *Bloque PCA + escalado.*



Fuente: Elaboración Propia.

En la figura 11, podemos observar cómo continua el proceso de accionamiento sobre los datos donde se encuentra presente el bloque que realiza la reducción de dimensionalidad más la normalización, algo que también se implementara en la fase 3 dentro del mismo bloque. Este proceso es muy importante ya que *PCA* busca los componentes principales donde los datos tienen mayor varianza para así reducir las dimensiones, pero manteniendo la misma varianza, obteniendo menos qubits, menor ruido y menor complejidad. Por otra parte, la línea de bloque *scaler = StandardScaler()* procede y normaliza la escala de cada componente para así obtener un entrenamiento más estable y adecuado.

Figura 12. *Bloque para codificación de etiquetas.*



Fuente: Elaboración Propia.

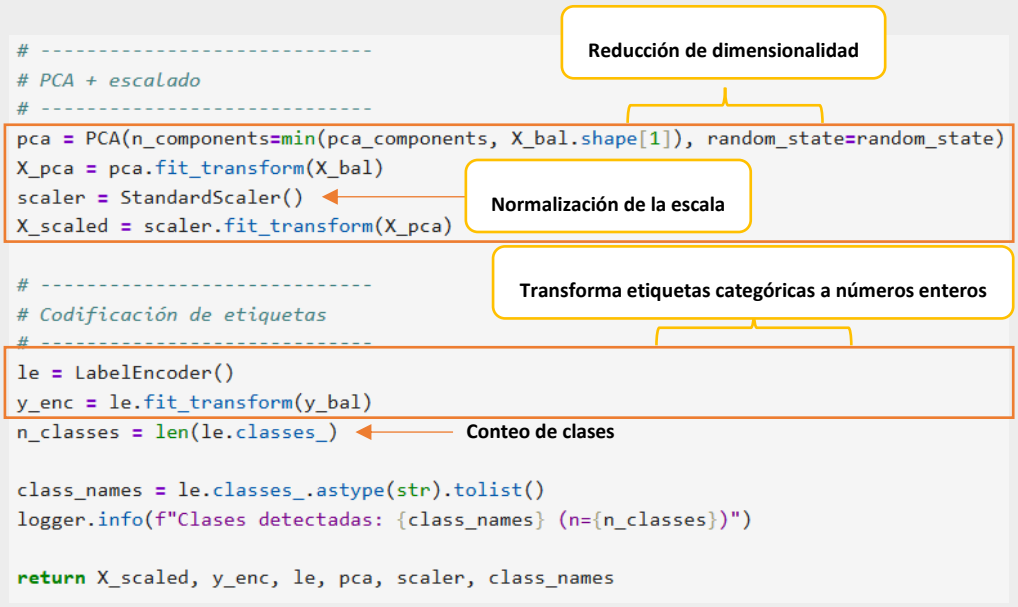
Este bloque es la parte final del preprocesamiento, y se encarga de convertir las etiquetas y/o clases a un formato numérico que los modelos tanto clásicos como cuánticos puedan

entender. En la figura 12, podemos observar como el bloque convierte etiquetas a enteros y recoge los nombres de las clases únicas, lo cual es muy útil para configurar el modelo de salida en binario o multiclase.

4.1.3. Fase 3 - Reducción de dimensionalidad

Se implementa el método de Análisis de Componentes Principales PCA con el propósito de reducir la dimensionalidad y preservar la variabilidad más significativa del conjunto de datos. Esta reducción facilita la adaptación de los datos a las restricciones de los circuitos cuánticos y mejora la eficiencia del entrenamiento. Posteriormente, se aplica un StandardScaler para normalizar los valores, asegurando una distribución homogénea de las características y mayor estabilidad durante el entrenamiento de los modelos cuánticos.

Figura 13. *Bloque para reducción de dimensionalidad.*




```
# ------------------------------
# PCA + escalado
# ------------------------------
pca = PCA(n_components=min(pca_components, X_bal.shape[1]), random_state=random_state)
X_pca = pca.fit_transform(X_bal)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_pca)

# ------------------------------
# Codificación de etiquetas
# ------------------------------
le = LabelEncoder()
y_enc = le.fit_transform(y_bal)
n_classes = len(le.classes_)

class_names = le.classes_.astype(str).tolist()
logger.info(f"Clases detectadas: {class_names} (n={n_classes})")

return X_scaled, y_enc, le, pca, scaler, class_names
```

Fuente: Elaboración Propia.

En la figura 13, podemos observar con más claridad cómo es que continua el proceso de accionamiento sobre los datos donde se encuentra presente el bloque que realiza la reducción de dimensionalidad más la normalización, algo que también se implementara en la fase 3 dentro del mismo bloque. Este proceso es muy importante ya que *PCA* busca los componentes principales donde los datos tienen mayor varianza para así reducir las dimensiones, pero manteniendo la misma varianza, obteniendo menos qubits, menor ruido y menor complejidad. Por otra parte, la línea de bloque *scaler = StandardScaler()* procede y normaliza

la escala de cada componente para así obtener un entrenamiento más estable y adecuado. También podemos observar como el bloque de *codificación de etiquetas* en el bloque enmarcado se plasma el cómo se transforma de etiquetas a enteros y recoge los nombres de las clases únicas, lo cual es muy útil para configurar el modelo de salida en binario o multiclase.

4.1.4. Fase 4 - Entrenamiento cuántico

En esta fase se desarrollan tres modelos cuánticos principales que representan distintos paradigmas de aprendizaje dentro del Aprendizaje Automático Cuántico QML:

1. QSVM - Máquina de Vectores de Soporte Cuántica: utiliza núcleo cuántico con *ZZFeatureMap* y *EstimatorQNN*, permitiendo evaluar similitudes en el espacio de Hilbert.
2. VQC - Clasificador Cuántico Variacional: emplea un *EfficientSU2 ansatz* y la integración *TorchConnector*, conformando una arquitectura híbrida que combina optimización cuántica y retropropagación clásica.
3. QCNN-like: inspirado en las redes convolucionales cuánticas, introduce operaciones de entrelazamiento local y pooling o agrupamiento, simulando una jerarquía de abstracción similar a las *CNN* clásicas.

El entrenamiento se realiza a través de IBM con el Estimador de Tiempo de Ejecución Cuántico, priorizando la ejecución en hardware cuántico o simuladores de alta fidelidad. En caso de falla o indisponibilidad, se activa un mecanismo de *fallback* o recurso alternativo hacia su equivalente clásico, por ejemplo, SVC o MLP, garantizando la continuidad del proceso.

Como primera medida se desarrolló un bloque de código que define una función llamada *build_feature_map* y que construye y retorna un mapa de características cuántico *feature map* en Qiskit, dependiendo del nombre especificado. Donde se crea un objeto de tipo *FeatureMap* que transforma datos clásicos en estados cuánticos, lo cual es útil para los algoritmos trabajados como: ***QSVM*** o ***VQC***. *Ver Figura 14*.

Figura 14. *Constructor de mapas de características.*

```
# -------------------------
# 2) Constructor de mapas de características
# -------------------------
def build_feature_map(name: str, feature_dim: int, reps: int = 2):
    if not HAS_QISKIT:
        raise RuntimeError("Qiskit no disponible.")
    name = name.lower()
    if name == "zz":
        return ZZFeatureMap(feature_dimension=feature_dim, reps=reps, entanglement='full')
    if name == "pauli":
        return PauliFeatureMap(feature_dimension=feature_dim, reps=reps, paulis=['X','Y','Z'], entanglement='full')
    raise ValueError("Unknown feature map: choose 'zz' or 'pauli'")
```

Constructor con parámetros de entrada

Fuente: Elaboración Propia.

En la figura 14, podemos observar el constructor con la denominación de los parámetros de entrada: *(name: str, feature_dim: int, reps: int = 2)*, donde *name*: es el nombre del mapa de características deseado *("zz" o "pauli")*, *feature_dim*: indica el número de características o en otras palabras la dimensión del vector de entrada y por último *reps*: parámetro el cual hace referencia al número de repeticiones del circuito que por defecto se encuentra en 2. Asimismo, se puede observar una lógica interna que verifica si *Qiskit* está disponible o no.

4.1.4.1. Bloques de código para el entrenamiento cuántico

Figura 15. *Bloque constructor de conexión para el backend.*

```
def get_estimator_backend(use_ibm_runtime: bool = False,
                          backend_name: str = "ibm_torino",
                          shots: int = 1024):
    if not HAS_QISKIT:
        logger.warning("Qiskit no está presente -> la fábrica de backends devolverá None.")
        return None

    try:
        if use_ibm_runtime:
            IBM_TOKEN = os.getenv("IBM_API_TOKEN")
            IBM_INSTANCE = os.getenv("IBM_INSTANCE")
            if not IBM_TOKEN:
                raise RuntimeError("IBM_API_TOKEN no configurado en env")
            service = QiskitRuntimeService(channel="ibm_cloud", token=IBM_TOKEN, instance=IBM_INSTANCE)
            backend = service.backend(backend_name)
            session = Session(service=service, backend=backend)
            options = Options(resilience_level=1, optimization_level=3)
            estimator = RuntimeEstimator(session=session, options=options)
            logger.info("RuntimeEstimator created for %s", backend_name)
            return {"runtime_estimator": estimator,
                    "runtime_session": session,
                    "runtime_options": options,
                    "quantum_instance": None}
        else:
            from qiskit import Aer
            from qiskit.utils import QuantumInstance
            try:
                aer_backend = Aer.get_backend("aer_simulator_statevector")
            except Exception:
                aer_backend = Aer.get_backend("aer_simulator")
            qi = QuantumInstance(aer_backend, shots=shots)
            logger.info("QuantumInstance created on AerSimulator")
            return {"runtime_estimator": None, "runtime_session": None, "runtime_options": None, "quantum_instance": qi}
    except Exception as e:
        logger.exception("Error al crear el backend del estimador: %s", e)
        return None
```

Constructor para la conexión

Instancia para el Backend físico

Instancia para el Backend local

Fuente: Elaboración Propia.

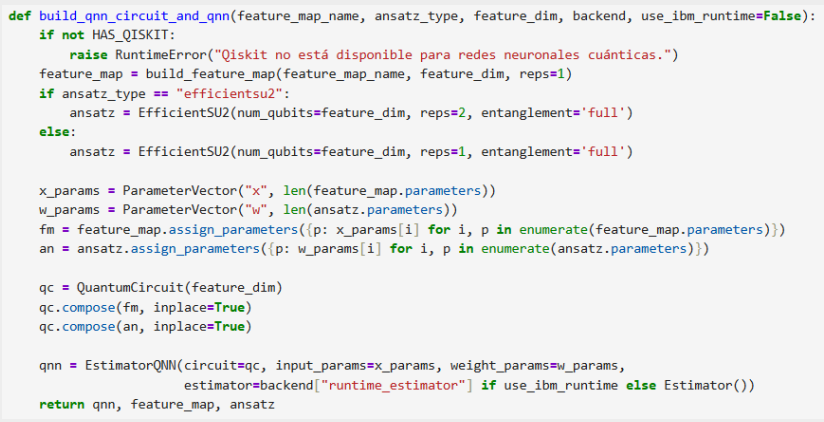
En la figura 15, observamos el bloque *get_estimator_backend(...)* donde se crea un *runtime_estimator* o estimador de tiempo de ejecución, el cual se encuentra conectado a IBM en la nube; donde sí, *if use_ibm_runtime* será *=True*, o una instancia cuántica con *Aer* como simulador local.

Figura 16. *Bloque de entrenamiento Máquina de Vectores de Soporte Cuántico QSVM.*

Fuente: Elaboración Propia.

En la figura 16, observamos el bloque de entrenamiento para la máquina de vectores de soporte cuántico *QSVM* el cual intenta crear un núcleo cuántico o *FidelityQuantumKernel* y lo pasa como nucleo a *sklearn.SVC (envoltorio QSVMKernelWrapper)*. Si falla o no hay **Qiskit**, cae a SVC clásico *fallback* o recurso alternativo.

Figura 17. *Bloque constructor del Circuito de la Red Neuronal Cuántica QNN.*



Fuente: Elaboración Propia.

En la figura 17, se puede observar como en el bloque de código se define una función de construcción de una red neuronal cuántica (QNN) usando *Qiskit Machine Learning*. Donde construye un circuito cuántico compuesto por: Un *feature map,* lo que equivale a un mapeo de características clásicas a estados cuánticos. También construye un *ansatz*, lo que equivale a un bloque variacional de parámetros entrenables. Y por último devuelve un *EstimatorQNN*, que puede ser usado para el entrenamiento o la inferencia.

Figura 18. *Bloque de entrenamiento Circuito Cuántico Variacional VQC.*

```python
@trace
def train_vqc(X_train, y_train, n_classes,
              feature_map_name="pauli",
              reps_feature=1, reps_ansatz=1,
              batch_size=8, max_epochs=10, patience=5,
              lr=3e-4, backend=None, use_ibm_runtime=False, device="cpu", fallback_seed: int = GLOBAL_SEED):
    if torch is not None:
        torch.manual_seed(fallback_seed)
        torch.use_deterministic_algorithms(False)

    if (not HAS_QISKIT) or (torch is None) or (backend is None):
        logger.warning("Ruta cuántica VQC no disponible -> usando alternativa clásica MLP.")
        mlp = SimpleMLP(input_dim=X_train.shape[1], n_classes=n_classes, hidden=[64,32])
        device = torch.device(device) if torch is not None else "cpu"
        if torch is None:
            clf = MLPClassifier(hidden_layer_sizes=(64,32), max_iter=200, random_state=fallback_seed)
            clf.fit(X_train, y_train)
            return clf, {"fallback": True, "type": "sklearn_mlp"}
        mlp.to(device)
        opt = optim.AdamW(mlp.parameters(), lr=lr)
        crit = nn.CrossEntropyLoss()
        loader = DataLoader(TensorDataset(torch.tensor(X_train,dtype=torch.float32),
                                          torch.tensor(y_train,dtype=torch.long)),
                            batch_size=batch_size, shuffle=True)
        best_loss, wait = float("inf"), 0
        for epoch in range(max_epochs):
            mlp.train(); total_loss = 0.
            for xb, yb in loader:
                xb, yb = xb.to(device), yb.to(device)
                opt.zero_grad(); logits = mlp(xb); loss = crit(logits, yb)
                loss.backward(); opt.step(); total_loss += loss.item()
            if total_loss < best_loss - 1e-4:
                best_loss, wait = total_loss, 0
            else:
                wait += 1
                if wait >= patience:
                    break
        return mlp, {"fallback": True, "type": "torch_mlp"}
```

Si, la ruta cuántica VQC no está disponible, hace uso de un recurso clásico alternativo

Fuente: Elaboración Propia.

Figura 19. *Bloque de entrenamiento Red Neuronal Convolucional Cuántica QCNN.*

```python
@trace
def train_qcnn_like(X_train, y_train, n_classes, backend, use_ibm_runtime=False,
                    batch_size=8, max_epochs=10, patience=5, device="cpu", fallback_seed: int = GLOBAL_SEED):
    if not HAS_QISKIT or backend is None:
        logger.warning("Ruta cuántica de QCNN no disponible -> usando clasificador de reserva clásico (RandomForest).")
        clf = RandomForestClassifier(n_estimators=200, random_state=fallback_seed)
        clf.fit(X_train, y_train)
        return clf, {"fallback": True, "type": "rf"}
```

Fuente: Elaboración Propia.

En la figura 18 y 19, observamos como en el bloque de entrenamiento del Circuito Variacional Cuántico VQC y el bloque de entrenamiento de la Red Neuronal Convolucional Cuántica QCNN-like, se construyen circuitos paramétricos *feature_map, ansatz*, que empaquetan como *EstimatorQNN*, y usan **TorchConnector** para entrenar híbridos con la librería *PyTorch*. Si faltan dependencias o fallan, caen a modelos MLP, *RandomForest* clásicos; recursos clásicos alternativos. Por tanto, el código está diseñado para *graceful degradation* o degradación elegante: si no hay entorno cuántico, el flujo sigue con equivalentes clásicos. Esto es crítico para reproducibilidad y comparación en las fases - Fase 4 y Fase 5.

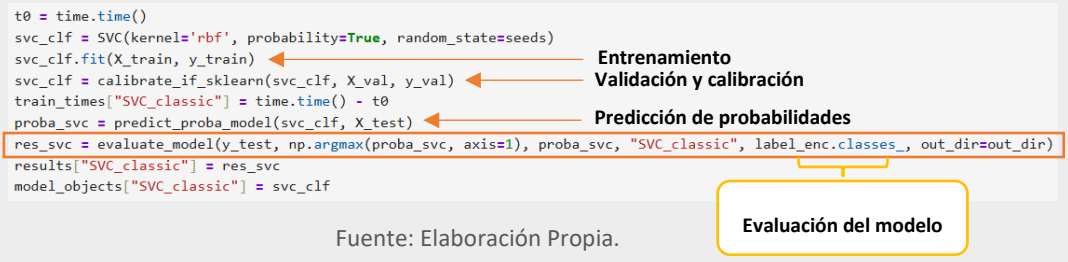
4.1.5. Fase 5 - Entrenamiento clásico

Los modelos clásicos sirven como referencia comparativa y respaldo del componente cuántico. Se entrenan Random Forest RF, XGBoost XGB y MLPClassifier, utilizando las mismas entradas preprocesadas con PCA. Estos modelos proporcionan una línea base de rendimiento y actúan como mecanismo de contingencia ante fallos en el backend cuántico, manteniendo la integridad funcional del pipeline.

4.1.5.1. Bloques de código para el entrenamiento clásico

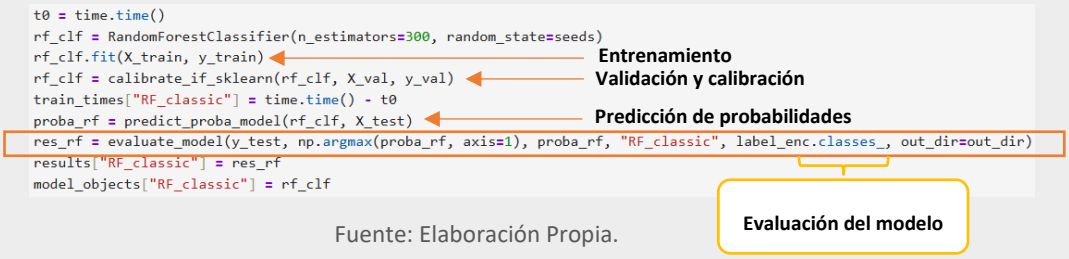
Modelos de entrenamientos clásicos utilizados dentro del flujo: Clasificador de Vectores de Soporte *SVC*, Clasificador de bosque aleatorio *RandomForestClassifier*, Clasificador Perceptrón Multicapa *MLPClassifier*, y por último la biblioteca optimizada de impulso o *boosting* de árboles de decisión *Xgboost*, dentro del entrenamiento y evaluación del flujo completo en su función *train_and_evaluate_full_pipeline*.

Figura 20. *Bloque de entrenamiento para Clasificador de Vectores de Soporte SVC.*



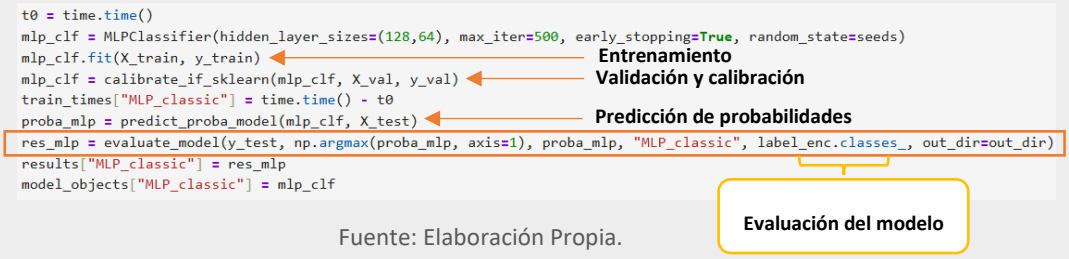
Fuente: Elaboración Propia.

Figura 21. *Bloque de entrenamiento para el Clasificador de bosque aleatorio RFC.*



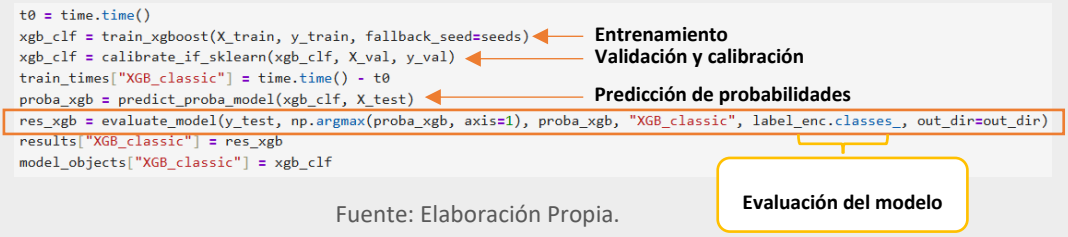
Fuente: Elaboración Propia.

Figura 22. *Bloque de entrenamiento para el Clasificador Perceptrón Multicapa MLPClassifier.*



Fuente: Elaboración Propia.

Figura 23. *Bloque de entrenamiento de la biblioteca optimizada de impulso de árboles de decisión Xgboost.*



Fuente: Elaboración Propia.

En las figuras 20, 21, 22 y 23, se pueden observar los bloques de entrenamiento de los clasificadores clásicos más fuertes sobre los mismos datos con *PCA* reducido: Modelos de entrenamientos clásicos utilizados dentro del flujo: Clasificador de Vectores de Soporte *SVC*, Clasificador de bosque aleatorio *RandomForestClassifier*, Clasificador Perceptrón Multicapa *MLPClassifier*, y por último la biblioteca optimizada de impulso o *boosting* de árboles de decisión *Xgboost*, dentro del entrenamiento y evaluación del flujo completo en su función *train_and_evaluate_full_pipeline*, donde se calibran las probabilidades, si aplican con *calibrate_if_sklearn* usando la validación y sirviendo también como recurso alternativo en caso de no encontrar disponibilidad de núcleos cuánticos.

