Intrusion Detection Systems using Quantum-Inspired Density Matrix Encodings

Larry Huynh*, Jin B. Hong*, Ajmal Mian*, Hajime Suzuki†, Seyit Camtepe†
*University of Western Australia, Australia
Emails: {larry.huynh, jin.hong, ajmal.mian}@uwa.edu.au
†Commonwealth Scientific and Industrial Research Organisation (CSIRO)'s Data61, Australia
Emails: {hajime.suzuki, seyit.camtepe}@data61.csiro.au

***Abstract*—We investigate the application of quantum-inspired density matrix encodings to intrusion detection systems (IDS) in both supervised and unsupervised machine learning contexts. By leveraging the density matrix formulation from quantum mechanics, we introduce a novel perspective on generating aggregate flow data for network traffic classification. The proposed encoding method is evaluated on a benchmark dataset, demonstrating its feasibility and effectiveness. Experimental results showcase its strong predictive capabilities, achieving over 99.89% accuracy, precision, recall, and F1 scores, highlighting the competitive performance of our approach when compared with common encoding methods. Our findings promote further exploration of density matrices in the domain of intrusion detection systems, offering a promising avenue for improving the efficiency and accuracy of network traffic classification tasks.**

I. Introduction

Intrusion detection systems (IDS) are crucial for protecting computer networks from malicious activities and unauthorised access. Over the years, researchers have approached the problem of intrusion detection from multiple angles, using techniques such as rule-based systems, statistical methods, and machine learning. The latter has been widely applied to both signature-based and anomaly-based IDS, leveraging the ability to learn complex patterns and relationships in network traffic data [1], [2], [3]. Despite these advancements, capturing the complex relationships and correlations between traffic features remains a challenge. In light of this, one possible approach could borrow ideas from the physics domain, by way of quantum-inspired machine learning techniques. Our work thus contributes to the field by exploring the application of quantum-inspired density matrix encoding methods to IDS. Density matrices, originating from quantum mechanics, have recently shown promise for representing probability distributions in machine learning. By combining linear algebra and probability theory, density matrices provide a framework for modelling uncertainty and complex relationships.

Inspired by the recent success of density matrices when applied to natural language processing (NLP) tasks [4], we adapt this approach to the IDS domain. We hypothesise that density matrix encodings can be similarly applied to capture the distribution and correlations of traffic features for intrusion detection. We aim to learn a rich representation that encapsulates both normal and anomalous traffic patterns. By operating at the packet level, density matrix encodings can potentially capture higher-level patterns and relationships that are more indicative of intrusions.

The main contributions of this paper are as follows:

- We propose a novel density matrix encoding scheme for representing network traffic features in the context of intrusion detection.
- We test this representation under both supervised and unsupervised machine learning settings, enabling anomaly-based intrusion detection.
- We assess the performance of our method on a given dataset and compare it against common baseline encoding methods.
- We provide insights into the learned representations and discuss the potential of quantum-inspired methods for cybersecurity applications.

The remainder of this paper is organized as follows. Section II reviews related work on machine learning-based IDS and density matrix encodings. Section III introduces the density matrix encoding scheme, while Section IV describes our proposed methodology. Section V presents our experimental setup and results. Finally, sections VI and VII conclude the paper and discuss future research directions.

II. Related Work

Machine learning-based IDS has been extensively studied in both supervised and unsupervised contexts. Supervised models such as Support Vector Machines (SVMs), Random Forests (RFs), and deep learning architectures have achieved high accuracies. Ahmad et al. [5] used SVMs, RFs, and neural networks with feature engineering, with their RF model performing best at 97.54% accuracy over flow and TCP clusters. Talaei Khoei and Kaabouch [6] compared several models, noting the strong performance of AlexNet and Variational Autoencoder deep learning approaches. Awajan [7] proposed a modular deep neural network architecture for IoT malicious traffic detection, achieving a 93.21% detection rate. Unsupervised learning is commonly applied for anomaly-based IDS, with autoencoders being prominent due to their ability to learn normal patterns and flag deviations as anomalies. Mirsky et al. [8] used an ensemble of autoencoders for near real-time IoT traffic anomaly detection. Yang et al. [9] similarly leveraged deep autoencoders for known and zero-day attack detection in real-time. Song et al. [10] applied autoencoders over various

2325-6664/24/$31.00 ©2024 IEEE
DOI 10.1109/DSN-W60302.2024.00019

datasets, finding latent size significantly impacts performance. Basati and Faghih [11] introduced APAE, using dilated convolutional neural network (CNN) filters to prepare 2D feature vectors for an asymmetric autoencoder, outperforming state-of-the-art methods. Other techniques like PCA and K-Means have also been explored [6].