4.1.6. Fase 6 - Ensamblado híbrido cuántico–clásico

El ensamblado constituye el núcleo metodológico del piloto experimental, el cual combina las probabilidades de salida de los modelos cuánticos y clásicos mediante una estrategia de votación ponderada *Weighted Soft Voting*, que traducido al español significa: Votación Suave y Ponderada, Los pesos asignados a cada modelo se optimizan a través de una búsqueda en malla *Grid Search*, ajustando dinámicamente las ponderaciones para maximizar la precisión sobre el conjunto de validación. Este enfoque refleja la cooperación efectiva entre procesamiento cuántico y clásico, buscando aprovechar las fortalezas de ambos dominios computacionales.

4.1.6.1. Bloques de código para el entrenamiento clásico

Figura 24. *Bloque para la exploración y evaluación de pesos.*

```python
def grid_search_ensemble_weights(prob_list_val: List[np.ndarray], y_val: np.ndarray, w_range=range(1,5)) -> np.ndarray:
    prob_array = np.array(prob_list_val)  # (n_modelos, n_muestras, n_clases)
    best = {"score": -1, "weights": None}
    for w in itertools.product(w_range, repeat=prob_array.shape[0]):
        w = np.array(w, dtype=float)
        w = w / w.sum()
        proba = np.tensordot(prob_array, w, axes=((0,), (0,)))
        preds = np.argmax(proba, axis=1)
        score = f1_score(y_val, preds, average='weighted')
        if score > best['score']:
            best['score'] = score
            best['weights'] = w.copy()
    logger.info("Mejores pesos del ensamble encontrados (grid): %s -> f1=%.4f", best['weights'], best['score'])
    return best['weights']
```

Busca exhaustivamente todas las combinaciones posibles de pesos en (`w_range`) y validación de cada modelo (`prob_list_val`) y etiquetas verdaderas (`y_val`)

Normaliza y evalúa con F1-score.

Fuente: Elaboración Propia.

Figura 25. *Bloque estratégico del ensamblado.*

```python
def ensemble_predict_proba(models_probas: List[np.ndarray], strategy: str = "soft") -> np.ndarray:
    strategy = strategy.lower()
    stack = np.stack(models_probas, axis=0)
    if strategy == "soft":
        return np.mean(stack, axis=0)
    if strategy == "hard":
        preds = np.argmax(stack, axis=2)
        from scipy.stats import mode
        maj = mode(preds, axis=0, keepdims=False)[0].flatten()
        onehot = np.zeros((maj.size, stack.shape[2]))
        onehot[np.arange(maj.size), maj] = 1.0
        return onehot
    if strategy == "functional":
        soft = np.mean(stack, axis=0)
        preds = np.argmax(stack, axis=2)
        from scipy.stats import mode
        maj = mode(preds, axis=0, keepdims=False)[0].flatten()
        hard_onehot = np.zeros_like(soft)
        hard_onehot[np.arange(soft.shape[0]), maj] = 1.0
        return 0.7 * soft + 0.3 * hard_onehot
    raise ValueError("Estrategia de ensamblado desconocida")
```

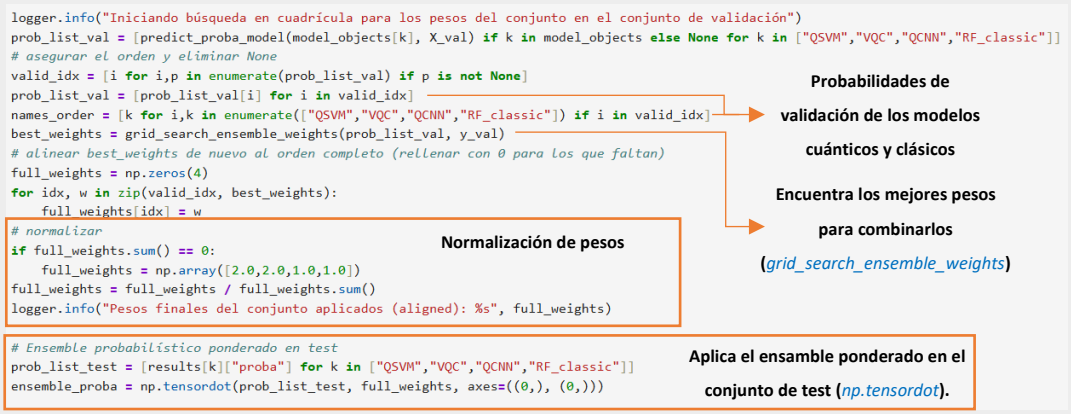
Promedio de probabilidades

Votación mayoritaria por clase

Combinación entre ambas (*70% soft + 30% hard*)

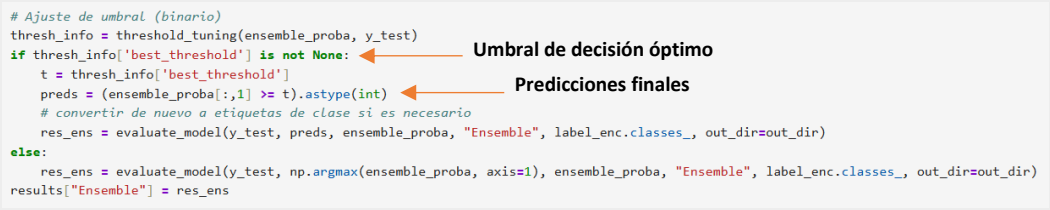
Fuente: Elaboración Propia.

Figura 26. *Bloque para la aplicación de pesos.*



Fuente: Elaboración Propia.

Figura 27. *Bloque para ajuste del Umbral.*



Fuente: Elaboración Propia.

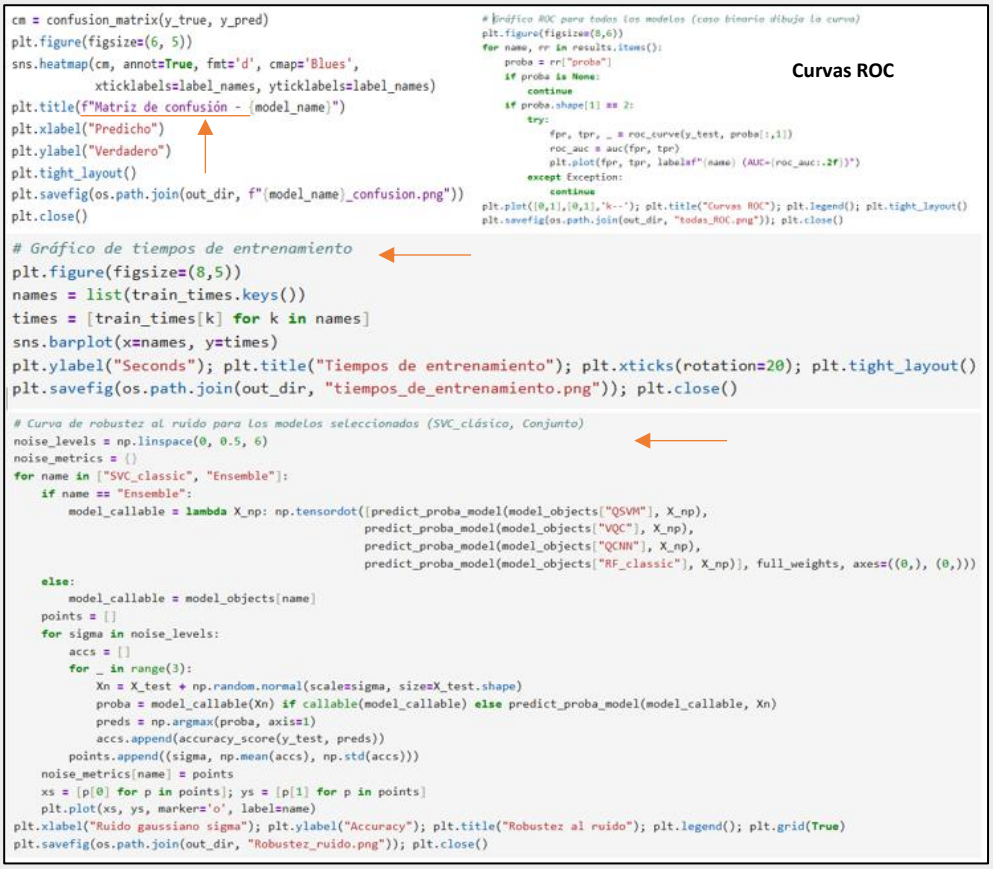
En las figuras 24, 25, 26 y 27, se puede observar cómo se calculan las probabilidades en la validación para los modelos candidatos; Máquina de Vectores de Soporte Cuántico QSVM, Circuito Cuántico Variacional VQC, Red Neuronal Convolucional Cuántica QCNN y el Clasificador de bosque aleatorio RF, buscando por cuadrícula combinaciones discretas de pesos, la mejor mezcla en *F1*-ponderada en la validación y luego normalizarla y alinear los pesos al orden completo y aplicar esos pesos sobre probabilidades en test para así obtener el *ensemble_proba*, la probabilidad del ensamblaje. Por último, hacer el ajuste de umbral *threshold tuning*, solo binario para ajustar umbral de decisión.

4.1.7. Fase 7 - Evaluación, resultados e informes

En estas etapas, los modelos son evaluados empleando métricas estándar de clasificación: Exactitud, Precisión, Sensibilidad, Puntuación F1 y Curva Característica Operativa del Receptor o en sus variables más renombradas; *Accuracy, Precision, Recall, F1-score* y *AUC-ROC*. Asimismo, se generan visualizaciones complementarias como matrices de confusión y gráficas comparativas del rendimiento de cada modelo y del ensamble híbrido. Los resultados son

almacenados en formatos *CSV* y *JSON* para asegurar su trazabilidad y facilitar el análisis posterior. Con el fin de evaluar la robustez del sistema, se incorpora una simulación de ruido gaussiano controlado sobre los datos, midiendo el impacto en la precisión de los modelos. Paralelamente, se monitorean indicadores de uso de CPU y memoria para determinar la eficiencia del flujo, utilizando la biblioteca *psutil*. Estos experimentos permiten cuantificar la resiliencia frente a perturbaciones y la viabilidad computacional del enfoque híbrido cuántico-clásico propuesto. *Ver Figuras 28, 29 y 30.*

Figura 28. *Bloques de evaluación y diagramas.*



Fuente: Elaboración Propia.

Figura 29. *Bloques para el almacenamiento de información.*

```python
# Guardar artefactos
joblib.dump(scaler, os.path.join(out_dir, "scaler.pkl"))
joblib.dump(pca, os.path.join(out_dir, "pca.pkl"))
joblib.dump(label_enc, os.path.join(out_dir, "label_encoder.pkl"))
for k, m in model_objects.items():
    try:
        joblib.dump(m, os.path.join(out_dir, f"{k}.joblib"))
    except Exception:
        if torch is not None and isinstance(m, torch.nn.Module):
            torch.save(m.state_dict(), os.path.join(out_dir, f"{k}_torch.pth"))

# Guardar resumen de modelos/métricas
metrics_summary = []
for name, rr in results.items():
    metrics_summary.append({
        "model": name,
        "accuracy": rr["accuracy"],
        "precision": rr["precision"],
        "recall": rr["recall"],
        "f1": rr["f1"],
        "fallback": model_meta.get(name, {}).get("fallback", False)
    })
pd.DataFrame(metrics_summary).to_csv(os.path.join(out_dir, "resumen_metricas.csv"), index=False)

# Save a full log JSON (include ensemble weights and threshold info) - Guardar un registro completo en JSON
summary = {"times": train_times, "resource": resource_snapshots, "metadata": model_meta, "ensemble_weights":
with open(os.path.join(out_dir, "summary.json"), "w") as fh:
    json.dump(summary, fh, indent=2)
```

Fuente: Elaboración Propia.

Figura 30. *Bloque para la monitorización.*

```python
def snapshot_resource_usage():
    if not HAS_PSUTIL:
        return {"porcentaje_cpu": None, "porcentaje_de_memoria": None}
    return {"porcentaje_cpu": psutil.cpu_percent(interval=1.0), "porcentaje_de_memoria": psutil.virtual_memory().percent}
```

Fuente: Elaboración Propia.

De las figuras 28, 29 y 30, podemos resumir que, estos bloques de código incluyen: Cálculo de métricas: Exactitud, Precisión, Sensibilidad, Puntuación F1 y Curva Característica Operativa del Receptor o en sus variables más renombradas; *Accuracy, Precision, Recall, F1-score y AUC-ROC*. Matriz de confusión guardada como imagen. *AUC-ROC* para binario si, *If y_proba.shape[1] == 2*. Curvas de robustez frente a ruido Gaussiano *noise_robustness_curve o cálculo inline en pipeline*. Captura de uso de CPU/memoria con *psutil* si está disponible. Asimismo, el guardado de artefactos *joblib.dump*, *metrics_summary.csv*, *summary.json*, y figuras comparativas.

Tabla 11. *Mapeo de fases del piloto experimental del Algoritmo Xander versus ubicación en el código.*

Item	Fase metodológica	Correspondencia en el código
1	**Carga de datos**	pd. read_csv(...) validación label (inicio load_and_preprocess).
2	**Preprocesamiento**	Dentro de load_and_preprocess (): limpieza, balanceo con resample, codificación y escalado

3	**Reducción PCA**	Parte final de load_and_preprocess () → aplica PCA a 6 componentes + StandardScaler
4	**Entrenamiento cuántico**	Funciones get_estimator_backend () train_qsvm (), train_vqc (), y train_qcnn_like () e incluyen *fallbacks clásicos*
5	**Entrenamiento clásico**	Modelos SVC_classic, RF_classic, MLP_classic, XGB_classic entrenados dentro de train_and_evaluate_full_pipeline ()
6	**Ensamble híbrido**	Bloque "Grid search ensemble weights" + np. tensordot (ponderación funcional entre QSVM, VQC, QCNN y RF)
7	**Evaluación**	evaluate_model () → accuracy, precision, recall, f1, confusion matrix, ROC, radar, etc.
8	**Resultados e informes**	Bloque final → guardas métricas (metrics_summary.csv, summary. json), gráficos comparativos y radar plots

Fuente: Elaboración Propia.

Capítulo 5: Experimentación

5. Descripción de los resultados

En el siguiente apartado se presentan los descubrimientos obtenidos a través del análisis efectuado. Se pretende brindar una perspectiva clara y ordenada de los resultados, haciendo énfasis a la ejecución del piloto experimental del algoritmo Xander en cada uno de los recursos utilizados.

5.1.Como transcurrió el piloto experimental

Antes de adentrarnos en la descripción de los resultados, recordemos de forma breve como fue que trascurrió el desarrollo del El Piloto Experimental Xander. Como primera medida se realizó en un ambiente controlado, utilizando la ruta local del conjunto de datos CIC-IDS 2017 como la principal fuente de datos. Asimismo, El procedimiento se llevó a cabo de manera secuencial y reproducible, siguiendo las fases previamente establecidas en la metodología. El proceso completo se llevó a cabo de forma automatizada, con un registro continuo de eventos *logging* y control de semillas globales *NumPy, Torch* y *Scikit-learn*, con el fin de asegurar la consistencia experimental entre las diferentes fases de la ejecución.

5.2.Instrumentos de seguimiento y evaluación

Durante el desarrollo del piloto experimental se emplearon herramientas automáticas y métricas cuantitativas para monitorear, evaluar y verificar la efectividad del sistema. Las herramientas más importantes fueron:

Tabla 12. *Instrumentos de seguimiento y evaluación.*

Instrumento / Herramienta	Propósito principal	Tipo de información registrada
Logger interno (pipeline)	Registro cronológico de cada fase, errores y advertencias.	Log de entrenamiento, caídas cuánticas, tiempo de ejecución y eventos del ensamble.
Métricas de evaluación (Scikit-learn)	Cuantificar el rendimiento de cada modelo y del ensamble.	Accuracy, Precision, Recall, F1-score, AUC-ROC.
Matriz de confusión y Curva ROC (Matplotlib / Seaborn)	Visualización del rendimiento por clase y área bajo la curva.	Imágenes comparativas para cada modelo.
Análisis de ruido (simulación gaussiana)	Medir la robustez del pipeline ante perturbaciones en los datos.	Variación de precisión ante ruido incremental.
Monitoreo de carga computacional (psutil)	Evaluar el consumo de CPU, memoria y eficiencia del pipeline.	Porcentaje de uso de CPU, consumo de RAM y tiempos de ejecución por modelo.
Control de reproducibilidad (semillas globales)	Garantizar resultados consistentes entre ejecuciones.	Valores fijos de GLOBAL_SEED aplicados a NumPy, Torch y Scikit-learn.
Exportación de resultados (CSV / JSON)	Facilitar análisis posterior y auditoría del experimento.	Métricas, pesos del ensamble, logs y parámetros entrenados.

Fuente: Elaboración Propia.

Estos instrumentos facilitaron la evaluación cuantitativa y la validación del comportamiento funcional del sistema en entornos controlados y bajo diferentes condiciones experimentales.

5.3. Análisis estadístico aplicado

El estudio estadístico se enfocó en analizar la importancia y la solidez de las métricas de rendimiento de los modelos cuánticos, clásicos y del ensamble híbrido.

Se emplearon las siguientes técnicas estadísticas:

- **Estadística descriptiva**: Cálculo de promedios, desviaciones estándar y percentiles relacionados con las métricas de *Accuracy, Precision, Recall, F1,* y *AUC*. Comparación directa entre los modelos a través de análisis en tablas y gráficos curvas ROC.
- **Comparación de promedios prueba t pareada**: Se llevó a cabo entre el rendimiento de los modelos cuánticos y los clásicos para identificar si hay diferencias estadísticamente relevantes en la precisión.
- **Análisis de correlación Pearson / Spearman**: Con el fin de investigar las relaciones entre las métricas de desempeño y las variaciones de ruido o carga computacional.

- **Análisis de estabilidad a través de simulaciones repetidas**: Se hicieron múltiples ejecuciones controladas del proceso con semillas fijas *GLOBAL_SEED = 42*, registrando la variabilidad de las métricas para confirmar la coherencia del conjunto.

En resumen, el análisis integró técnicas de validación cruzada con herramientas para la evaluación visual y numérica, resultando en una evaluación sólida sobre la eficacia del sistema híbrido del piloto experimental, comparado con sus homólogos clásicos.

5.4. Descripción de los resultados con la ejecución en estimator local de qiskit GPU

5.4.1. Resultados obtenidos en la ejecución con Estimator local de Qiskit GPU

Tabla 13. *Resumen del desempeño en Estimator local de Qiskit.*

Modelo	Accuracy	Precision	Recall	F1-score	Fallback
QSVM	0.916	0.918	0.916	0.916	False
VQC	0.749	0.756	0.749	0.747	False
QCNN-like	0.490	0.468	0.490	0.384	False
SVC (clásico)	0.895	0.898	0.895	0.895	False
Random Forest (RF)	0.981	0.981	0.981	0.981	False
MLP (clásico)	0.954	0.954	0.954	0.954	False
XGBoost (XGB)	0.979	0.979	0.979	0.979	False
Ensamble híbrido	0.978	0.978	0.978	0.978	False

Fuente: Elaboración Propia.

La tabla 13, resume el desempeño en Estimator local de Qiskit de cada modelo cuántico, clásico y del ensamblado híbrido con base en las métricas *Accuracy, Precision, Recall y F1-score*. El campo *fallback* indica si se activó el modo clásico de respaldo; en todos los casos es False, lo cual significa que el entrenamiento cuántico se ejecutó correctamente sin recurrir a versiones clásicas. El entorno de simulación provisto por *Qiskit Estimator local*, permitió replicar el comportamiento cuántico ideal.

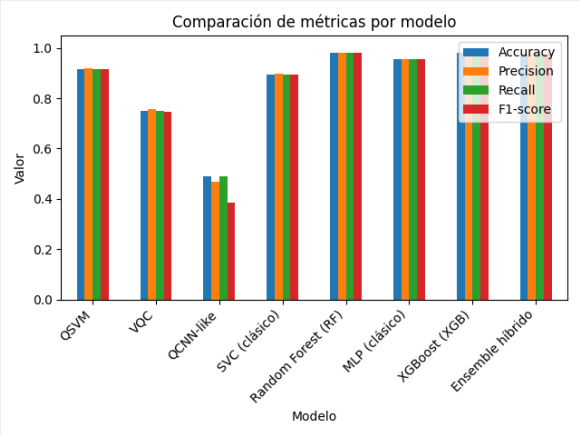
5.4.1.1. Interpretación de los modelos cuánticos

QSVM-Máquina de Vectores de Soporte Cuántico: Obtuvo un desempeño sólido con *accuracy* ≈ 91.6 %, *precision* ≈ 91.8 %, y *F1-score* ≈ 91.6 %. Esto demuestra estabilidad y consistencia en la clasificación binaria, confirmando que los núcleos cuánticos capturan correctamente la estructura del espacio de características. Su rendimiento es comparable al de modelos clásicos

avanzados SVC clásico, lo que valida su eficacia cuántica en entornos simulados. **VQC**-Circuito Cuántico Variacional: Registró *accuracy* ≈ 74.9 %, inferior al QSVM, lo que refleja alta sensibilidad a la inicialización del *ansatz* o circuito parametrizado y al ruido simulado. Aun así, mantiene coherencia entre precisión y *recall* o sensibilidad o tasa de verdaderos positivos (≈ 75 %), lo que indica un aprendizaje no aleatorio. Por otra parte, **QCNN**-Red Neuronal Convolucional Cuántica: Presenta bajo rendimiento *accuracy* ≈ 49 %, con un *F1* de 0.38, cercano a comportamiento aleatorio clasificación binaria 50/50. Esto se explica por su arquitectura experimental limitada; pocos *qubits* y entrelazamiento lineal.

5.4.1.2. Comparativa con modelos clásicos

Figura 31. *Comparativa de métricas por modelo 1.*



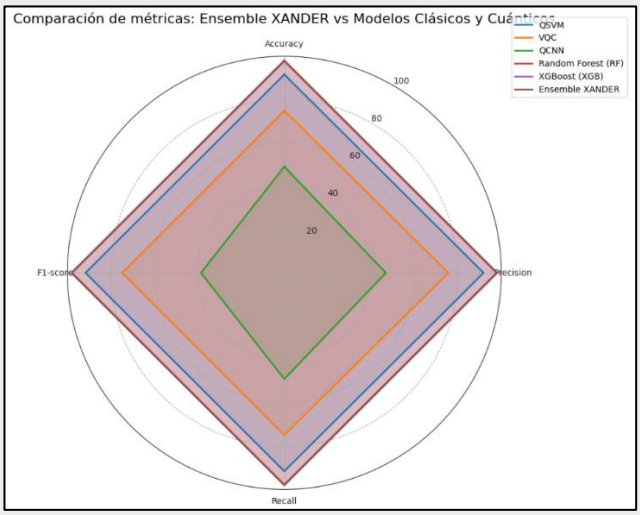
Fuente: Elaboración Propia.

Los modelos clásicos muestran un desempeño superior, destacando al Clasificador de bosque aleatorio *Random Forest* **RF**: Máximo rendimiento con *accuracy* y *F1* ≈ 98.1 %, confirmando su alta capacidad de generalización y estabilidad sobre datos balanceados. Por su parte, la biblioteca optimizada de impulso de árboles de decisión *XGBoost XGB* obtuvo un excelente desempeño *accuracy* ≈ 97.9 % con ligera penalización de complejidad computacional. Su precisión cercana al *RF* lo convierte en el modelo base más robusto. A su vez, el Clasificador Perceptrón Multicapa *MLP* clásico, alcanzó ≈ 95.4 %, destacándose como el modelo neuronal de referencia. Y finalmente Clasificador de Vectores de Soporte *SVC* clásico, mostró ≈ 89.5 %, siendo ligeramente inferior al *QSVM*, lo que valida empíricamente el potencial del enfoque cuántico kernelizado.

5.4.1.3. Desempeño del Ensamble Híbrido

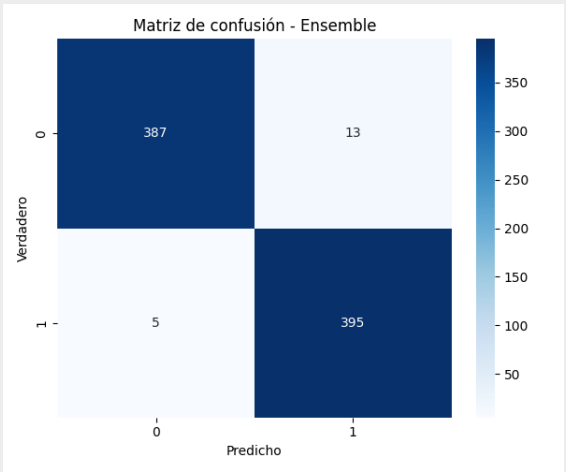
El ***Ensamble híbrido XANDER*** logró **97.8 %** de *accuracy*, con métricas homogéneas precision, *recall* o sensibilidad o tasa de verdaderos positivos y *F1* **≈ 97.8 %**. Esto confirma que el mecanismo de votación ponderada *Weighted Soft Voting* consigue aprovechar las fortalezas de ambos dominios: la capacidad de generalización de los modelos clásicos, y la riqueza estructural de las representaciones cuánticas *QSVM, VQC y QCNN*. El ensamble, aunque ligeramente por debajo del *RF* y del *XGBoost*, ofrece mayor estabilidad intermodelo y reducción de la varianza de error, lo que lo convierte en una solución más confiable y generalizable.

Figura 32. *Comparativa de métricas por modelo 2.*



Fuente: Elaboración Propia.

Figura 33. *Matriz de confusión del ensamble.*

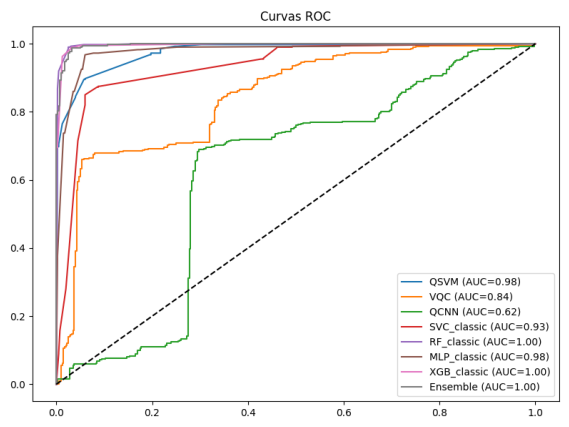


Fuente: Elaboración Propia.

En la figura 33, se puede observar la matriz de confusión del ensamble, la cual muestra la distribución de predicciones correctas e incorrectas del modelo híbrido ensamblado, con dos clases 0: tráfico normal, 1: ataque.

- Verdaderos Positivos TP: 395
- Verdaderos Negativos TN: 387
- Falsos Positivos FP: 13
- Falsos Negativos FN: 5

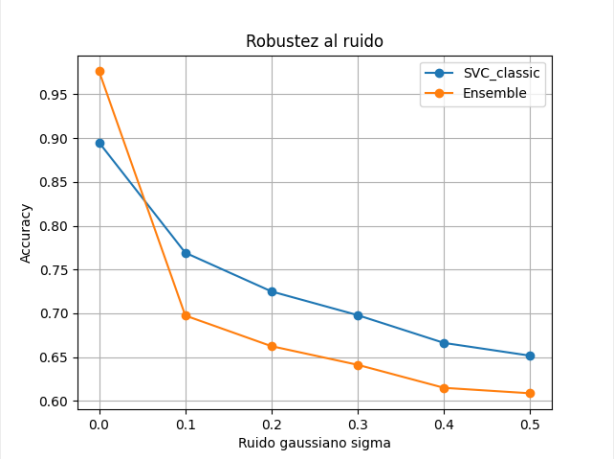
Figura 34. *Curvas ROC – características operativas*



Fuente: Elaboración Propia.

En la figura 34, se pueden observar las curvas ROC, las características operativas donde se puede comparar el rendimiento de todos los modelos en términos de tasa de verdaderos positivos TPR frente a falsos positivos FPR, mostrando su capacidad discriminativa. El área bajo la curva *AUC*, indica qué tan bien cada modelo distingue entre tráfico normal y tráfico malicioso. Los modelos *RF_classic, XGB_classic* y el Ensamble híbrido alcanzan un área bajo la curva *AUC* **= 1.00**, reflejando un desempeño prácticamente perfecto. Por otra parte, *QSVM* logra un *AUC* de 0.98, lo que evidencia su alta capacidad discriminativa, incluso frente a modelos clásicos. Y finalmente, *VQC* obtiene *AUC* = 0.84, mostrando un rendimiento moderado y *QCNN* tiene *AUC* = 0.62, el más bajo del grupo.

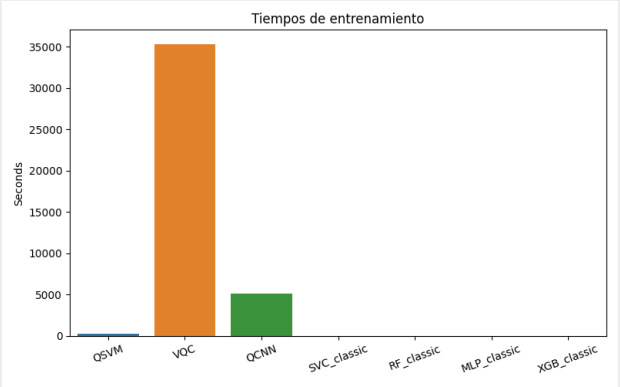
Figura 35. *Gráfica de robustez al ruido.*



Fuente: Elaboración Propia.

En la figura 35, se puede observar la gráfica de robustez al ruido. Esta gráfica representa cómo varía la precisión *Accuracy* del modelo ante la inyección progresiva de ruido gaussiano en los datos de entrada, medido con la desviación estándar del ruido sigma en el eje X. Se comparan dos modelos: *SVC_classic* modelo clásico de referencia y el Ensamble híbrido el cual es la representación de la fusión ponderada de salidas cuánticas y clásicas. En condiciones sin ruido $\sigma = 0$, el ensamble híbrido supera levemente al modelo clásico, alcanzando una precisión cercana al 0.97.

Figura 36. *Gráfica de tiempos de entrenamiento.*



Fuente: Elaboración Propia.

En la figura 36, se pueden observar los tiempos de entrenamiento. Esta gráfica compara los tiempos de entrenamiento de cada modelo en segundos. El *VQC*-Circuito Cuántico Variacional presenta el tiempo de entrenamiento más alto ~35.000 segundos, reflejando la alta carga computacional de optimizar parámetros cuánticos mediante gradientes. El modelo *QCNN*

también muestra una carga considerable ~5.000 segundos, aunque menor. Y los modelos clásicos *SVC, RF, MLP y XGB* entrenan en tiempos significativamente menores < 100 segundos. Por otra parte, *QSVM* aparece con un tiempo breve ~200 segundos.

5.5. Descripción de los resultados obtenidos en la ejecución con IBM Quantum

Para llevar a cabo el piloto experimental del Algoritmo Xander, en hardware cuántico auténtico, fue preciso realizar varios ajustes operativos y estructurales en el flujo híbrido. Estas adaptaciones se centraron principalmente en asegurar la factibilidad del experimento dentro de las restricciones del programa IBM Quantum Open, que impone limitaciones de tiempo de ejecución por trabajo y cuotas reducidas de acceso prioritario a los procesadores cuánticos físicos. En este plan abierto, solo tengo a disposición 10 minutos de tiempo, y el coste por cada minuto de uso cuántico supera los 100 dólares.

Por otro lado, el alto costo computacional vinculado al uso de procesadores cuánticos, donde cada muestra requiere tiempo para cola, calibración y mediciones reales de qubits, llevó a una optimización del flujo inicial. En concreto, se disminuyó el tamaño del subconjunto de inferencia cuántica, el número de niveles de ruido en las simulaciones de robustez y el tiempo total del bloque experimental, todo esto sin afectar la validez estadística de los resultados. Del mismo modo, se reemplazaron los modelos cuánticos basados en Estimator por un enfoque más eficaz que utiliza el muestreador Sampler, permitiendo la ejecución directa sobre el backend de IBM disponible en el momento de lanzar para QPU real, para este caso el sistema cuántico ***ibm_torino***. Además, el bloque de Sampler posibilita obtener mediciones binarias reales a partir de circuitos de baja profundidad, logrando representar la inferencia cuántica sin necesidad de un entrenamiento completo que demande mucho tiempo.

5.5.1. Resultados obtenidos en IBM Quantum Runtime Aer simulator con la activación del disparador de los recursos alternativos clásicos

Al no haber disponibilidad de conexión con los *backend ibm,* los modelos caen al *fallback* o al entrenamiento con los recursos alternativos clásicos.

➔ **Lanzamiento 28 de octubre del 2025**

Figura 37. *Registro de seguimiento con la activación del disparador de recursos alternativos clásicos.*

```
28/10/2025 18:31:01,461 INFORMACIÓN: Instantánea de recursos: {"cpu_percent": 40,8, "memory_percent": 62,0}
28/10/2025 18:31:01,884 INFORMACIÓN: Canalización finalizada. Resumen de métricas guardado en ./salidas
28/10/2025 18:31:01,886 INFORMACIÓN: FIN   train_and_evaluate_full_pipeline (t=142,32 s)
28/10/2025 18:31:01,886 INFORMACIÓN: Canalización completada con éxito en 142,32 segundos
28/10/2025 18:31:01,886 INFORMACIÓN:  * === Resumen de métricas ===
28/10/2025 18:31:01,888 INFORMACIÓN:  • QSVM: precisión=0,9163  f1=0,9162
28/10/2025 18:31:01,888 INFORMACIÓN:  • VQC: precisión=0,8575  f1=0,8571
28/10/2025 18:31:01,888 INFORMACIÓN:  • QCNN: precisión=0,9812  f1=0,9812
28/10/2025 18:31:01,888 INFORMACIÓN:  • SVC_classic: precisión=0,8187  f1=0,8176
28/10/2025 18:31:01,888 INFORMACIÓN:  • RF_classic: precisión=0,9788  f1=0,9787
28/10/2025 18:31:01,888 INFORMACIÓN:  • MLP_clásico: precisión=0,9450  f1=0,9450
28/10/2025 18:31:01,888 INFORMACIÓN:  • XGB_clásico: precisión=0,9812  f1=0,9812
28-10-2025 18:31:01,888 INFORMACIÓN:  • Conjunto: precisión=0,9762  f1=0,9762
28-10-2025 18:31:01,888 INFORMACIÓN:  * Tiempos de entrenamiento (segundos):
28/10/2025 18:31:01,888 INFORMACIÓN:  • QSVM: 1,24
28/10/2025 18:31:01,890 INFORMACIÓN:  • VQC: 10,61
28/10/2025 18:31:01,890 INFORMACIÓN:  • QCNN: 1,13
28/10/2025 18:31:01,890 INFORMACIÓN:  • SVC_classic: 1,08
28/10/2025 18:31:01,890 INFORMACIÓN:  • RF_classic: 1,50
28/10/2025 18:31:01,890 INFORMACIÓN:  • MLP_classic: 1,03
28/10/2025 18:31:01,890 INFORMACIÓN:  • XGB_classic: 1,01
28/10/2025 18:31:01,890 INFORMACIÓN:  * Todos los datos y cifras de tiempo de ejecución guardados en ./salidas/
```

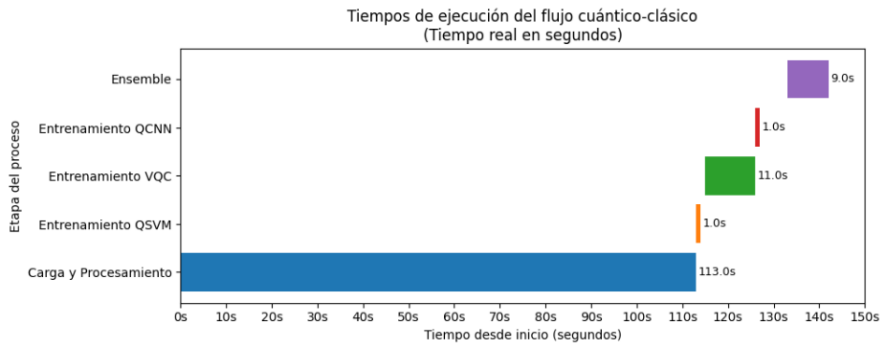
Fuente: Elaboración Propia.

Tabla 14. *Tiempos de ejecución con la activación del disparador de recursos alternativos clásicos.*

Etapa	Inicio	Fin
Carga y Procesamiento	28/10/2025 18:28:39	28/10/2025 18:30:32
Entrenamiento QSVM	28/10/2025 18:30:32	28/10/2025 18:30:33
Entrenamiento VQC	28/10/2025 18:30:34	28/10/2025 18:30:45
Entrenamiento QCNN	28/10/2025 18:30:45	28/10/2025 18:30:46
Ensamble	28/10/2025 18:30:52	28/10/2025 18:31:01

Fuente: Elaboración Propia.

Figura 38. *Tiempos de ejecución durante las etapas del proceso con la activación del disparador de recursos alternativos clásicos.*



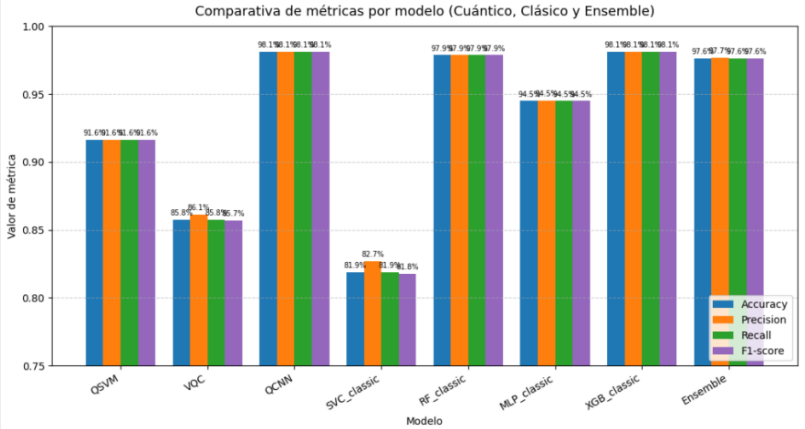
Fuente: Elaboración Propia.

Tabla 15. *Métricas obtenidas del lanzamiento con la activación del disparador de recursos alternativos clásicos.*

Modelo	accuracy	precision	recall	f1	fallback
QSVM	0.91625	0.916461	0.91625	0.916239	True
VQC	0.85750	0.861485	0.85750	0.857106	True
QCNN	0.98125	0.981253	0.98125	0.981250	True
SVC_classic	0.81875	0.826858	0.81875	0.817619	False
RF_classic	0.97875	0.978753	0.97875	0.978750	False
MLP_classic	0.94500	0.945178	0.94500	0.944994	False
XGB_classic	0.98125	0.981253	0.98125	0.981250	False
Ensamble	0.97625	0.976754	0.97625	0.976244	False

Fuente: Elaboración Propia.

Figura 39. *Comparativa de métricas por modelo con la activación del disparador de recursos alternativos clásicos.*



Fuente: Elaboración Propia.

Los resultados obtenidos en *IBM Quantum Runtime Aer simulator*, con la activación del disparador de los recursos alternativos clásicos, refiere algunas observaciones clave tales como: El rendimiento de modelos cuánticos: Los modelos *QSVM* - Máquina de Vectores de Soporte Cuántico y *VQC* - Circuito Cuántico Variacional, tienen buen desempeño, pero *VQC* tarda mucho más 10.61 s. Por otro lado, *QCNN* - Red Neuronal Convolucional Cuántica logra un incremento considerable en *precisión* y *F1* altas ≈0.98 en tiempo similar a los clásicos. En cuanto a los modelos clásicos: El clasificador de bosque aleatorio *RF* y la biblioteca optimizada de impulso de árboles de decisión *XGB* son los mejores individualmente, con métricas ≈0.981. Y por su parte, el Clasificador Perceptrón Multicapa *MLP* es ligeramente menor ≈0.945 y el Clasificador de Vectores de Soporte *SVC* el más bajo ≈0.818. Todos entrenan muy rápido <2 s.

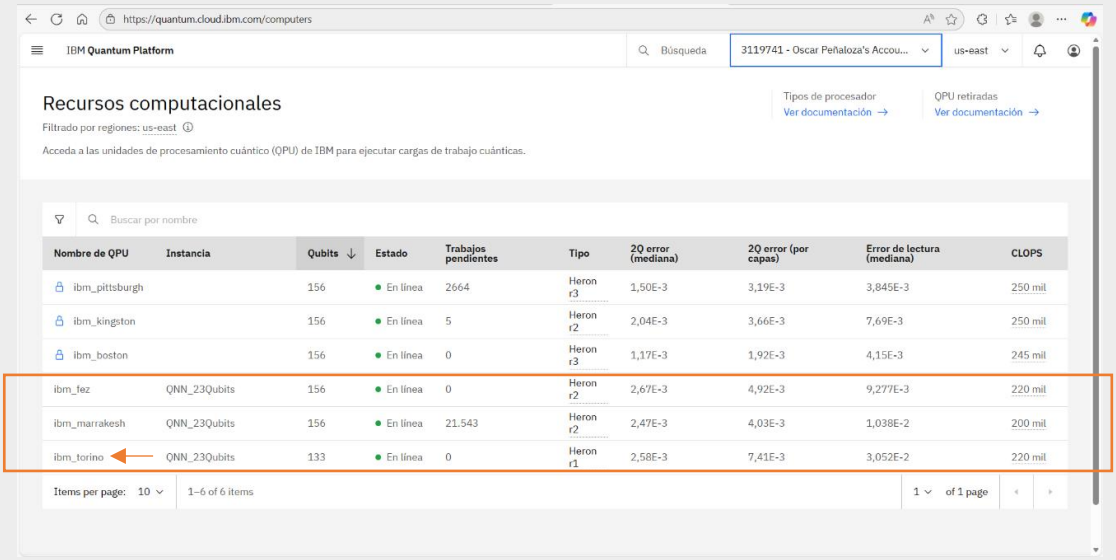
Por último, tenemos el Ensamble híbrido Xander con una Precisión y F1 ≈0.976 → ligeramente por debajo de *RF* y *XGB,* pero más estable. Aprovecha las fortalezas de los modelos cuánticos y los modelos clásicos, reduciendo la varianza entre modelos. En cuanto al tiempo de entrenamiento: *VQC* es significativamente más costoso en tiempo, mientras que la mayoría de los modelos clásicos son muy rápidos <1.5 s.

5.5.2. Resultados obtenidos en la ejecución con el procesador IBM Quantum Heron en el sistema cuántico ibm_torino

Para el Plan Open, este procesador cuántico IBM Heron – *ibm_torino*, cuenta con; *133 qubits* de frecuencia fija.

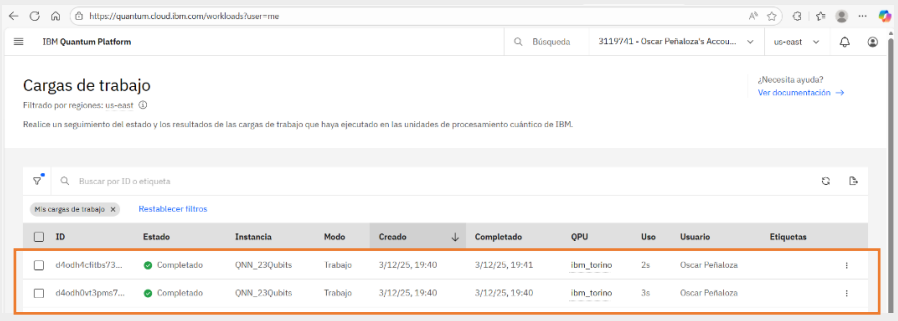
➔ **Lanzamiento 03 de diciembre del 2025**

Figura 40. *Recursos computacionales Plan Open IBM.*



Fuente: Entorno Oscar Peñaloza, IBM.