The density matrix representation of quantum systems has shown utility in NLP tasks by capturing probabilistic mixtures of semantic bases [4]. The density matrix can either encapsulate the entire dataset within a single matrix [12] or represent each data point as an individual density matrix [13], [14], [15]. Density matrix encoding has been compared with traditional encoding methods, such as n-gram and Doc2vector, and has shown comparable or improved performance at the cost of increased training time [13]. While its application to intrusion detection remains unexplored, the ability of density matrices to represent probabilistic mixtures of basis states and capture complex relationships suggests potential benefits for encoding network traffic data [15], [16], [11]. By treating traffic features or patterns as basis vectors, a density matrix could encapsulate the distribution and correlations of these features across the dataset. Further, several works have noted the enhanced predictive performance of flow-based IDS versus packet-based IDS [17], [18]. Density matrices may offer an alternative method of constructing flows; this novel perspective on traffic data representation may offer new insights and improved performance for intrusion detection tasks.

III. Density Matrices for Encoding:

In quantum mechanics, density matrices are used to represent both pure and mixed states, the latter of which are statistical combinations of various quantum states typically arising from partial or incomplete information about a system. The density matrix ρ for a quantum system is defined as an operator acting on the system's Hilbert space. For a system in a pure state described by a normalised state vector $|\psi\rangle$, the corresponding density matrix is given by the outer product $|\psi\rangle\langle\psi|$. For a mixed state composed of possible states $|\psi_i\rangle$ each occurring with probability p_i, the density matrix is expressed as a sum of the individual state's density matrices, weighted by their respective probabilities: $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$.

From a classical viewpoint, the density matrix can be conceptualised as a linear algebraic construct where classical data is represented as a sum of weighted outer products. Specifically, if we consider a classical data point $\mathbf{x}$ in a dataset X to be analogous to a mixed quantum state [1], its corresponding mixed density matrix representation ρ_i can be formulated as:

$$\rho = \sum_i p_i |\phi_i\rangle\langle\phi_i| . \tag{1}$$

[1]In this context, the analogy of a classical data point $\mathbf{x}$ to a mixed quantum state does not imply the encoding of data into actual quantum states. Instead, it represents a classical interpretation of quantum constructs, using the mathematical framework of quantum mechanics as a tool for classical data analysis. This approach utilises quantum-inspired models for processing classical data, avoiding the complexities of physical quantum computation.

$|\phi_i\rangle$ can be achieved from $\mathbf{x}$ via the unit normalisation of $\mathbf{x}$: $|\phi_i\rangle = \mathbf{x}/\|\mathbf{x}\|$. In the quantum setting, this process is referred to as amplitude encoding. p_i are the chosen weights associated with $\mathbf{x}$. The choice of basis states and associated weights may be dependent on the data domain. In natural language processing, each density matrix ρ may model a sentence or document, which is a composite of basis vectors ϕ_i, where each basis vector represents a word [13], [14], [15]. In the case of network traffic data used in intrusion detection systems (IDS), which often consists of sequences of packets or flows, where the context and order of these packets are crucial for accurate analysis. Thus, we may similarly consider each ρ as a *flow*, constituted by a sequence of packets, with each packet considered as a basis state.

For a classical data point $\mathbf{x}$ of m features, its output density matrix ρ_i is of size m^2; a quadratic cost is incurred in utilising this data for machine learning application, with all other variables being fixed. In the encoding of sequences of data points as density matrices, the number of density matrices will be less than the initial number of data points. This reduction is due to the aggregation of information into a more compact form, where each sequence of data points is represented by a single density matrix. Thus the cost is scaled by a factor of s/N, where s is the number of sequences identified and N is the number of data points within the dataset. The formulation of a dataset of density matrices is thus described as:

$$\rho(X) = \sum_s \rho_n = \sum_s \left(\sum_i p_i |\phi_i\rangle\langle\phi_i| \right)_s . \tag{2}$$

IV. Methodology

We describe the methodology for applying density matrix encodings to learn from network traffic data, as illustrated in Figure 1. First, raw packet capture (pcap) files are segmented into flows. Then, a density matrix encoding is applied to map the flow data into the quantum-inspired representation. Using this encoded data, we conduct two experiments: supervised learning with a neural network classifier to predict traffic labels (e.g. normal vs attack), and unsupervised learning with an autoencoder to detect anomalies based on reconstruction errors of the compressed representation. The following sections detail the dataset, encoding process, model architectures, and evaluation metrics.