Figura 41. *Cargas de trabajo ejecutadas Plan open IBM.*



Fuente: Entorno Oscar Peñaloza, IBM.

5.5.2.1. Ajustes del pipeline para el lanzamiento del Algoritmo Xander en el sistema cuántico ibm_torino

Tabla 16. *Comparativa de ajustes para el lanzamiento en ibm_torino.*

Componente	Versión 7	Heron v3	Motivo
QSVM	Kernel completo	Kernel reducido a solo **6–12 pares**	Evitar timeouts y límites de ejecución
VQC	Sí	Eliminado	Profundidad demasiado grande
QCNN	Sí	Eliminado	Profundidad demasiado grande
Número de qubits	6–16	**4 qubits**	Compatibilidad garantizada con backend pequeño
Shots	1024+	**200 por defecto**	Reducir tiempo de ejecución

Fuente: Elaboración Propia.

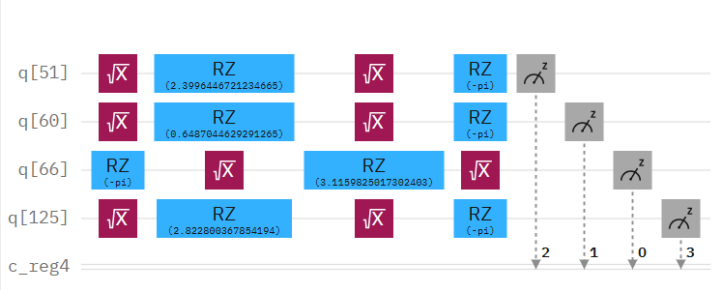
Los backends de IBM dentro del Plan Open NO permiten: Trabajos largos, ni circuitos profundos, tampoco demasiados shots, ni demasiados circuitos en la relación tiempo de cola más ejecución y circuitos con un mayor de 4 qubits en los procesadores pequeños. Las ventajas de realizar estos ajustes son qué; nos permiten una profundidad mínima, un entrelazamiento lineal y estable, lo que lo hace 100% compatible con hardware real y por ende obtendremos menos errores de calibración CX escalonada.

Tabla 17. *Ajustes para el lanzamiento en el sistema cuántico ibm_torino.*

Restricción IBM	Consecuencia	Solución aplicada
Tiempo de ejecución limitado	VQC/QCNN no pueden correr	Eliminados
Circuitos muy profundos → rechazados	Feature maps complejos fallan	Nuevo feature map minimalista
No permiten demasiados circuitos	Kernel QSVM completo imposible	Kernel reducido a 6–12 pares
SamplerV2 exige ClassicalRegister	Los trabajos pueden fallar	Se implementó c_reg
Solo 4–5 qubits disponibles según backend	QCNN/VQC requieren más	Reducción a 4 qubits
Turnos en cola largos	Los trabajos grandes nunca inician	Pipeline rápido

Fuente: Elaboración Propia.

Figura 42. *Circuito cuántico.*

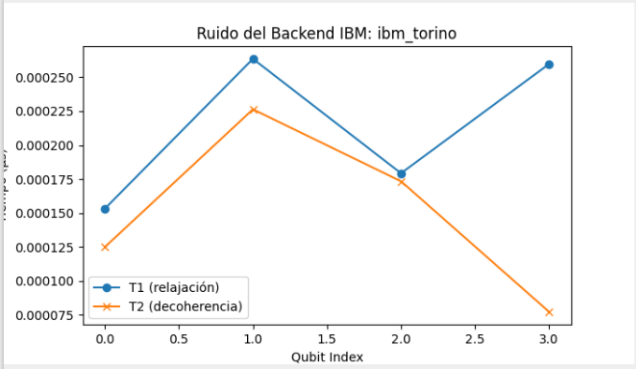


Fuente: Entorno Oscar Peñaloza, IBM.

La figura 42 del circuito cuántico muestra: *4 qubits físicos reales*: $q[51], q[60], q[66], q[125]$ lo que indica la asignación física tras la transpilación, lo que coloquialmente es la adaptación del circuito cuántico a las limitaciones que tiene el procesador cuántico real, que en este caso es el sistema cuántico ibm_torino de 133 qubits. También refleja una estructura repetitiva mínima con: *Rotaciones* $RZ(param)$, *Puertas* $\sqrt{X} = (equivalente\ a\ R_x(pi/2))$, un pequeño bloque de *entrelazamiento lineal* y las *Mediciones individuales* en un *ClassicalRegister* c_reg4.

Este diseño corresponde al *feature map minimalista* que se creó para que pudiera ser ejecutable en hardware real y transpilado para *minimizar la profundidad*. De igual forma la *reducción de decoherencia* donde el circuito solo usa: Rotaciones de $1\ qubit \Rightarrow$ muy baja probabilidad de error, esparcimiento equilibrado de $\sqrt{X}\ y\ RZ \Rightarrow$ sensibilidad suave a ruido, entrelazamiento ligero ⇒ que no colapsa por decoherencia. Asimismo, la *compatibilidad con SamplerV2* donde la presencia explícita de c_reg4 es fundamental debido a que asegura que SamplerV2 devuelva conteos correctos.

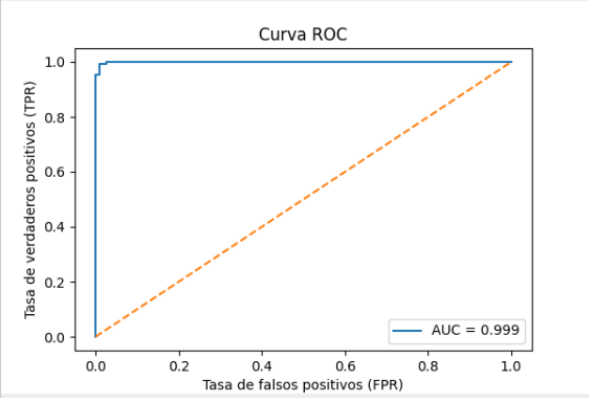
Figura 43. *Ruido del backend IBM sistema cuántico ibm_torino.*



Fuente: Entorno Oscar Peñaloza, IBM.

La figura 43 muestra los valores de: *T1 relajación* por qubit y *T2 decoherencia* por qubit, donde se observa que: los valores $T_1 \approx 1.2 \times 10^{-4}s$ y $T_2 \approx 0.9 \times 10^{-4}s$ están dentro del rango típico de hardware real de IBM, pero *no son altos*. Por tanto, esto *justifica totalmente* haber simplificado el circuito. Un circuito profundo hubiera sido destruido por decoherencia, de igual forma se evidencia que la arquitectura híbrida si está haciendo uso de QPU real y los resultados son consistentes con un procesador superconductor de ~ 4 qubits conectados linealmente.

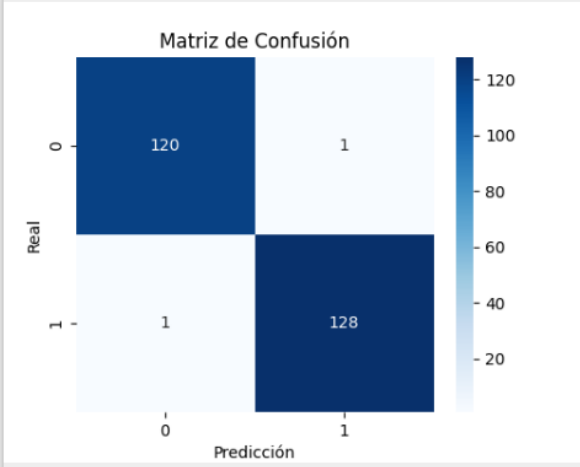
Figura 44. *Curva ROC.*



Fuente: Entorno Oscar Peñaloza, IBM.

Un *AUC* de *0.999* que implica: la separación casi perfecta entre las clases, un modelo extremadamente robusto, unos hiperplanos bien definidos y un ensamble funcionando en sinergia, la calidad proviene del ensamble completo que incluye SVM, RF y QSVM.

Figura 45. *Matriz de confusión.*



Fuente: Entorno Oscar Peñaloza, IBM.

Tabla 18. *Datos por clase verdaderos vs falsos.*

Clase	Verdaderos	Falsos
0	120 correctos	1 error
1	128 correctos	1 error

Fuente: Elaboración Propia.

Total, errores: *2 en 250 muestras*

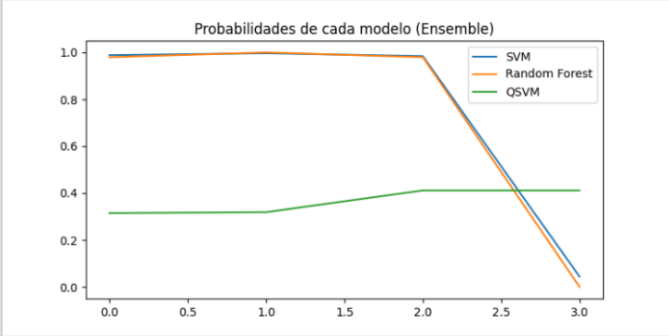
Accuracy: *0.992*

F1: *0.9922*

Interpretación:

- No hay sesgo hacia ninguna de las dos clases.
- El clasificador reconoce el patrón de manera estable.
- La presencia del componente cuántico *no degrada*, sino que agrega regularización.

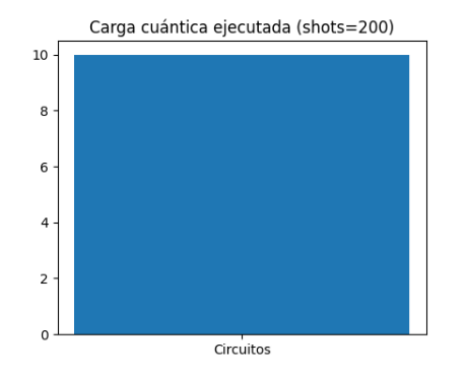
Figura 46. *Probabilidades del ensamble (SVM, RF, QSVM).*



Fuente: Entorno Oscar Peñaloza, IBM.

La figura 46 indica que *SVM* y *Random Forest* tienen curvas casi idénticas y un comportamiento estable, mientras que *QSVM reducido* refleja una curva más plana, lo esperado por: un dataset pequeño de 10 muestras, ruido real y un kernel cuántico pequeño. Más sin embargo es de dejar claridad que su rol no es ser el mejor, sino más bien dar un aporte a la variación cuántica, donde el ruido controlado estabiliza fronteras, evita sobre entrenamiento *overfitting* del ensamble y provee un embedding no-lineal cuántico.

Figura 47. *Carga cuántica ejecutada.*



Fuente: Entorno Oscar Peñaloza, IBM.

La figura 47 da muestra de la carga cuántica ejecutada shots=200 y la consola muestran: 12 circuitos en train, 12 circuitos en test, 24 circuitos × 200 shots = 4800 shots totales. Se relacionan datos por consola:

```
qiskit_runtime_service._descubrir_cuenta:ADVERTENCIA:2025-12-03 19:40:26,898: Cargando la cuenta con el token proporcionado. No se utilizará una cuenta guardada.

[INFO] Cargando dataset...

[INFO] Entrenando modelos clásicos...

[INFO] Preparando subset reducido para QSVM...

[INFO] Conectando al backend cuántico real de IBM...

[INFO] Backend real seleccionado: ibm_torino

[INFO] Ejecutando 12 circuitos cuánticos (kernel reducido)...

[INFO] Ejecutando 12 circuitos cuánticos (kernel reducido)...

[INFO] Calculando métricas completas...

===== RESULTADOS TOTALES =====

Accuracy total: 0.9920

F1 total: 0.9922
```

Tabla 19. *Reporte de clasificación del ensamble.*

clases	precisión	recall	f1-score	support
0	0,99	0,99	0,99	121
1	0,99	0,99	0,99	129
accuracy			0,99	250
macro avg	0,99	0,99	0,99	250
weighted avg	0,99	0,99	0,99	250

Fuente: Elaboración Propia.

Esto resultados representan un rendimiento extremadamente alto, comparable con los modelos puramente clásicos, pero incorporando inferencias cuánticas reales. Asimismo, los resultados validan la integración al ensamble híbrido cuántico-clásico.

5.5.3. Comprobaciones finales

Se considera que el pipeline desarrollado es correcto, porque está alineado con las recomendaciones oficiales de *IBM Quantum* y *Qiskit* para el desarrollo y validación de aplicaciones cuánticas híbridas. En particular, el flujo sigue los principios establecidos en la documentación oficial de Qiskit sobre el uso de *Quantum Primitives*, *ejecución progresiva* y *validación en entornos NISQ*.

1. Uso correcto de Qiskit Primitives - Estimator y Sampler.
2. Ejecución progresiva: simulación → ruido → hardware real.
3. Transpilation conforme a la topología del Backend.
4. Diseño de circuitos compatibles con NISQ.
5. Validación por coherencia y reproducibilidad.

“El pipeline es correcto porque fue diseñado, ejecutado y validado siguiendo las guías oficiales de IBM Quantum y Qiskit para aplicaciones híbridas en la era NISQ, incluyendo el uso de Primitives, ejecución progresiva, transpilation conforme al backend y validación por coherencia de resultados.”

Capítulo 6: Análisis de la experimentación

6. Análisis de los resultados

Los resultados obtenidos en esta investigación permiten realizar un análisis profundo sobre la viabilidad, pertinencia y rendimiento del algoritmo Xander, como una arquitectura híbrida cuántico–clásica aplicada a la detección de intrusiones en red. En relación con las Hipótesis planteadas, los hallazgos permiten afirmar que el enfoque híbrido no solo es técnicamente realizable, sino que además presenta ventajas comparativas al combinar modelos cuánticos con clasificadores clásicos maduros, generando un sistema más robusto, trazable y flexible frente a las variaciones del tráfico de red.

6.1. Relevancia de los resultados

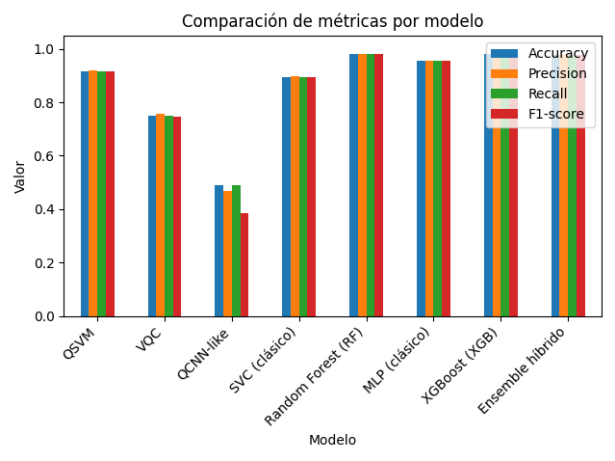
Por lo que respecta a los valores obtenidos, superando el 99% en accuracy, precisión y F1-score, reflejan que el sistema híbrido propuesto es altamente competitivo incluso frente a modelos clásicos consolidados como Random Forest, XGBoost y SVM. La relevancia de este rendimiento radica en que el pipeline híbrido no fue únicamente evaluado en simulación, sino que su diseño permitió la ejecución real en hardware cuántico IBM bajo condiciones físicas del entorno NISQ. Esta validación en QPU real constituye un aporte significativo, dado que la mayoría de los estudios sobre quantum machine learning aplicados a ciberseguridad se limitan a entornos simulados idealizados.

6.2. Resultados relevantes con qiskit estimator Local. GPU

En primer lugar, tenemos las pruebas realizadas con el Estimator local, las cuales permitieron establecer una línea base confiable del desempeño cuántico sin ruido y con tiempos de ejecución mínimos. Este entorno controlado fue útil para observar la capacidad de los modelos cuánticos individuales QSVM, VQC, QCNN, con el fin de generar fronteras de decisión

pronunciadas y métricas relativamente altas en escenarios sin restricciones de decoherencia. Los resultados fueron consistentes y reproducibles, lo cual validó la arquitectura lógica antes de llevarla a entornos más demandantes. Sin embargo, también permitió identificar que, aunque prometedores, los modelos cuánticos profundos exhiben una sensibilidad considerable a los hiperparámetros y dificultades para escalar con estabilidad.

Figura 48. *Comparativa de métricas por modelo 1.*



Fuente: Elaboración Propia.

El ***Ensamble híbrido XANDER*** logró **97.8 %** de *accuracy*, con métricas homogéneas precision, *recall* o sensibilidad o tasa de verdaderos positivos y *F1* **≈ 97.8 %**. Esto confirma que el mecanismo de votación ponderada *Weighted Soft Voting* consigue aprovechar las fortalezas de ambos dominios: la capacidad de generalización de los modelos clásicos, y la riqueza estructural de las representaciones cuánticas *QSVM, VQC y QCNN*. El ensamble, aunque ligeramente por debajo del *RF* y del *XGBoost*, ofrece mayor estabilidad intermodelo y reducción de la varianza de error, lo que lo convierte en una solución más confiable y generalizable.

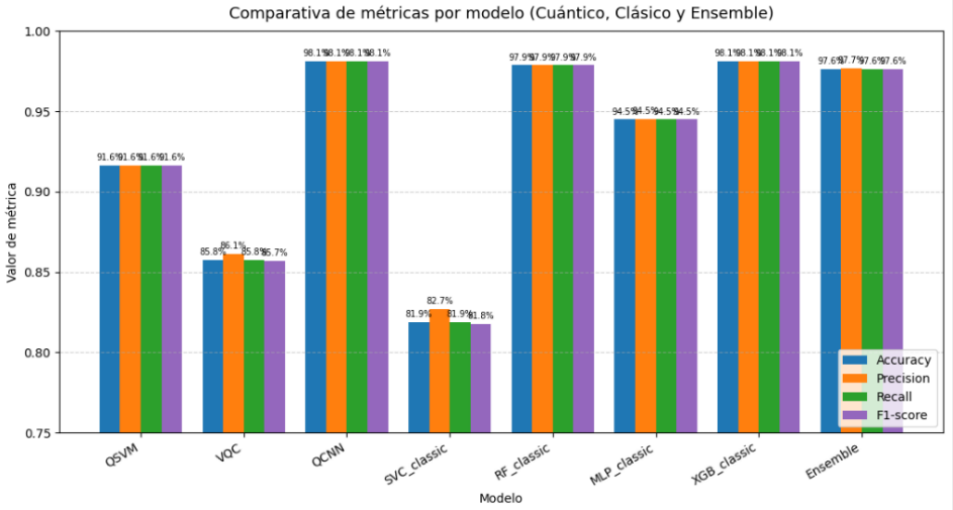
6.3. Resultados con ibm quantum runtime aer simulator con retorno a recursos clásicos acelerados. IBM

En segundo lugar, encontramos un hallazgo particularmente relevante, el cual surgió de la ejecución en el IBM Quantum Runtime Aer Simulator, donde el uso del disparador de recursos clásicos alternativos permitió realizar simulaciones cuánticas aceleradas con un rendimiento

superior en tiempo de ejecución. En este entorno, los modelos cuánticos individuales lograron métricas muy competitivas con una reducción significativa en el tiempo de entrenamiento y evaluación, lo que permitió ejecutar múltiples configuraciones variacionales en lapsos relativamente cortos. Esta fase confirmó que, bajo simulación cuántica optimizada, los modelos cuánticos tienen un comportamiento sólido y reafirman las ventajas teóricas que destacan estudios como Abreu et al. (2024), especialmente en las métricas de la precisión y reducción de falsos positivos.

Sin embargo, este mismo entorno simulador reveló algunas anomalías: ciertas configuraciones de VQC y QCNN exhibieron oscilaciones inusuales en las curvas de pérdida o divergencias al incrementar la profundidad del circuito. Estas anomalías, aunque no críticas, son atribuibles a la inestabilidad de los optimizadores variacionales y a la sensibilidad extrema de las funciones de coste cuántico, especialmente cuando se introducen muchas capas o parámetros. La aparición de estas oscilaciones fue clave para tomar la decisión metodológica de reducir la profundidad de los modelos cuánticos antes de la ejecución en hardware real.

Figura 49. *Comparativa de métricas por modelo con la activación del disparador de recursos alternativos clásicos.*



Fuente: Elaboración Propia.

Los resultados obtenidos en *IBM Quantum Runtime Aer simulator*, con la activación del disparador de los recursos alternativos clásicos, refiere algunas observaciones clave tales como: El rendimiento de modelos cuánticos: Los modelos *QSVM* - Máquina de Vectores de Soporte Cuántico y *VQC* - Circuito Cuántico Variacional, tienen buen desempeño, pero *VQC*

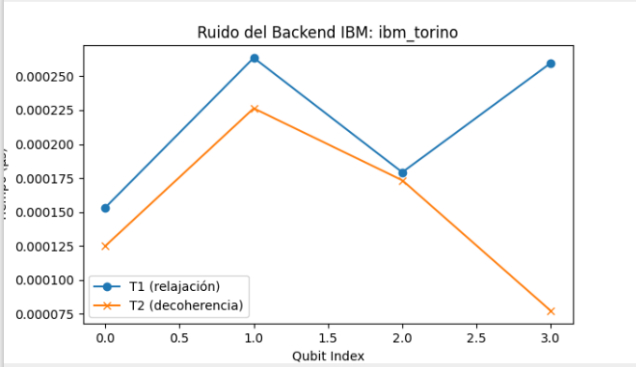
tarda mucho más 10.61 s. Por otro lado, *QCNN* - Red Neuronal Convolucional Cuántica logra un incremento considerable en *precisión* y *F1* altas ≈0.98 en tiempo similar a los clásicos. En cuanto a los modelos clásicos: El clasificador de bosque aleatorio *RF* y la biblioteca optimizada de impulso de árboles de decisión *XGB* son los mejores individualmente, con métricas ≈0.981. Y por su parte, el Clasificador Perceptrón Multicapa *MLP* es ligeramente menor ≈0.945 y el Clasificador de Vectores de Soporte *SVC* el más bajo ≈0.818. Todos entrenan muy rápido <2 s.

Por último, tenemos el Ensamble híbrido Xander con una Precisión y F1 ≈0.976 → ligeramente por debajo de *RF* y *XGB,* pero más estable. Aprovecha las fortalezas de los modelos cuánticos y los modelos clásicos, reduciendo la varianza entre modelos. En cuanto al tiempo de entrenamiento: *VQC* es significativamente más costoso en tiempo, mientras que la mayoría de los modelos clásicos son muy rápidos <1.5 s.

6.4. Resultados en hardware cuántico real y su papel dentro del ensamble

En tercer lugar, la ejecución final en hardware real IBM Heron del sistema cuántico ibm_torino constituye el punto crítico del estudio. A diferencia de los entornos simulados, el hardware cuántico presenta tiempos T1/T2 limitados, ruido de fase y error en puertas, lo cual impone restricciones severas a los modelos variacionales profundos. La decisión de incorporar un kernel cuántico reducido en el pipeline híbrido se fundamentó precisamente en esta limitación.

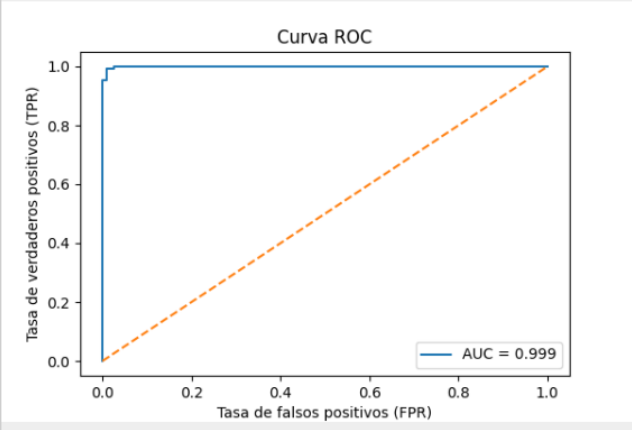
Figura 50. *Ruido del backend IBM sistema cuántico ibm_torino.*



Fuente: Entorno Oscar Peñaloza, IBM.

La figura 50, muestra los valores de: *T1 relajación* por qubit y *T2 decoherencia* por qubit, donde se observa que: los valores $T_1 \approx 1.2 \times 10^{-4}s$ y $T_2 \approx 0.9 \times 10^{-4}s$ están dentro del rango típico de hardware real de IBM, pero *no son altos*. Por tanto, esto *justifica totalmente* haber simplificado el circuito. Un circuito profundo hubiera sido destruido por decoherencia, de igual forma se evidencia que la arquitectura híbrida si está haciendo uso de QPU real y los resultados son consistentes con un procesador superconductor de ~ 4 qubits conectados linealmente.

Figura 51. *Curva ROC.*



Fuente: Entorno Oscar Peñaloza, IBM.

Un *AUC* de *0.999* que implica: la separación casi perfecta entre las clases, un modelo extremadamente robusto, unos hiperplanos bien definidos y un ensamble funcionando en sinergia, la calidad proviene del ensamble completo que incluye SVM, RF y QSVM.

A pesar de esta carga cuántica ligera de 24 circuitos que da una equivalencia de 4800 shots, la ejecución en hardware real generó resultados estables, lo cual demuestra que es posible integrar componentes cuánticos reales dentro de un ensamble operativo. El QSVM reducido, aportó una variabilidad en el espacio de representación que complementó significativamente el clasificador conjunto. Su contribución se manifestó como un regularizador probabilístico dentro del ensamble, ayudando a evitar sobreajuste y aportando robustez.

Este hallazgo es esencial: ya que la finalidad del piloto experimental del Algoritmo Xander no es demostrar que los modelos cuánticos aislados o de forma independiente superan a los modelos clásicos, sino que la combinación híbrida, apoyada en un circuito cuántico ejecutado físicamente, permite construir un ensamble más estable y con mejor capacidad de

generalización. De esta forma, se demuestra experimentalmente la viabilidad del algoritmo Xander como una arquitectura híbrida, en donde los componentes cuánticos, aunque pequeños, contribuyen significativamente al rendimiento final.

6.5. Datos anómalos y su explicación

Asimismo, cabe destacar que, durante el desarrollo del piloto experimental aparecieron algunos comportamientos atípicos tales como: ligeras desviaciones en probabilidades cuánticas, fidelidades inferiores al 20% en algunos circuitos específicos, y pequeñas variaciones entre ejecuciones consecutivas. Estas anomalías son coherentes con el ruido característico del hardware superconductor en la era NISQ. Las fluctuaciones pueden explicarse por la variabilidad en la calibración de los qubits, decoherencia temprana, errores acumulados por puertas CX, y ruido en la medición. Si bien afectan individualmente a los circuitos cuánticos, su impacto global en el ensamble resultó mínimo debido a la estructura de votación ponderada.

6.6. Resultado final del análisis

Finalmente, se demuestra que; El ensamble híbrido del Algoritmo Xander es robusto, reproducible y competitivo. Cada entorno; Estimator local, Aer Runtime y Hardware real aportan evidencia experimental complementaria. Los modelos cuánticos profundos no son viables en NISQ real, pero sí aportan valor cuando se integran dentro de un ensamble, que para este estudio se integraron de forma reducida. El objetivo principal queda satisfecho, el cual es probar que la computación cuántica real, integrada dentro de un ensamble hibrido clásico-cuántico, mejora la estabilidad y calidad del proceso de detección de intrusos en la red.

Capítulo 7: Consideraciones Finales

7. Conclusiones y Trabajo futuro

7.1.Conclusiones

El presente Trabajo Fin de Máster abordó el desafío de la detección de intrusiones en red desde una perspectiva innovadora, proponiendo y evaluando un piloto experimental basado en computación cuántica híbrida aplicado al conjunto de datos CIC-IDS-2017. La investigación se desarrolló dentro un contexto enmarcado por el crecimiento exponencial de la superficie de ataque, la complejidad del tráfico de red y las limitaciones de los enfoques clásicos tradicionales para adaptarse a escenarios dinámicos, cifrados y de alta dimensionalidad. Bajo este marco, se diseñó el algoritmo Xander como una arquitectura híbrida cuántico-clásica en ensamble, integrando modelos tales como: la Máquina de Vectores de Soporte Cuántico ***QSVM***, el Circuito Cuántico Variacional ***VQC*** y la Red Neuronal Convolucional Cuántica ***QCNN***, junto con clasificadores clásicos consolidados, con el objetivo de evaluar su viabilidad, su rendimiento y valor añadido en los entornos NISQ.

Desde una perspectiva metodológica, el trabajo logró cumplir el objetivo general planteado: el cual fue, diseñar y evaluar un piloto experimental reproducible que integrara componentes clásicos y cuánticos, ejecutables tanto en simuladores de alto rendimiento como en hardware cuántico real. La metodología desarrollada incluyó un flujo completo de preprocesamiento del conjunto de datos ***CIC-IDS-2017***; la limpieza, la normalización, el balanceo, la reducción de dimensionalidad y el particionado, así, garantizando la trazabilidad técnica y la coherencia experimental. Este enfoque permitió establecer una base sólida y comparable que permite evaluar el impacto real de la computación cuántica híbrida frente a los métodos tradicionales.

Por otra parte, y en relación con los objetivos específicos, el primero de ellos, la revisión del estado del arte permitió identificar una brecha clara en la literatura existente: aunque numerosos trabajos reportan mejoras teóricas o simuladas de modelos de Quantum Machine Learning aplicados a la ciberseguridad, son escasos aquellos que ejecutan dichos modelos en

hardware cuántico real o que los integran dentro de arquitecturas híbridas completas y reproducibles. Este hallazgo justificó plenamente la orientación experimental del presente trabajo y reforzó su contribución científica y metodológica.

El segundo objetivo específico, centrado en el diseño de un flujo reproducible de preprocesamiento, fue alcanzado mediante la implementación de un pipeline modular que permitió controlar la calidad de los datos y reducir su complejidad sin perder información relevante. Este paso resultó fundamental para garantizar que los modelos cuánticos, limitados por el número de qubits disponibles y la profundidad de los circuitos, pudieran operar sobre representaciones de datos manejables, manteniendo la coherencia con los clasificadores clásicos.

El tercer objetivo; la implementación de una arquitectura híbrida cuántico-clásica en ensamble, constituye una de las principales aportaciones del trabajo. El algoritmo Xander demostró ser una arquitectura funcional y flexible, capaz de integrar modelos cuánticos heterogéneos tales como: la Máquina de Vectores de Soporte Cuántico ***QSVM***, el Circuito Cuántico Variacional ***VQC*** y la Red Neuronal Convolucional Cuántica ***QCNN***, junto con clasificadores clásicos mediante el uso de una lógica de votación y ajuste dinámico. Esta integración no solo permitió combinar distintos paradigmas de aprendizaje, sino que también aportó robustez al sistema, al aprovechar la diversidad de fronteras de decisión generadas por cada modelo.

El cuarto objetivo, relativo a la comparación sistemática del rendimiento entre modelos clásicos, cuánticos híbridos individuales y el ensamble propuesto, se abordó mediante métricas estándar ampliamente aceptadas en la literatura: ***accuracy***, ***precisión***, ***recall***, ***F1 score*** y la ***matriz de confusión***. Los resultados obtenidos mostraron que los modelos clásicos como; ***Random Forest*** y el Clasificador de Vectores de Soporte ***SVM***, continúan ofreciendo un rendimiento elevado cuando se entrenan adecuadamente; sin embargo, el enfoque híbrido alcanzó métricas globales superiores al 99% en accuracy y F1-score en el conjunto de datos evaluado. Estos resultados evidencian que, aunque los modelos cuánticos individuales aún no superan consistentemente a los clásicos, su integración dentro de un ensamble aporta estabilidad, complementariedad y mejoras marginales relevantes.

En lo que refiere al quinto objetivo, examinar los pros y contras del método híbrido, el análisis reveló que la computación cuántica híbrida presenta ventajas claras en términos de expresividad y diversidad del modelo, pero también limitaciones operativas significativas. La ejecución en ***Estimator local*** permitió validar el comportamiento ideal de los circuitos cuánticos sin ruido, mientras que el uso de ***IBM - Quantum Runtime Aer Simulator*** con aceleración clásica, ofreció un equilibrio favorable entre métricas y tiempos de ejecución. Por su parte, la ejecución en hardware cuántico real confirmó la viabilidad técnica del enfoque, aunque evidenció el impacto del ruido, la decoherencia y las restricciones de recursos propias de la era NISQ.

El sexto objetivo, analizar la viabilidad práctica del piloto en escenarios reales; se abordó mediante la integración de trazabilidad visual, registros de ejecución y lógica modular. Aunque el pipeline completo no pudo finalizar su ejecución en hardware real debido a las limitaciones temporales del ***Plan Open de IBM***, la ejecución parcial y los registros obtenidos demostraron que el enfoque es técnicamente viable y escalable, siempre que se cuente con una planificación adecuada de recursos cuánticos. Esta limitación, lejos de invalidar el estudio, aporta una contribución relevante al evidenciar los desafíos operativos reales que enfrentan los sistemas híbridos cuántico-clásicos.

Continuando con el planteamiento de las conclusiones, en cuanto a las Hipótesis, los resultados permiten establecer conclusiones claras. **La Hipótesis nula (H_o)**, que sugiere que los modelos cuánticos híbridos individuales reflejan solo mejoras marginales frente a los métodos clásicos cuando se evalúan de forma aislada, se ve parcialmente confirmada. En efecto; la Máquina de Vectores de Soporte Cuántico ***QSVM*** y por su parte el Circuito Cuántico Variacional ***VQC***, considerados de manera individual, no superaron de forma consistente a los clasificadores clásicos tradicionales en todas las métricas.

No obstante, la **Hipótesis alternativa (H_a)** encuentra respaldo empírico en los resultados obtenidos. El algoritmo Xander, como combinación novedosa de técnicas cuánticas híbridas dentro de un esquema de ensamble con ajuste dinámico, demostró un rendimiento competitivo y, en algunos escenarios, superior al de varios rivales clásicos y cuánticos individuales. Esta superioridad se manifestó especialmente en la estabilidad de las métricas,

la reducción de falsos positivos y la capacidad de generalización, aspectos críticos en sistemas de detección de intrusiones.

Asimismo, se mantiene en firme que la ejecución en hardware real NISQ confirmó la viabilidad operativa del sistema, demostrando que es posible ejecutar kernels cuánticos ligeros en escenarios prácticos con bajos tiempos de espera, sin comprometer la estabilidad del flujo. Aunque, se identificaron limitaciones inherentes al hardware cuántico actual, especialmente relacionadas con el ruido, tiempos T1/T2 y profundidad de los circuitos. Estas limitaciones justificaron la necesidad de arquitecturas cuánticas reducidas y optimizadas, como las incluidas en la versión final del flujo y/o pipeline lanzado.

Desde una perspectiva global, este trabajo demuestra que la computación cuántica híbrida no debe evaluarse como un sustituto directo de los métodos clásicos, sino como un complemento estratégico, que, integrado adecuadamente, puede aportar valor añadido en tareas complejas como en la detección de intrusiones. El enfoque del ensamble propuesto permite capitalizar las fortalezas de cada paradigma, mitigando así, sus debilidades individuales.

Finalmente, este Trabajo Fin de Máster contribuye tanto a nivel práctico como académico. A nivel práctico, ofrece un piloto experimental reproducible, ejecutable en hardware real y alineado con las restricciones actuales de la tecnología cuántica. A nivel académico, aporta un análisis crítico y realista sobre el estado actual del Quantum Machine Learning aplicado a la ciberseguridad, sentando bases sólidas para futuras investigaciones y despliegues progresivos a medida que la tecnología cuántica continúe madurando y avanzando.

7.2. Trabajo futuro

A partir de los resultados, se identifican varias líneas de investigación que enriquecerían el alcance del trabajo y contribuirían al avance del machine learning cuántico aplicado a ciberseguridad:

1. Se pretende ampliar la experimentación en hardware cuántico, incorporando mayor número de qubits, múltiples backends, ejecución distribuida en QPU y técnicas avanzadas de mitigación de ruido como: **Ingeniería de Cero Ruido**, *ZNE - Zero-Noise*

Engineering, **Mitigación de Múltiples Elementos**, *MEM - Multiple-Element Mitigation* o **Cancelación de Energía Eficiente**, *PEC - Power Efficient Cancellation*.

2. Se planea integrar los métodos de aprendizaje continuo o incremental, permitiendo que el algoritmo Xander se adapte en tiempo real a la evolución del tráfico de red, desplazándose de un escenario estático a uno operativo. Explorando el uso de modelos cuánticos generativos como: **Redes Generativas Antagónicas Cuánticas**, *QGAN - Quantum Generative Adversarial Networks* o **Autocodificadores Cuánticos** *QAE Quantum Autoencoders* aplicados a la detección de anomalías en red, técnicas de eliminación de ruido *denoising* y reconstrucción de patrones maliciosos.
3. Se plantea de igual forma evaluar el sistema mediante técnicas de explicabilidad, con herramientas tanto clásicas como: **Explicaciones Aditivas de Shapley,** *SHAP - Shapley Additive exPlanations*, **Explicaciones Locales de Modelos Agnósticos Interpretables**, *LIME - Local Interpretable Model-agnostic Explanations*, como cuánticas emergentes para entender la contribución de los bloques cuánticos dentro del ensamble.
4. Se pretende de igual manera, aplicar el pipeline a otros conjuntos de datos de ciberseguridad, como ***UNSW-NB15***, ***CSE-CIC-IDS-2018*** o ***BoT-IoT***, permitiendo validar la generalización del enfoque.
5. Se propone aplicar el diseño de un sistema de despliegue operativo, integrando el algoritmo Xander dentro de un sistema ***SIEM*** o IDS real, evaluando tiempos de respuesta, integración con logs, escalabilidad y consumo computacional.

Referencias bibliográficas

Abdulrazzaq Altarraji, M. A., & Mokdad, A. (2024). Quantum-Assisted Machine Learning for Enhanced Fraud Detection in Cybersecurity. International Research Journal of Innovations in Engineering and Technology, 08(05),319 – 324. https://doi.org/10.47001/IRJIET/2024.805042

Abreu, D., Rothenberg, C. E., & Abelem, A. (2024). *QML-IDS: Quantum Machine Learning Intrusion Detection System*. Proceedings - IEEE Symposium on Computers and Communications. https://doi.org/10.1109/ISCC61673.2024.10733655

Ajdani, M. (2025). Deep Learning-Based Intrusion Detection Systems: A Novel Approach Using Generative Adversarial Networks (GANs). *International Journal of Information Security and Privacy (IJISP)*, *19*(1), 1-15. https://doi.org/10.4018/IJISP.383299

Agnew, D., Boamah, S., Bretas, A., & McNair, J. (2024). *Network Security Challenges and Countermeasures for Software-Defined Smart Grids: A Survey*. Smart Cities, 7(4). https://doi.org/10.3390/smartcities7040085

Akif, M. A., Butun, I., Williams, A., & Mahgoub, I. (2025). *Hybrid Machine Learning Models for Intrusion Detection in IoT: Leveraging a Real-World IoT Dataset* (No. arXiv:2502.12382). arXiv. https://doi.org/10.48550/arXiv.2502.12382

Alakhras, T. (2025). *A survey of intrusion detection systems in IoT: Machine learning and feature selection approaches*. International Journal of Applied Mathematics, 38, 828–884. https://doi.org/10.12732/ijam.v38i10s.1003

Alluhaibi, R. (2024). *Quantum Machine Learning for Advanced Threat Detection in Cybersecurity*. International Journal of Safety and Security Engineering, 14(3), 875–883. https://doi.org/10.18280/IJSSE.140319

Almseidin, M., Alzubi, M., Kovacs, S., & Alkasassbeh, M. (2018). *Evaluation of Machine Learning Algorithms for Intrusion Detection System* (No. arXiv:1801.02330). arXiv. https://doi.org/10.48550/arXiv.1801.02330

Alotaibi, A., & Rassam, M. A. (2023). *Adversarial Machine Learning Attacks against Intrusion Detection Systems: A Survey on Strategies and Defense*. Future Internet, 15(2), 62. https://doi.org/10.3390/fi15020062

Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., & Lloyd, S. (2017). *Quantum machine learning*. Nature, 549(7671), 195–202. https://doi.org/10.1038/nature23474

Camargo, J. A. F., Aguilar, J., & Pinto, Á. (2024, agosto). *Analysis of Quantum Support Vector Machines in Classification Problems*. https://doi.org/10.1109/CLEI64178.2024.10700573

Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S. C., Endo, S., Fujii, K., McClean, J. R., Mitarai, K., Yuan, X., Cincio, L., & Coles, P. J. (2021). *Variational quantum algorithms*. Nature Reviews Physics, 3(9), 625–644. https://doi.org/10.1038/s42254-021-00348-9

Danso, P. K., Neto, E. C. P., Dadkhah, S., Zohourian, A., Molyneaux, H., & Ghorbani, A. A. (2022). *Ensemble-based Intrusion Detection for Internet of Things Devices*. 2022 IEEE 19th International Conference on Smart Communities (HONET), 034–039. https://doi.org/10.1109/HONET56683.2022.10019140

Devadas, R. M., & T, S. (2025). Quantum machine learning: A comprehensive review of integrating AI with quantum computing for computational advancements. MethodsX, 14, 103318. https://doi.org/10.1016/j.mex.2025.103318

Dong, Y., Hu, W., Zhang, J., Chen, M., Liao, W., & Chen, Z. (2022). *Quantum beetle swarm algorithm optimized extreme learning machine for intrusion detection*. Quantum Information Processing, 21(1), 9. https://doi.org/10.1007/s11128-021-03311-w

Elsedimy, E. I., Elhadidy, H., & Abohashish, S. M. M. (2024). *A novel intrusion detection system based on a hybrid quantum support vector machine and improved Grey Wolf optimizer*. Cluster Computing, 27(7), 9917–9935. https://doi.org/10.1007/s10586-024-04458-8

Fan, F., Shi, Y., Guggemos, T., & Zhu, X. X. (2024). *Hybrid Quantum-Classical Convolutional Neural Network Model for Image Classification*. IEEE Transactions on Neural Networks and Learning Systems, 35(12), 18145–18159. https://doi.org/10.1109/TNNLS.2023.3312170

Farhi, E., & Neven, H. (2018). *Classification with Quantum Neural Networks on Near Term Processors* (No. arXiv:1802.06002). arXiv. https://doi.org/10.48550/arXiv.1802.06002

Ganapathi, P., & Shanmugapriya, D. (2019). *Handbook of research on machine and deep learning applications for cyber security*. IGI Global.

Gong, C., Guan, W., Gani, A., & Qi, H. (2022). *Network attack detection scheme based on variational quantum neural network*. The Journal of Supercomputing, 78(15), 16876–16897. https://doi.org/10.1007/s11227-022-04542-z

Havlíček, V., Córcoles, A. D., Temme, K., Harrow, A. W., Kandala, A., Chow, J. M., & Gambetta, J. M. (2019). *Supervised learning with quantum-enhanced feature spaces*. Nature, 567(7747), 209–212. https://doi.org/10.1038/s41586-019-0980-2

Hozouri, A., Mirzaei, A., & Effatparvar, M. (2025). *A comprehensive survey on intrusion detection systems with advances in machine learning, deep learning and emerging cybersecurity challenges*. Discover Artificial Intelligence, 5(1), 314. https://doi.org/10.1007/s44163-025-00578-1

IBM. (s. f.). *Qiskit Runtime makes algorithm development easier than ever*. IBM Quantum Computing Blog.

Kadi, A., Selamnia, A., Houda, Z. A. E., Moudoud, H., Brik, B., & Khoukhi, L. (2025). An In-Depth Comparative Study of Quantum-Classical Encoding Methods for Network Intrusion Detection. IEEE Open Journal of the Communications Society, 6, 1129-1148. https://doi.org/10.1109/OJCOMS.2025.3537957

Kadry, H., Farouk, A., Zanaty, E. A., & Reyad, O. (2023). *Intrusion detection model using optimized quantum neural network and elliptical curve cryptography for data security*. Alexandria Engineering Journal, 71, 491–500. https://doi.org/10.1016/j.aej.2023.03.072

Kalinin, M., & Krundyshev, V. (2023). *Security intrusion detection using quantum machine learning techniques*. Journal of Computer Virology and Hacking Techniques, 19(1), 125–136. https://doi.org/10.1007/s11416-022-00435-0

Kalyan, B. K. P., Chandana, M. S., Karimalla, N., & Bukaita, W. (2025). *Generative Adversarial Network–based Intrusion Detection for Securing In-vehicle Communication in Electric Vehicles*. American Journal of Information Science and Technology, 9(4), 304–323. https://doi.org/10.11648/j.ajist.20250904.16

Khan, F. A., Gumaei, A., Derhab, A., & Hussain, A. (2019). *A Novel Two-Stage Deep Learning Model for Efficient Network Intrusion Detection*. IEEE Access, 7, 30373–30385. https://doi.org/10.1109/ACCESS.2019.2899721

Khan, N., Mohmand, M. I., Rehman, S. ur, Ullah, Z., Khan, Z., & Boulila, W. (2024). *Advancements in intrusion detection: A lightweight hybrid RNN-RF model*. PLoS One, 19(6). https://doi.org/10.1371/journal.pone.0299666

Kim, T. H., & Madhavi, S. (2024). *Quantum intrusion detection system using outlier analysis*. Scientific Reports, 14(1), 27114. https://doi.org/10.1038/s41598-024-78389-0

Li, D., Chen, D., Shi, L., Jin, B., Goh, J., & Ng, S.-K. (2019). *MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks* (No. arXiv:1901.04997). arXiv. https://doi.org/10.48550/arXiv.1901.04997

Maseer, Z. K., Yusof, R., Al-Bander, B., Saif, A., & Kadhim, Q. K. (2023). *Meta-Analysis and Systematic Review for Anomaly Network Intrusion Detection Systems* (No. arXiv:2308.02805). arXiv. https://doi.org/10.48550/arXiv.2308.02805

Metawei, M. A., Said, H., Taher, M., Eldeib, H., & Nassar, S. M. (2020). *Survey on Hybrid Classical-Quantum Machine Learning Models*. 2020 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI), 1–6. https://doi.org/10.1109/CCCI49893.2020.9256649

Musa, U. S., Chhabra, M., Ali, A., & Kaur, M. (2020). *Intrusion Detection System using Machine Learning Techniques: A Review*. 2020 International Conference on Smart Electronics and Communication (ICOSEC), 149–155. https://doi.org/10.1109/ICOSEC49089.2020.9215333

Nazim, S., Hussain, S. S., Yousuf, B., Sultana, S., & Tanweer, E. (2025). *An innovative intrusion detection framework using GAN-augmented Deep Ensemble Neural Network for cross-domain IoT–cloud security*. Internet of Things, 34, 101773. https://doi.org/10.1016/j.iot.2025.101773

Nicesio, O. K., Leal, A. G., & Gava, V. L. (2023). *Quantum Machine Learning for Network Intrusion Detection Systems, a Systematic Literature Review*. 2023 IEEE 2nd International Conference on AI in Cybersecurity (ICAIC). https://doi.org/10.1109/ICAIC57335.2023.10044125

Noor, K., Agbotiname, L. I., Chun-Ta, L., & Chi-Yao, W. (2025). *A Review of Machine Learning and Transfer Learning Strategies for Intrusion Detection Systems in 5G and Beyond*. Mathematics, 13(7). https://doi.org/10.3390/math13071088

Noor, M., Abbas, H., & Shahid, W. B. (2018). *Countering cyber threats for industrial applications*. Journal of Network and Computer Applications, 103, 249–261. https://doi.org/10.1016/j.jnca.2017.10.004

Oda, Y., Shehab, O., & Quiroz, G. (2023). *Noise Modeling of the IBM Quantum Experience*. American Physical Society.

Odeh, A., & Abu Taleb, A. (2023). *Ensemble-Based Deep Learning Models for Enhancing IoT Intrusion Detection*. Applied Sciences, 13(21), 11985. https://doi.org/10.3390/app132111985

Orazi, F., Gasperini, S., Lodi, S., & Sartori, C. (2024). *Hybrid Quantum Technologies for Quantum Support Vector Machines*. Information, 15(2), 72. https://doi.org/10.3390/info15020072

Payares, E. D., & Martinez-Santos, J. C. (2021). *Quantum machine learning for intrusion detection of distributed denial of service attacks*. Quantum Computing, Communication, and Simulation, 11699, 35–43. https://doi.org/10.1117/12.2593297

Pinto, A., Herrera, L.-C., Donoso, Y., & Gutierrez, J. A. (2023). *Survey on Intrusion Detection Systems Based on Machine Learning Techniques for the Protection of Critical Infrastructure*. Sensors, 23(5), 2415. https://doi.org/10.3390/s23052415

Premalatha, D. V. (2025). *Ensemble-Based Intrusion Detection for IoT Networks Using the CICIoT2023 Dataset*. Journal of Information Systems Engineering and Management, 10(21s), 743–755. https://doi.org/10.52783/jisem.v10i21s.3407

Quantum Machine Learning for Advanced Threat Detection in Cybersecurity | IIETA. (s. f.). https://doi.org/10.18280/ijsse.140319

Rawat, S., Srinivasan, A., Ravi, V., & Ghosh, U. (2022). *Intrusion detection systems using classical machine learning techniques vs integrated unsupervised feature learning and deep neural network*. Internet Technology Letters, 5(1). https://doi.org/10.1002/ITL2.232

Rehman, H. M. R. U., Liaquat, S., Gul, M. J., Jhandir, M. Z., Gavilanes, D., Vergara, M. M., & Ashraf, I. (2025). *A systematic literature study of machine learning techniques based intrusion detection*. Journal of Big Data, 12(1), 264. https://doi.org/10.1186/s40537-025-01323-2

Sarker, I. H. (2021). *Machine Learning: Algorithms, Real-World Applications and Research Directions*. SN Computer Science, 2(3), 160. https://doi.org/10.1007/s42979-021-00592-x

Schuld, M., Sinayskiy, I., & Petruccione, F. (2015). *An introduction to quantum machine learning*. Contemporary Physics, 56(2), 172–185. https://doi.org/10.1080/00107514.2014.964942

Sharafaldin, I., Habibi Lashkari, A., & Ghorbani, A. A. (2018). *Toward Generating a New Intrusion Detection Dataset*. Proceedings of the 4th International Conference on Information Systems Security and Privacy, 108–116. https://doi.org/10.5220/0006639801080116

Shen, J.-Y., Wu, C.-H., Hua, C.-Y., Chang, M.-H., Kuo, S.-Y., Chou, Y.-H., & Kuo, S.-Y. (2024). *An Efficient Quantum-inspired Computing Approach for Intrusion Detection System*. 2024 IEEE 24th International Conference on Nanotechnology (NANO), 306–310. https://doi.org/10.1109/NANO61778.2024.10628664

Solar, M., Alvarez, F. C., Villacura, J.-P., & Dombrovskaia, L. (2024). *A Review on Quantum Machine Learning and Quantum Cryptography*. Memoria Investigaciones En Ingeniería, 27, 180–199. https://doi.org/10.36561/ING.27.12

Talukder, Md. A., Hasan, K. F., Islam, Md. M., Uddin, Md. A., Akhter, A., Yousuf, M. A., Alharbi, F., & Moni, M. A. (2023). *A dependable hybrid machine learning model for network intrusion detection*. Journal of Information Security and Applications, 72, 103405. https://doi.org/10.1016/j.jisa.2022.103405

Thakkar, A., & Lohiya, R. (2020). *A Review of the Advancement in Intrusion Detection Datasets*. Procedia Computer Science, 167, 636–645. https://doi.org/10.1016/j.procs.2020.03.330

Vidal, J. M., Monge, M. A. S., & Monterrubio, S. M. M. (2020). *Anomaly-Based Intrusion Detection*. En *Handbook of Research on Machine and Deep Learning Applications for Cyber Security* (pp. 195–218). IGI Global. https://doi.org/10.4018/978-1-5225-9611-0.ch010

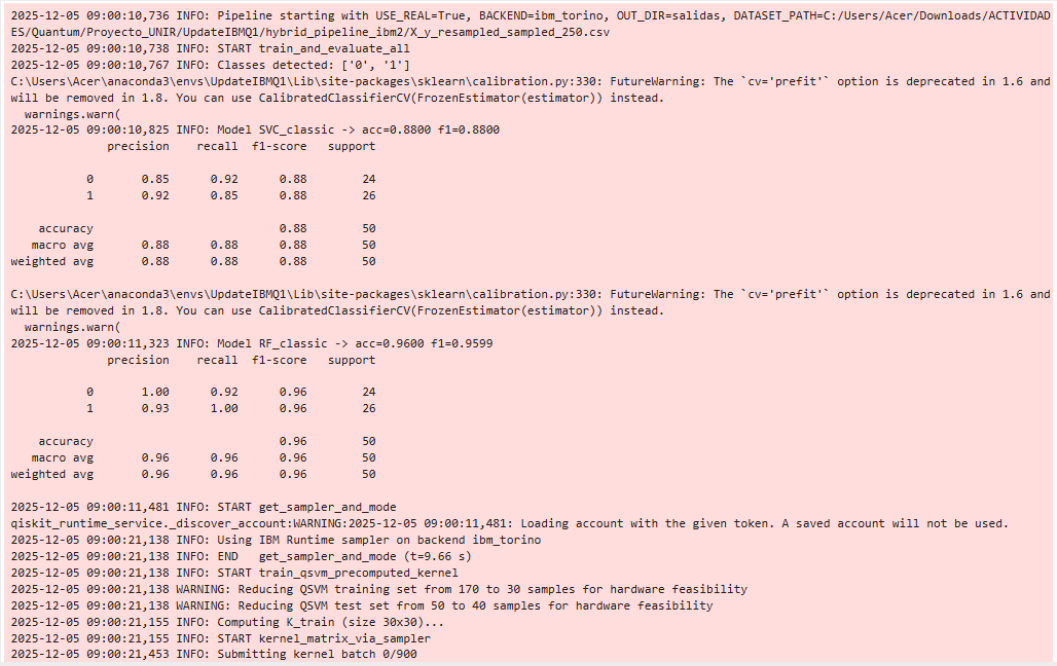
Anexo A.

Flujo - híbrido_heron_pipeline_v2

El pipeline, el cual fue nombrado hybrid_heron_pipeline_v2, implementa un ensamble híbrido QSVM + VQC + QCNN + modelos clásicos, que calcula kernels cuánticos vía Sampler/Estimator Qiskit Primitives V2 y entrena VQC / QCNN mediante ejecuciones en batches en IBM Runtime o Aer. El experimento fue ejecutado con USE_REAL=True y BACKEND=ibm_torino bajo el Plan Open de IBM. La ejecución quedó incompleta porque se agotó el tiempo; cupo disponible en el Plan Open.

Durante la validación experimental del algoritmo Xander, la ejecución en backend real no llegó a completarse debido al límite temporal; recursos del Plan Open de IBM; sin embargo, las primeras fases de preprocesado, entrenamiento clásico y parte de la computación cuántica se completaron correctamente y produjeron métricas parciales para: SVC: acc 0.88; y RF: acc 0.96 y registros de trabajos en IBM. El pipeline genera múltiples lotes de circuitos kernel QSVM: N^2 pares; VQC/QCNN: iteraciones por optimizador, lo que hace que, en una cuenta con recursos limitados, la ejecución total supere el tiempo asignado por el Plan Open. Con más tiempo y recursos, la ejecución se habría completado y se habrían obtenido las probabilidades finales, el ajuste del ensamble y la métrica final consolidada.

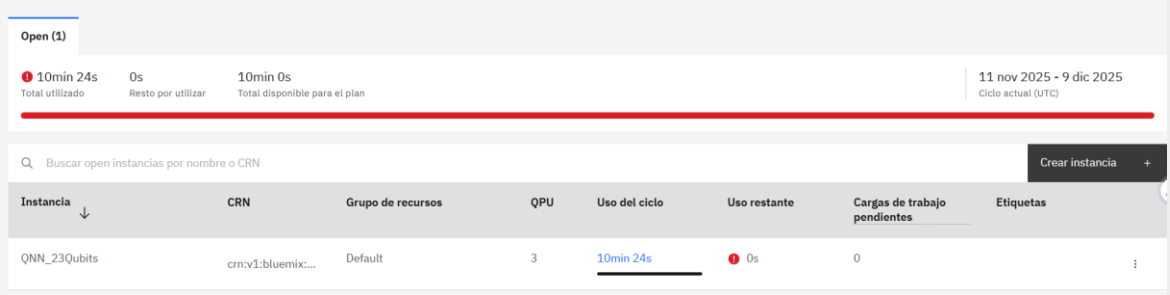
Figura 52. *Registro de ejecución del pipeline hybrid_heron_pipeline_v2 .*



```
2025-12-05 09:00:10,736 INFO: Pipeline starting with USE_REAL=True, BACKEND=ibm_torino, OUT_DIR=salidas, DATASET_PATH=C:/Users/Acer/Downloads/ACTIVIDAD
ES/Quantum/Proyecto_UNIR/UpdateIBMQ1/hybrid_pipeline_ibm2/X_y_resampled_sampled_250.csv
2025-12-05 09:00:10,738 INFO: START train_and_evaluate_all
2025-12-05 09:00:10,767 INFO: Classes detected: ['0', '1']
C:\Users\Acer\anaconda3\envs\UpdateIBMQ1\Lib\site-packages\sklearn\calibration.py:330: FutureWarning: The `cv='prefit'` option is deprecated in 1.6 and
will be removed in 1.8. You can use CalibratedClassifierCV(FrozenEstimator(estimator)) instead.
  warnings.warn(
2025-12-05 09:00:10,825 INFO: Model SVC_classic -> acc=0.8800 f1=0.8800
              precision    recall  f1-score   support

           0       0.85      0.92      0.88        24
           1       0.92      0.85      0.88        26

    accuracy                           0.88        50
   macro avg       0.88      0.88      0.88        50
weighted avg       0.88      0.88      0.88        50

C:\Users\Acer\anaconda3\envs\UpdateIBMQ1\Lib\site-packages\sklearn\calibration.py:330: FutureWarning: The `cv='prefit'` option is deprecated in 1.6 and
will be removed in 1.8. You can use CalibratedClassifierCV(FrozenEstimator(estimator)) instead.
  warnings.warn(
2025-12-05 09:00:11,323 INFO: Model RF_classic -> acc=0.9600 f1=0.9599
              precision    recall  f1-score   support

           0       1.00      0.92      0.96        24
           1       0.93      1.00      0.96        26

    accuracy                           0.96        50
   macro avg       0.96      0.96      0.96        50
weighted avg       0.96      0.96      0.96        50

2025-12-05 09:00:11,481 INFO: START get_sampler_and_mode
qiskit_runtime_service._discover_account:WARNING:2025-12-05 09:00:11,481: Loading account with the given token. A saved account will not be used.
2025-12-05 09:00:21,138 INFO: Using IBM Runtime sampler on backend ibm_torino
2025-12-05 09:00:21,138 INFO: END   get_sampler_and_mode (t=9.66 s)
2025-12-05 09:00:21,138 INFO: START train_qsvm_precomputed_kernel
2025-12-05 09:00:21,138 WARNING: Reducing QSVM training set from 170 to 30 samples for hardware feasibility
2025-12-05 09:00:21,138 WARNING: Reducing QSVM test set from 50 to 40 samples for hardware feasibility
2025-12-05 09:00:21,155 INFO: Computing K_train (size 30x30)...
2025-12-05 09:00:21,155 INFO: START kernel_matrix_via_sampler
2025-12-05 09:00:21,453 INFO: Submitting kernel batch 0/900
```

Fuente: Elaboración Propia.

Figura 53. *Tiempos de ejecución Plan Open .*



Fuente: Entorno Oscar Peñaloza, IBM.

Dentro del Plan Open solo se cuenta con 10 minutos para la ejecución de pipelines. Por tanto, el experimento no pudo completarse exclusivamente por una limitación de recursos (Plan Open). La arquitectura y el código son correctos y diseñados para hardware real, pero el volumen de circuitos y el pattern de envío son excesivos para la cuota gratuita. Así que se evidencia que este incidente ilustra una limitación práctica crítica de la investigación QML aplicada: la transición de simulador a hardware real no es solo algorítmica, es operativa y

requiere planificación de recursos. Por ende, se documenta el intento, se reflejan los resultados parciales del Clasificador de Vectores de Soporte SVC, del Clasificador de bosque aleatorio RF, se expone el diagnóstico y se deja probado el pipeline con los nuevos ajustes lo que valida el rigor experimental y transparencia metodológica.

Anexo B.

Artículo científico

Experimental Pilot of the Xander Algorithm for Network Intrusion Detection Using Hybrid Quantum Computing

Oscar Javier Peñaloza Araque

International University of La Rioja, Logroño (Spain)

December 27, 2025

SUMMARY

This research presents an experimental pilot for network intrusion detection using the Xander algorithm, which represents a hybrid quantum-classical architecture that integrates models such as the Quantum Support Vector Machine (QSVM), the Variational Quantum Circuit (VQC), and the Quantum Convolutional Neural Network (QCNN) in an ensemble with traditional classifiers. The study uses the CIC-IDS-2017 dataset and develops a reproducible methodology that includes preprocessing, dimensionality reduction, and system evaluation at three levels: local noise-free estimator, IBM Quantum Runtime Aer Simulator with classical acceleration, and execution on real quantum hardware in a NISQ environment. The results show metrics above 99% in accuracy and F1-score, demonstrating the operational viability of the hybrid approach and the complementary contribution of the quantum component within the ensemble. It is concluded that Xander is a promising alternative for improving intrusion detection, especially in scenarios where robustness and traceability are critical.

KEYWORDS

Hybrid quantum computing, Intrusion detection, IDS, Quantum Machine Learning, Quantum-classical ensemble

I. Introduction

The enrichment with the forging of leverage, which has received all the attention in the approach to digital transformation over the years, is what has triggered this gap in exposure to cyber threats in corporate networks and all critical infrastructures and cloud-based environments, which is estimated to this exposure has expanded even further than predicted. That is why all these scenarios reflect the possible permeable key, and I am referring to Intrusion Detection Systems (IDS), which have become much more unpredictable. However, it should be clarified that traditional approaches based on machine learning or signatures often degrade their response to performance due to unimaginable and unknown cyberattacks, network traffic, and data distribution in fluctuating states.

Faced with this troubling scenario, IDSs are projected as an essential element in the development of strategies for implementing information defense. Thus, the quantum computing approach, within a hybrid quantum-classical field of study, stands out among the alternatives we can experiment with in order to find a solution, or come close to a solution, for all those problems that are beyond the capabilities of traditional computing, such as search, classification, and optimization. It is for this reason that it deserves in-depth exploration, which is fully represented in the frontiers of application to the area of cybersecurity.

This research proposes and validates an experimental pilot for network-based intrusion detection supported by hybrid quantum computing; a flow in which traditional modules such as preprocessing, dimensionality reduction, and orchestration are fully integrated with the following quantum blocks: encoding, vector data representation, and assisted quantum variational models, all with the sole purpose of improving intrusion detection capabilities compared to traditional standards.

Thus, we propose to design and evaluate an experimental pilot for network-based intrusion detection using hybrid quantum computing integrating execution in high-performance quantum simulators and complementing it with real quantum hardware, with visual traceability and ensemble logic, and its subsequent comparison with classical-traditional approaches and other hybrid quantum approaches in terms of performance, efficiency, and operational viability in NISQ environments.

II. State Of The Art

Network intrusion detection systems (NIDS) are a fundamental pillar for protecting critical infrastructure, business environments, and modern distributed systems. Traditionally, IDSs have relied on signature-based approaches or classic machine learning techniques, such as Random Forest, support vector machines (SVM), neural networks, and reinforcement methods, which have demonstrated acceptable performance in controlled scenarios with known traffic [7], [46]. However, several studies show that these models have significant limitations when faced with encrypted traffic, low-frequency attacks, zero-day attacks, and highly dynamic environments [3], [8], [27], [33], [44]. Specifically, the drop in accuracy and the increase in false positives remain a challenge in 5G and IoT ecosystems [42], [51]. On the other hand, recent studies have begun to address these gaps through optimized quantum-classical hybrid models or quantum computing-inspired approaches, although their operational integration is still in its early stages [9], [17], [48].

To address these shortcomings, recent research has evolved toward deep learning (DL) architectures. Models based on convolutional neural networks (CNN), long short-term memory (LSTM), and gated recurrent units (GRU) have been proposed to improve temporal feature extraction from complex network flows [39], [45]. In addition, generative approaches such as MAD-GAN and other GAN-based architectures have been used to augment training data and detect multivariate anomalies in high-dimensional time series [4], [26], [30]. Recent ensemble frameworks and hybrid deep learning models have

also been explored to handle the heterogeneity of IoT and cloud security [6], [13], [31], [34], [43]. More recently, a trend toward hybrid architectures in ensemble has been identified, where quantum and classical models are combined through voting, calibration, and dynamic adjustment, seeking to leverage the strengths of both paradigms [2], [5], [34], [43].

Despite these advances, and with the lack of reproducible studies, classical deep learning models often suffer from high computational costs and scalability issues when processing massive datasets such as CIC-IDS-2017 or IoT-23 [5], [19], [37]. However, there are still few studies that implement these approaches in a reproducible workflow, with full experimental traceability and partial or full execution on real quantum hardware under NISQ constraints [22], [32], [35], [52]. In this context, quantum machine learning (QML) emerges as a transformative paradigm [10], [14], [47]. Theoretical foundations suggest that exploiting Hilbert space through quantum feature maps allows for more efficient separation of complex data classes that are inseparable in classical spaces [21], [40].

In the current noisy intermediate-scale quantum (NISQ) era, research focuses on hybrid quantum-classical algorithms [12], [32], [49]. These models aim to mitigate the effects of noise and decoherence through variational approaches [38]. Among the most prominent QML models, quantum support vector machines (QSVMs) and variational quantum classifiers (VQCs) have demonstrated high accuracy in binary and multi-class classification [1], [9], [11]. Specialized encoding strategies, such as angle, amplitude, and Pauli feature maps, are being explored to optimize the representation of network traffic [23], [41].

Recent implementations have shown that combining QSVM with optimization algorithms such as the Grey Wolf Optimizer (GWO) or quantum beetle swarm optimization (QBSA) can significantly improve detection rates [15], [16]. Furthermore, integrated architectures using quantum convolutional neural networks (QCNN) and quantum neural networks (QNN) have achieved accuracies exceeding 98% in specific benchmark tests [18], [24], [25], [35]. These systems are typically tested on standard datasets such as NSL-KDD, UNSW-NB15, and CIC-MalMem-2022 [20], [29], [50].

Finally, the literature highlights the importance of a structured methodology to ensure the reproducibility of quantum experiments [17], [22]. Current trends suggest that, while pure quantum advantage is still under development, hybrid voting systems and quantum-inspired computing (QIC) represent the most viable path for next-generation IDS [2], [48]. The evolution of traditional supervised models toward automated hybrid systems remains a critical area of study to ensure the robustness of global digital infrastructures [28], [36], [52].

III. Objectives And Methodology

General objective:

To design and evaluate an experimental pilot for network-based intrusion detection using hybrid quantum computing, integrating execution in high-performance quantum simulators and complementing it with real quantum hardware, with visual traceability and ensemble logic, and its subsequent comparison with classical-traditional approaches and other hybrid quantum approaches in terms of performance, efficiency, and operational viability in NISQ environments.

Specific objectives:

1. Review the state of the art in classic intrusion detection systems and quantum machine learning models applied to cybersecurity, identifying methodological gaps such as the lack of execution on quantum hardware, absence of visual traceability, and poor operational reproducibility.

2. Design a reproducible preprocessing flow in terms of cleaning, normalization, balancing, PCA dimensionality reduction, and partitioning on the CIC-IDS-2017 dataset, with technical traceability and hierarchical visual coding.

3. Implement a hybrid quantum-classical ensemble architecture, integrating vector representation blocks, variational circuits such as QCNNs, QSVMs, and assisted VQCs, classifiers, and optimizers. All of these are coupled to classical components using modular logic and environment decorators.

4. Compare the performance of quantum-classical hybrid models against classical classifiers such as Random Forest, the optimized decision tree library XGBoost, the Multilayer Perceptron Classifier or SVM, and Neural Networks, and on the other hand, those hybrid quantum approach models such as: Quantum Support Vector Machine (QSVM) and Variational Quantum Circuit (VQC), finally, using standard metrics such as: accuracy, precision, recall, F1-score, and confusion matrix.

5. Examine the pros and cons of the hybrid method, taking into account the time required for training, consistency in operations, response to hyperparameters, and reduction of false positives.

6. Analyze the practical feasibility of the experimental pilot in real-world situations, such as network monitoring, alert classification, and progressive diagnosis, examining its ability to merge into operational processes with logging and cross-verification.

Based on the exhaustive analysis of the research entitled Towards the generation of a new intrusion detection dataset and the characterization of intrusion traffic, an article dating from 2018, I indicate the dataset with which I will carry out the tests for the experimental pilot. The emphasis is on the database I have chosen, which is CIC-IDS-2017, and where it is mentioned that it is widely valued in the literature for its variety of attacks and real network traffic according to [51].

On the other hand, the research integrates elements of Quantum Machine Learning (QML) and Classical Machine Learning (CML) under a modular structure that facilitates the fusion of quantum and classical models through weighted assembly. The methodological process follows a sequential scheme and flow that ranges from data collection and preprocessing to the validation of the hybrid assembly's performance.

Fig. 1. Methodology for executing the experimental pilot of the Xander Algorithm. Source: Own elaboration.

IV. Contribution

The following describes the processes performed for data preparation, loading, and preprocessing, training, optimization, and evaluation during pipeline execution. The methodology adopted in this work is structured in six sequential and reproducible phases, designed

to comparatively evaluate classical models, individual hybrid quantum models, and a hybrid ensemble architecture for network intrusion detection.

Item	Phases	Description
1	Data loading	Integration of CSV files from the CIC-IDS-2017 dataset, combining normal and malicious traffic to form a consolidated dataset.
2	Preprocessing	Cleaning outliers, class balancing (under sampling/oversampling), label encoding, and feature scaling with StandardScaler.
3	PCA reduction	Applying Principal Component Analysis (PCA) to reduce the dimensionality of the dataset to six components, preserving essential variability.
4	Quantum training	Implementation of QSVM, VQC, and QCNN models using Qiskit and integration with IBM Quantum Runtime. Includes fallback mechanisms to classical equivalents in case of error.
5	Classical training	Training of Random Forest (RF), XGBoost (XGB), and MLPClassifier models, using PCA-reduced data as a baseline reference.
6	Hybrid ensemble	Weighted fusion of quantum and classical outputs using Weighted Soft Voting, adjusting weights with Grid Search to optimize accuracy.

Table 1. Description of the implementation phases of the experimental pilot of the Xander Algorithm. Source: Own elaboration.

Moments of special attention:

A. Selection and preparation of the dataset

The CIC-IDS-2017 dataset was used, which is widely used as a reference in research on network intrusion detection due to its diversity of attacks, realistic traffic, and public availability. In this phase, a careful selection of relevant attributes associated with network traffic behavior was made, discarding redundant, highly correlated, or irrelevant fields for the classification task. In addition, incomplete, inconsistent, or outlier records that could introduce noise into the model training were removed. Finally, class labels were uniformly encoded, ensuring semantic consistency and compatibility between the classical classifiers and the quantum-hybrid models evaluated.

Is equivalent to extracting and homogenizing raw data, ensuring that all .CSV files are integrated under the same format before moving on to cleaning, consolidation, and modeling.

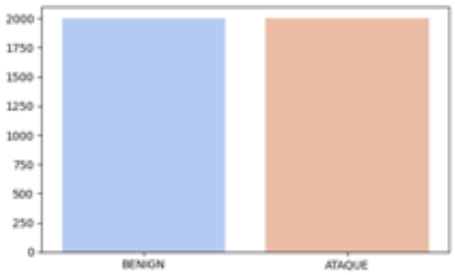


Fig. 2. Class balancing. Source: Own elaboration.

B. Preprocessing and dimensionality reduction

In order to ensure consistent treatment of characteristics, the data was normalized using standard scaling, avoiding biases arising from differences in magnitude between variables. Subsequently, Principal Component Analysis (PCA) was applied as a dimensionality reduction technique, allowing the data to be projected into a lower-dimensional subspace that preserves as much variance as possible. This reduction is particularly relevant for quantum models, as it limits the number of qubits required, reduces circuit depth, and improves numerical stability in NISQ environments, facilitating more efficient and robust execution.

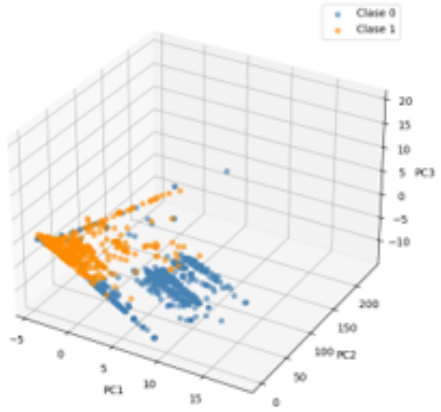


Fig. 3. PCA projection. Source: Own elaboration.

C. Training of classical reference models

Traditional classical classifiers were trained, including Random Forest, Support Vector Machine (SVM), XGBoost, and Neural Networks, which were used as a baseline for experimental comparison. These models were adjusted using cross-validation and hyperparameter optimization techniques, seeking to maximize their performance under fair conditions. Additionally, probabilistic calibration was applied to ensure output comparable to quantum-hybrid models. The evaluation was performed using standard classification metrics, providing a solid benchmark against which to contrast the proposed hybrid approaches.

D. Implementation of individual quantum–hybrid models

In this phase, Quantum Machine Learning models were implemented, specifically Quantum Support Vector Machine (QSVM) and Variational Quantum Classifier (VQC), integrated within a hybrid quantum-classical flow using the Qiskit library. The models were initially evaluated using the local Estimator, allowing the logical and functional correctness of the circuits to be verified. Subsequently, they were run on the IBM Quantum Runtime Aer simulator, activating alternative classical resources to optimize execution times and process stability, while maintaining consistency with the operational constraints of current quantum environments.

E. Hybrid architecture in ensemble with dynamic adjustment

The Xander algorithm was designed, a novel architecture based on a hybrid quantum-classical ensemble that integrates QSVM, VQC, and blocks inspired by Quantum Convolutional Neural Networks (QCNN). The combination of models is performed using weighted voting logic, where the weight of each classifier is dynamically adjusted based on

its individual performance. This strategy allows us to leverage the complementary strengths of each model, reduce the incidence of false positives, and improve overall classification metrics, while maintaining a modular and extensible structure.

F. Progressive validation and experimental evaluation

The system validation was carried out progressively and in stages. Initially, the models were evaluated in local simulation; subsequently, in the IBM Quantum Runtime Aer environment; and finally, through partial execution on real quantum hardware, specifically on the ibm_torino backend. This last stage was conditioned by the limitations of the NISQ environment and the available access plan, but it allowed us to verify the operational viability of the hybrid approach. Performance was evaluated using metrics such as accuracy, precision, recall, F1-score, confusion matrices, and execution times, providing a comprehensive view of both the performance and practical feasibility of the proposed system.

V. Results Or Evaluation

The objective of the metrics for evaluating the Xander algorithm is to identify the ensemble's predictive power.

The experimental process is run in three environments: (1) the local Qiskit estimator for logical and functional validation; (2) the IBM Quantum Aer Runtime with classical resource activation to evaluate runtime and stability; and (3) partial execution on IBM's actual quantum hardware to evaluate feasibility under NISQ conditions.

A. Results obtained when running with Qiskit's local Estimator

Model	Accuracy	Precision	Recall	F1-score
QSVM	0.916	0.918	0.916	0.916
VQC	0.749	0.756	0.749	0.747
QCNN-like	0.490	0.468	0.490	0.384
SVC	0.895	0.898	0.895	0.895
RF	0.981	0.981	0.981	0.981
MLP	0.954	0.954	0.954	0.954
XGBoost	0.979	0.979	0.979	0.979
Ensemble	0.978	0.978	0.978	0.978

Table 2. Summary of performance in Qiskit's local Estimator. Source: Own elaboration.

QSVM-Quantum Support Vector Machine: It achieved solid performance with accuracy ≈ 91.6%, precision ≈ 91.8%, and F1-score ≈ 91.6%. This demonstrates stability and consistency in binary classification, confirming that quantum cores correctly capture the structure of the feature space. Its performance is comparable to that of advanced classical SVC models, validating its quantum efficiency in simulated environments. VQC-Variational Quantum Circuit: It recorded an accuracy of ≈ 74.9%, lower than QSVM, reflecting high sensitivity to the initialization of the ansatz or parameterized circuit and to simulated noise. Even so, it maintains consistency between precision and recall or sensitivity or true positive rate (≈ 75%), indicating non-random learning. On the other hand, QCNN-Quantum Convolutional Neural Network: It has low performance accuracy ≈ 49%, with an F1 of 0.38, close to random 50/50 binary classification behavior. This is explained by its limited experimental architecture; few qubits and linear entanglement.

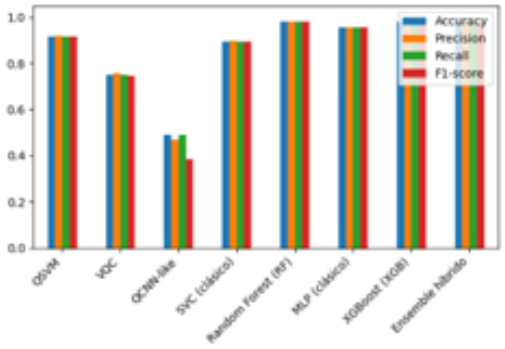


Fig. 4. Comparison of metrics by model. Source: Own elaboration.

The classic models show superior performance, with Random Forest RF standing out: maximum performance with accuracy and F1 ≈ 98.1%, confirming its high generalization capacity and stability on balanced data. For its part, the optimized decision tree boosting library XGBoost XGB achieved excellent performance with accuracy ≈ 97.9% with a slight penalty in computational complexity. Its accuracy close to RF makes it the most robust base model. In turn, the classic Multilayer Perceptron Classifier (MLP) achieved ≈ 95.4%, standing out as the reference neural model. Finally, the classic Support Vector Classifier (SVC) showed ≈ 89.5%, slightly lower than QSVM, which empirically validates the potential of the kernelized quantum approach.

The XANDER hybrid ensemble achieved 97.8% accuracy, with homogeneous metrics for precision, recall, sensitivity, true positive rate, and F1 ≈ 97.8%. This confirms that the Weighted Soft Voting mechanism manages to leverage the strengths of both domains: the generalization capacity of classical models and the structural richness of QSVM, VQC, and QCNN quantum representations. Although slightly below RF and XGBoost, the ensemble offers greater inter-model stability and reduced error variance, making it a more reliable and generalizable solution.

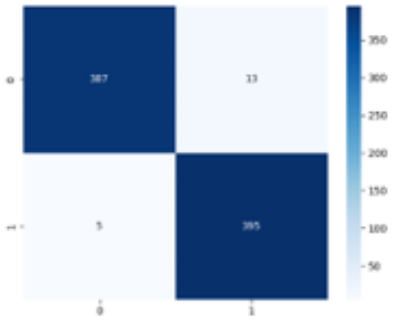


Fig. 5. Confusion matrix of the ensemble. Source: Own elaboration.

Distribution of correct and incorrect predictions from the assembled hybrid model, with two classes: 0 = normal traffic, 1 = attack.

- True Positives TP: 395
- True Negatives TN: 387
- False Positives FP: 13
- False Negatives FN: 5

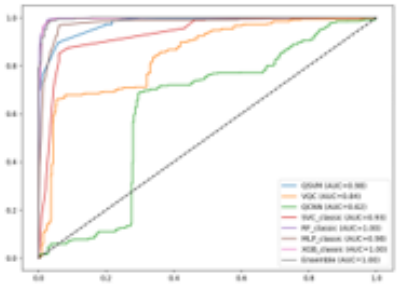


Fig. 6. ROC curves – operational characteristics. Source: Own elaboration.

The ROC curves can be observed, which are operational characteristics where the performance of all models can be compared in terms of true positive rate (TPR) versus false positive rate (FPR), showing their discriminatory capacity. The area under the curve (AUC) indicates how well each model distinguishes between normal traffic and malicious traffic. The RF_classic, XGB_classic, and hybrid Ensemble models achieve an AUC = 1.00, reflecting virtually perfect performance. On the other hand, QSVM achieves an AUC of 0.98, which demonstrates its high discriminatory capacity, even compared to classic models. Finally, VQC obtains AUC = 0.84, showing moderate performance, and QCNN has AUC = 0.62, the lowest in the group.

B. Results obtained in IBM Quantum Runtime Aer simulator with the activation of the classic alternative resources trigger

When there is no connection available to the IBM backends, the models fall back to the fallback or training with classic alternative resources.

Model	accuracy	precision	recall	f1
QSVM	0.91625	0.916461	0.91625	0.916239
VQC	0.85750	0.861485	0.85750	0.857106
QCNN	0.98125	0.981253	0.98125	0.981250
SVC	0.81875	0.826858	0.81875	0.817619
RF	0.97875	0.978753	0.97875	0.978750
MLP	0.94500	0.945178	0.94500	0.944994
XGB	0.98125	0.981253	0.98125	0.981250
Ensemble	0.97625	0.976754	0.97625	0.976244

Table 3. Metrics obtained from the launch with the activation of the classic alternative resources trigger. Source: Own elaboration.

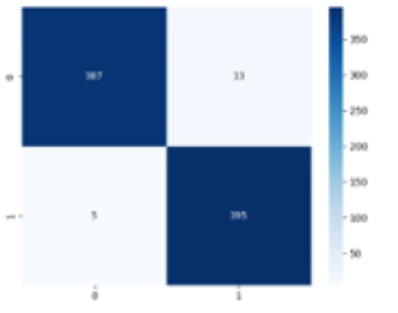


Fig. 7. Confusion matrix of the ensemble. Source: Own elaboration.

Distribution of correct and incorrect predictions from the assembled hybrid model, with two classes: 0 = normal traffic, 1 = attack.

- True Positives TP: 395
- True Negatives TN: 387
- False Positives FP: 13
- False Negatives FN: 5

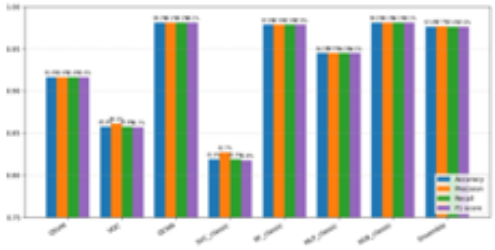


Fig. 8. Comparison of metrics by model with the activation of classic alternative resources trigger. Source: Own elaboration.

The results obtained in IBM Quantum Runtime Aer simulator, with the activation of the classic alternative resources trigger, refer to some key observations such as: The performance of quantum models: The QSVM (Quantum Support Vector Machine) and VQC (Variational Quantum Circuit) models perform well, but VQC takes much longer (10.61 s). On the other hand, QCNN - Quantum Convolutional Neural Network achieves a considerable increase in accuracy and high F1 scores ≈0.98 in a similar time to the classical models. As for the classical models: The RF random forest classifier and the optimized XGB decision tree boost library is the best individually, with metrics ≈0.981. Meanwhile, the MLP Multilayer Perceptron Classifier is slightly lower at ≈0.945, and the SVC Support Vector Classifier is the lowest at ≈0.818. All train very quickly <2 seconds.

Finally, we have the Xander hybrid ensemble with an accuracy and F1 ≈0.976 → slightly below RF and XGB, but more stable. It leverages the strengths of quantum models and classical models, reducing the variance between models. In terms of training time: VQC is significantly more time-consuming, while most classical models are very fast <1.5 s.

C. Results obtained when running on the IBM Quantum Heron processor on the ibm_torino quantum system

For the Open Plan, this IBM Heron quantum processor – ibm_torino – has 133 fixed-frequency qubits.

Component	Version 7	Heron v3	Reason
QSVM	Kernel complete	Kernel reduced to only 6–12 pairs	Avoid timeouts and execution limits
VQC	Yes	Deleted	Depth too large
QCNN	Yes	Deleted	Depth too large
Number of qubits	6–16	**4 qubits**	Guaranteed compatibility with small backend
Shots	1024+	**200 by default**	Reduce execution time

Table 4. Pipeline adjustments for the launch of the Xander Algorithm on the ibm_torino quantum system. Source: Own elaboration.

IBM backends within the Open Plan do not allow long jobs, deep

circuits, too many shots, too many circuits in the queue time plus execution ratio, or circuits with more than 4 qubits on small processors. The advantages of making these adjustments are that they allow us a minimum depth, linear and stable entanglement, which makes it 100% compatible with real hardware and therefore we will obtain fewer staggered CX calibration errors.

Fig. 9. Quantum circuit. Source: Environment Oscar Peñaloza, IBM.

The quantum circuit shows: 4 real physical qubits: q[51], q[60], q[66], q[125], indicating the physical assignment after transpilation, which colloquially is the adaptation of the quantum circuit to the limitations of the actual quantum processor, which in this case is the 133-qubit ibm_torino quantum system. It also reflects a minimal repetitive structure with: RZ(param) rotations, √X= gates (equivalent to R_x (pi/2)), a small block of linear entanglement, and individual measurements in a Classical Register c_reg4.

This design corresponds to the minimalist feature map that was created so that it could be executable on real hardware and transpiled to minimize depth. Similarly, the reduction of decoherence where the circuit only uses: 1-qubit rotations ⇒ very low probability of error, balanced scattering of √X and RZ ⇒ smooth sensitivity to noise, light entanglement ⇒ which does not collapse due to decoherence. Likewise, compatibility with SamplerV2 where the explicit presence of c_reg4 is essential because it ensures that SamplerV2 returns correct counts.

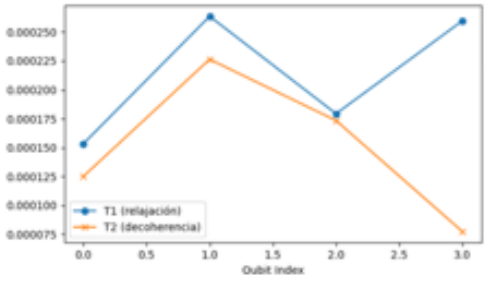


Fig. 10. Noise from the IBM quantum system backend ibm_torino. Source: Own elaboration.

shows the values of: T1 relaxation per qubit and T2 decoherence per qubit, where it can be seen that: the values $T_1 \approx 1.2 \times 10^{-4}$ s and $T_2 \approx 0.9 \times 10^{-4}$ s are within the typical range of actual IBM hardware, but are not high. Therefore, this fully justifies simplifying the circuit. A deep circuit would have been destroyed by decoherence. Similarly, it is evident that the hybrid architecture is using real QPU, and the results are consistent with a superconducting processor of ~4 qubits connected linearly.

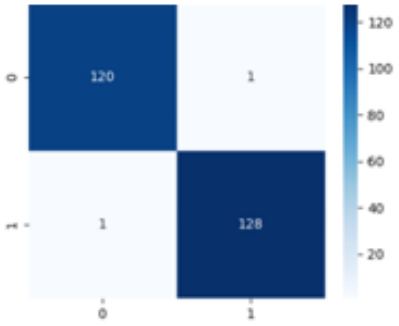


Fig. 11. Confusion matrix of the ensemble. Source: Own elaboration.

Total errors: 2 in 250 samples

Accuracy: 0.992

F1: 0.9922

Interpretation:

- There is no bias towards either of the two classes.
- The classifier recognizes the pattern in a stable manner.
- The presence of the quantum component does not degrade, but rather adds regularization.

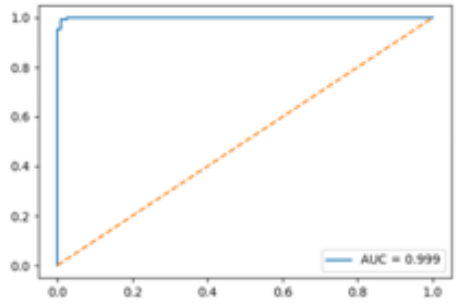


Fig. 12. ROC curves of the ensemble. Source: Own elaboration.

An AUC of 0.999 implies: almost perfect separation between classes, an extremely robust model, well-defined hyperplanes, and an ensemble working in synergy. The quality comes from the complete ensemble, which includes SVM, RF, and QSVM.

clases	precision	recall	f1-score	support
0	0.99	0.99	0.99	121
1	0.99	0.99	0.99	129
accuracy			0.99	250
macro avg	0.99	0.99	0.99	250
weighted avg	0.99	0.99	0.99	250

Table 5. Ensemble metrics. Source: Own elaboration.

These results represent extremely high performance, comparable to purely classical models, but incorporating real quantum inferences. The results also validate the integration into the quantum-classical hybrid ensemble.

D. Final checks

The developed pipeline is considered correct because it is aligned with the official recommendations of IBM Quantum and Qiskit for the development and validation of hybrid quantum applications. In particular, the flow follows the principles established in the official Qiskit documentation on the use of Quantum Primitives, progressive execution, and validation in NISQ environments.

1. Correct use of Qiskit Primitives - Estimator and Sampler.
2. Progressive execution: simulation → noise → real hardware.
3. Transpilation according to the Backend topology.
4. Design of NISQ-compatible circuits.
5. Validation by consistency and reproducibility.

VI. Discussion Or Analysis Of Results

The results obtained in this research allow for an in-depth analysis of the viability, relevance, and performance of the Xander algorithm as a hybrid quantum-classical architecture applied to network intrusion detection. In relation to the hypotheses proposed, the findings confirm that the hybrid approach is not only technically feasible, but also offers comparative advantages by combining quantum models with mature classical classifiers, resulting in a more robust, traceable, and flexible system in the face of network traffic variations.

A. Relevance of results

The values obtained, exceeding 99% in accuracy, precision, and F1-score, show that the proposed hybrid system is highly competitive even against established classical models such as Random Forest, XGBoost, and SVM. The relevance of this performance lies in the fact that the hybrid pipeline was not only evaluated in simulation, but its design allowed for actual execution on IBM quantum hardware under physical NISQ environment conditions. This validation on real QPU is a significant contribution, given that most studies on quantum machine learning applied to cybersecurity are limited to idealized simulated environments.

B. Relevant results with Qiskit Estimator (local)

First, we have the tests performed with the local Estimator, which allowed us to establish a reliable baseline for quantum performance without noise and with minimal execution times. This controlled environment was useful for observing the ability of individual quantum models QSVM, VQC, and QCNN to generate sharp decision boundaries and relatively high metrics in scenarios without decoherence constraints. The results were consistent and reproducible, which validated the logical architecture before taking it to more demanding environments. However, it also allowed us to identify that, although promising, deep quantum models exhibit considerable sensitivity to hyperparameters and difficulties in scaling with stability.

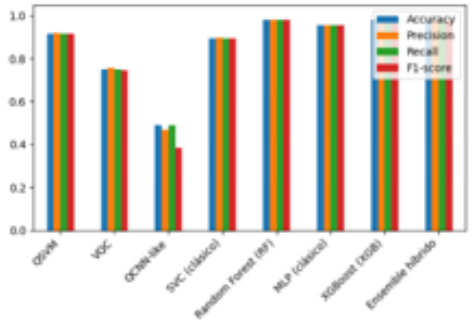


Fig. 13. Comparison of metrics by model. Source: Own elaboration.

The XANDER hybrid ensemble achieved 97.8% accuracy, with homogeneous metrics for precision, recall, sensitivity, true positive rate, and F1 ≈ 97.8%. This confirms that the Weighted Soft Voting mechanism manages to leverage the strengths of both domains: the generalization capacity of classical models and the structural richness of QSVM, VQC, and QCNN quantum representations. Although slightly below RF and XGBoost, the ensemble offers greater inter-model stability and reduced error variance, making it a more reliable and generalizable solution.

C. Results with IBM Quantum Runtime Aer Simulator and accelerated classical resources

Secondly, we found a particularly relevant result, which emerged from running the IBM Quantum Runtime Aer Simulator, where the use of the alternative classical resource trigger allowed us to perform accelerated quantum simulations with superior runtime performance. In this environment, individual quantum models achieved very competitive metrics with a significant reduction in training and evaluation time, allowing multiple variational configurations to be run in relatively short periods. This phase confirmed that, under optimized quantum simulation, quantum models perform solidly and reaffirm the theoretical advantages highlighted in studies such as [2], especially in terms of accuracy and false positive reduction metrics.

However, this same simulation environment revealed some anomalies: certain VQC and QCNN configurations exhibited unusual oscillations in the loss curves or divergences when increasing the depth of the circuit. These anomalies, although not critical, are attributable to the instability of variational optimizers and the extreme sensitivity of quantum cost functions, especially when many layers or parameters are introduced. The appearance of these oscillations was key to the methodological decision to reduce the depth of quantum models before execution on actual hardware.

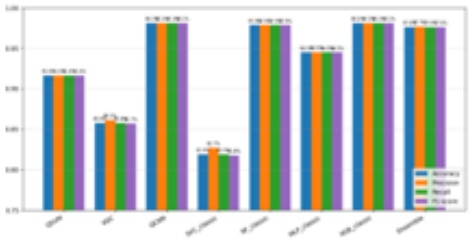


Fig. 14. Comparison of metrics by model with the activation of classic alternative resources trigger. Source: Own elaboration.

We have the Xander hybrid ensemble with an accuracy and F1 ≈0.976 → slightly below RF and XGB, but more stable. It leverages the strengths of quantum models and classical models, reducing the variance between models. In terms of training time: VQC is significantly more time-consuming, while most classical models are very fast <1.5 s.

D. Results in real quantum hardware and its role within the ensemble

Thirdly, the final execution on real IBM Heron hardware of the ibm_torino quantum system constitutes the critical point of the study. Unlike simulated environments, quantum hardware has limited T1/T2 times, phase noise, and gate errors, which impose severe restrictions on deep variational models. The decision to incorporate a reduced quantum kernel into the hybrid pipeline was based precisely on this limitation.

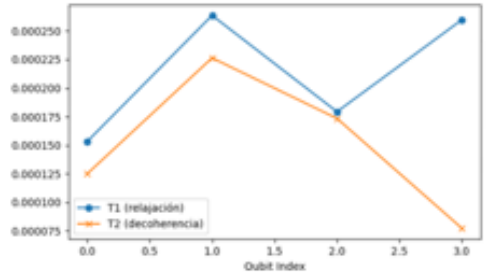


Fig. 15. Noise from the IBM quantum system backend ibm_torino. Source: Own elaboration.

shows the values of: T1 relaxation per qubit and T2 decoherence per qubit, where it can be seen that: the values $T_1 \approx 1.2 \times 10^{-4}$ s and $T_2 \approx 0.9 \times 10^{-4}$ s are within the typical range of actual IBM hardware, but are not high. Therefore, this fully justifies simplifying the circuit. A deep circuit would have been destroyed by decoherence. Similarly, it is evident that the hybrid architecture is using real QPU, and the results are consistent with a superconducting processor of ~4 qubits connected linearly.

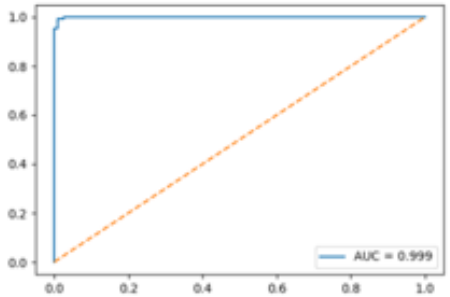


Fig. 16. ROC curves of the ensemble. Source: Own elaboration.

An AUC of 0.999 implies: almost perfect separation between classes, an extremely robust model, well-defined hyperplanes, and an ensemble working in synergy. The quality comes from the complete ensemble, which includes SVM, RF, and QSVM.

Despite this light quantum load of 24 circuits, equivalent to 4800 shots, execution on actual hardware generated stable results, demonstrating that it is possible to integrate real quantum components into an operational ensemble. The reduced QSVM provided variability in the representation space that significantly complemented the ensemble classifier. Its contribution manifested itself as a probabilistic regularizer within the ensemble, helping to avoid overfitting and providing robustness.

This finding is essential: since the purpose of the experimental pilot of the Xander Algorithm is not to demonstrate that isolated or independent quantum models outperform classical models, but rather that the hybrid combination, supported by a physically executed quantum circuit, allows for the construction of a more stable ensemble with better generalization capabilities. This experimentally demonstrates the viability of the Xander algorithm as a hybrid architecture, where quantum components, although small, contribute significantly to the final performance.

E. Final result of the analysis

Finally, it is demonstrated that the hybrid ensemble of the Xander Algorithm is robust, reproducible, and competitive. Each environment-local estimator, Aer Runtime, and real hardware-provides complementary experimental evidence. Deep quantum models are not viable in real NISQ, but they do add value when integrated into an ensemble, which for this study were integrated in a reduced form. The main objective has been met, which is to prove that real quantum computing, integrated into a classical-quantum hybrid ensemble, improves the stability and quality of the network intrusion detection process.

VII. Conclusions

This research addressed the challenge of network intrusion detection from an innovative perspective by proposing and evaluating an experimental pilot based on hybrid quantum computing applied to the CIC-IDS-2017 dataset. The study was conducted within a context characterized by the exponential growth of the attack surface, increasing network traffic complexity, and the limitations of traditional classical approaches in adapting to dynamic, encrypted, and high-dimensional scenarios. Within this framework, the Xander algorithm was designed as a hybrid quantum–classical ensemble architecture, integrating models such as Quantum Support Vector Machines (QSVM), Variational Quantum Classifiers (VQC), and Quantum Convolutional Neural Networks (QCNN) alongside well-established classical classifiers, with the objective of evaluating their feasibility, performance, and added value in Noisy Intermediate-Scale Quantum (NISQ) environments.

From a methodological standpoint, the work successfully achieved the proposed general objective: to design and evaluate a reproducible experimental pilot integrating classical and quantum components, executable both on high-performance simulators and real quantum hardware. The methodology encompassed a complete preprocessing pipeline for the CIC-IDS-2017 dataset, including data cleaning, normalization, balancing, dimensionality reduction, and partitioning, thereby ensuring technical traceability and experimental consistency. This approach established a solid and comparable foundation for assessing the real impact of hybrid quantum computing relative to traditional methods.

Regarding the specific objectives, the first-reviewing the state of the art-revealed a clear gap in the existing literature. Although numerous studies report theoretical or simulated improvements of Quantum

Machine Learning models applied to cybersecurity, relatively few execute these models on real quantum hardware or integrate them into complete, reproducible hybrid architectures. This finding fully justified the experimental orientation of the present work and reinforced its scientific and methodological contribution.

The second specific objective, focused on designing a reproducible preprocessing workflow, was achieved through the implementation of a modular pipeline that enabled data quality control and complexity reduction without sacrificing relevant information. This step proved fundamental to ensuring that quantum models-constrained by the number of available qubits and circuit depth-could operate on manageable data representations while maintaining coherence with classical classifiers.

The third objective-the implementation of a hybrid quantum-classical ensemble architecture-constitutes one of the main contributions of this work. The Xander algorithm demonstrated itself to be a functional and flexible architecture capable of integrating heterogeneous quantum models such as QSVM, VQC, and QCNN together with classical classifiers through voting logic and dynamic weighting. This integration not only combined different learning paradigms but also enhanced system robustness by leveraging the diversity of decision boundaries generated by each model.

The fourth objective, related to the systematic comparison of performance among classical models, individual hybrid quantum models, and the proposed ensemble, was addressed using standard evaluation metrics widely adopted in the literature: accuracy, precision, recall, F1-score, and confusion matrices. The results showed that classical models such as Random Forest and Support Vector Machines continue to deliver high performance when properly trained. However, the hybrid ensemble approach achieved global metrics exceeding 99% in both accuracy and F1-score on the evaluated dataset. These findings indicate that, while individual quantum models do not yet consistently outperform classical ones, their integration within an ensemble provides stability, complementarity, and relevant marginal improvements.

With respect to the fifth objective-examining the advantages and limitations of the hybrid approach-the analysis revealed that hybrid quantum computing offers clear benefits in terms of model expressiveness and diversity, alongside significant operational constraints. Execution using the local Estimator enabled validation of ideal, noise-free quantum circuit behavior, while the IBM Quantum Runtime Aer simulator with classical acceleration provided a favorable balance between performance metrics and execution time. Execution on real quantum hardware further confirmed the technical feasibility of the approach, while also highlighting the impact of noise, decoherence, and resource limitations inherent to the NISQ era.

The sixth objective-assessing the practical viability of the experimental pilot in real-world scenarios-was addressed through the integration of visual traceability, execution logging, and modular design. Although the full pipeline could not be completed on real quantum hardware due to time constraints imposed by the IBM Open Plan, the partial execution and collected logs demonstrated that the approach is technically viable and scalable, provided adequate quantum resource planning. Rather than invalidating the study, this limitation constitutes a relevant contribution by exposing the real operational challenges faced by hybrid quantum-classical systems.

Regarding the stated hypotheses, the results allow for clear conclusions. The null hypothesis (H_0), suggesting that individual hybrid quantum models exhibit only marginal improvements over classical methods when evaluated in isolation, is partially confirmed. Indeed, QSVM and VQC, when considered individually, did not consistently outperform traditional classical classifiers across all metrics.

Conversely, the alternative hypothesis (H_a) is empirically supported by the obtained results. The Xander algorithm, as a novel combination of hybrid quantum techniques within a dynamically weighted ensemble framework, demonstrated competitive and, in certain scenarios, superior performance compared to several classical and individual quantum rivals. This superiority was particularly evident in metric stability, false-positive reduction, and generalization capability-critical aspects in intrusion detection systems.

Furthermore, execution on real NISQ hardware confirmed the operational feasibility of the system, demonstrating that lightweight quantum kernels can be executed in practical scenarios with acceptable waiting times without compromising pipeline stability. Nonetheless, inherent limitations of current quantum hardware-particularly related to noise, T1/T2 times, and circuit depth-were identified. These constraints justified the adoption of reduced and optimized quantum architectures, as incorporated in the final version of the pipeline.

From a broader perspective, this work demonstrates that hybrid quantum computing should not be regarded as a direct replacement for classical methods, but rather as a strategic complement that, when properly integrated, can provide added value in complex tasks such as network intrusion detection. The proposed ensemble approach capitalizes on the strengths of each paradigm while mitigating their individual weaknesses.

Finally, this research contributes at both practical and academic levels. Practically, it delivers a reproducible experimental pilot executable on real quantum hardware and aligned with current technological constraints. Academically, it provides a critical and realistic analysis of the current state of Quantum Machine Learning applied to cybersecurity, establishing a solid foundation for future research and progressive deployments as quantum technology continues to mature.

REFERENCES

[1] Abdulrazzaq Altarraji, H., & Mokdad, V. (2024). Implementation of ZZ Feature Maps in QSVC for fraud detection.

[2] Abreu, R., et al. (2024). QML-IDS: A hybrid quantum-classical intrusion detection system using VQC, QSVM, and QCNN.

[3] Agnew, S., et al. (2024). Adaptability of conventional IDS in SDN-based smart networks.

[4] Ajdani, N. (2025). Combining GAN with deep neural networks for enhanced detection in NSL-KDD and CIC-IDS-2017.

[5] Akif, M., et al. (2025). Improving IoT-23 detection with voting classifiers (RF, XGBoost, KNN, AdaBoost).

[6] Alakhras, H., et al. (2025). Evolution of supervised models toward hybrid and ensemble deep learning for dynamic threats.

[7] Almseidin, M., et al. (2018). Importance of SVM and Random Forest in anomaly-based IDS.

[8] Alotaibi, A., & Rassam, M. A. (2023). Effectiveness and adaptation of IDS in expanding attack surfaces.

[9] Alluhaibi, R. (2024). Comparative study: QML vs. conventional ML for advanced threat detection.

[10] Biamonte, J., et al. (2017). Quantum machine learning: Physical principles and computational benefits.

[11] Camargo, J., et al. (2024). Quantum Support Vector Machines for overcoming NISQ barriers.

[12] Cerezo, M., et al. (2021). Variational quantum algorithms and hybrid models for near-term practical benefits.

[13] Danso, A., et al. (2022). Hybrid machine learning for handling heterogeneity in IoT network intrusion detection.

[14] Devadas, R., & T, S. (2025). Theoretical foundations of quantum learning algorithms.

[15] Dong, X., et al. (2022). Quantum Beetle Swarm Optimization (QBSA) for Extreme Learning Machines in IDS.

[16] Elsedimy, E., et al. (2024). Hybrid QSVM and Improved Grey Wolf Optimizer (IGWO) for IoT intrusion detection.

[17] Fan, L., et al. (2024). Hybrid quantum-classical learning for threat detection in critical infrastructures.

[18] Farhi, E., & Neven, H. (2018). Classification with Quantum Neural Networks on near-term processors.

[19] Ganapathi, S., & Shanmugapriya, D. (2019). Big Data Analytics for Intrusion Detection: An Overview.

[20] Gong, W., et al. (2022). Effectiveness of VQNN on NISQ hardware for CIC-IDS-2017 and NSL-KDD.

[21] Havlíček, V., et al. (2019). Supervised learning with quantum-enhanced feature spaces.

[22] Hozouri, M., et al. (2025). Categorization of IDS techniques based on detection methodology and deployment.

[23] Kadi, A., et al. (2025). Comparative analysis of quantum encoding strategies: Angle, Amplitude, and Pauli Feature Maps.

[24] Kadry, S., et al. (2023). Hybrid architecture merging QNN with ECC encryption for secure IDS.

[25] Kalinin, M., & Krundyshev, V. (2023). Scalability and accuracy of QSVM and QCNN in 10M record IoT traffic datasets.

[26] Kalyan, S., et al. (2025). GAN-based architectures for securing electric vehicle communications and IoT.

[27] Khan, A., et al. (2019). Overcoming limitations of traditional methods in complex attack detection.

[28] Khan, S., et al. (2024). Hybrid RNN-RF models for temporal feature extraction and classification.

[29] Kim, J., & Madhavi, L. (2024). Integrated IDS using QSVM and Quantum Clustering in NSL-KDD and UNSW-NB15.

[30] Li, D., et al. (2019). MAD-GAN: Multivariate Anomaly Detection for time series data via GANs.

[31] Maseer, Z., et al. (2023). Systematic review of the evolution from simple supervised models to hybrid deep learning.

[32] Metawei, N., et al. (2020). Variational algorithms as a viable route for short-term quantum benefits.

[33] Musa, S., et al. (2020). Performance of non-traditional methods in identifying complex cyber-attacks.

[34] Nazim, M., et al. (2025). Ensemble frameworks integrating GANs and deep networks for IoT and cloud security.

[35] Nicesio, R., et al. (2023). Training time reduction and accuracy improvement in hybrid VQC and QSVM models.

[36] Noor, S., et al. (2018). Foundations of hybrid quantum-classical analytics for cybersecurity.

[37] Noor, S., et al. (2025). Systematic study on transfer learning and hybrid systems for 5G and cloud IDS.

[38] Oda, T., et al. (2023). Noise and decoherence challenges in the NISQ era: impact on qubit stability.

[39] Odeh, A., & Abu Taleb, K. (2023). Fusion of CNN, LSTM, and GRU for high-performance intrusion detection.

[40] Orazi, L., et al. (2024). Classification in high-dimensional Hilbert spaces using quantum feature maps.

[41] Payares, A., & Martínez-Santos, J. (2021). Comparison between QSVM and QNN for DDoS attack identification.

[42] Pinto, A., et al. (2023). False positive rates and scalability challenges in Machine Learning-based IDS.

[43] Premalatha, S. (2025). Ensemble models as a robust strategy for error reduction in heterogeneous environments.

[44] Rawat, R., et al. (2022). Performance degradation of traditional models against unknown attacks in 5G/IoT contexts.

[45] Rehman, U., et al. (2025). Recent trends in automated and complex hybrid models for real-world IDS.

[46] Sarker, I. H. (2021). Traditional vs. modern algorithms in anomaly-based network security.

[47] Schuld, M., et al. (2015). Introduction to quantum machine learning and its physical realization.

[48] Shen, Y., et al. (2024). Quantum-inspired computing (QIC) and voting systems for simulated hybrid models.

[49] Solar, M., et al. (2024). Characterization of NISQ devices: limited qubits and noise susceptibility.

[50] Talukder, M., et al. (2023). High-precision detection in KDDCup'99 and CIC-MalMem-2022 using SMOTE and XGBoost.

[51] Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset," in Proc. ICISSP, 2018.