A. Dataset

To assess our proposed method, we use a simple network traffic dataset — the Mirai BotNet Dataset [8]. This dataset was collected from a Wi-Fi network comprised of 9 IoT devices, including a thermostat, baby monitor, webcam, two different doorbells, and four security cameras, along with three PCs. One of the security cameras in this setup was intentionally infected with a real sample of the Mirai botnet malware. The dataset itself consists of network traffic data captured in pcap format, featuring detailed packet information such as headers and payload. Each packet has an associated 'benign' or 'malicious' label, with the malicious packets representing

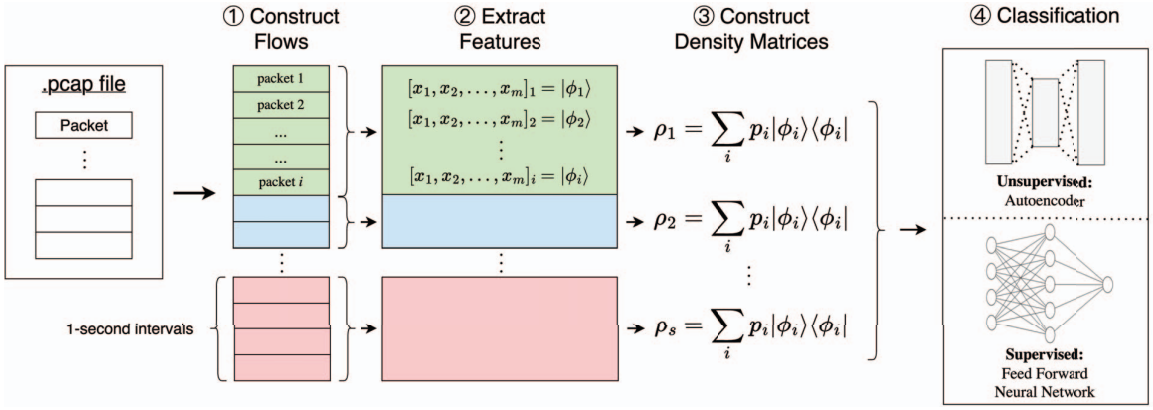

Fig. 1. Overview of the Density Matrix Encoding-based Machine Learning Pipeline.

network behaviour indicative of the Mirai botnet's activity. For preprocessing, we utilise Wireshark to extract features from the pcap file, including Ethernet and IP details (source and destination addresses, protocol specifications), TCP and UDP specifics (ports, flags, sequence, and window size), as well as ICMP and ARP protocol information.

B. Density Matrix Encoding

For the density matrix encoding, packets within 1-second intervals are aggregated into 7137 flows, with each packet considered as a basis element for the flow's density matrix. The average number of packets per flow is approximately 107. Following Equations 1 and 2, we calculate the density matrix, where the probabilities p_i are based on the packet's protocol and its frequency within the pcap file. This choice is made due to the protocol's influence on network traffic patterns. However, practitioners can explore alternative influential factors that might offer additional insights, allowing for flexibility in defining the significance of each basis.

C. Detection Models

Supervised Learning: For supervised learning, our model employs a simple feedforward neural network (FNN). FNNs are characterised by their uni-directional flow of information, consisting of an input layer, one or more hidden layers, and an output layer. Each layer comprises of neurons that process the data as it passes through the network. The model is trained using a supervised learning approach, learning to map inputs to the correct outputs based on a dataset of labelled examples. Our specific FNN model is structured with three layers: an input layer, a hidden layer, and an output layer. The hidden layer contains 64 nodes, and the output layer consists of a single node. The number of input nodes is determined by the length of the input feature dimensions.

Unsupervised Learning: Online IDS faces the challenge of modelling data without complete knowledge of the labels, as there may be unknown threats present that signature-based approaches fail to detect. Various models have been employed for this task in the literature; in our work, we utilise an autoencoder (AE) architecture. Autoencoders are unsupervised neural networks that learn efficient data representations by reconstructing the input at the output layer. The network is forced to compress the input into a lower-dimensional representation at the bottleneck hidden layer, capturing the most salient features of the data. By reconstructing normal traffic patterns with minimal error while struggling to reproduce anomalous samples, the AE is able to detect deviations indicative of potential intrusions. Our AE model consists of two hidden layers, with the latent dimension set to 64. The input and output dimensions match the size of the input feature vectors. Both models are trained for 100 epochs using the Adam optimiser with a batch size of 32, employing early stopping to prevent over-fitting.

D. Evaluation Metrics

For the supervised learning experiment, we use 5-fold cross-validation and evaluate the trained neural network classifier using accuracy, precision, recall, F1 score, and the receiver operating characteristic (ROC) curve. These metrics assess the model's ability to correctly identify normal and anomalous network traffic. In the unsupervised setting, we reserve 10% of the benign data for testing and train the autoencoder on the remaining 90% to learn a compressed representation of normal network behaviour. We evaluate the autoencoder's performance on a test set containing both reserved normal data and anomalous traffic samples. By thresholding the reconstruction error, the autoencoder can discriminate between normal and malicious traffic. We assess its performance using the same metrics as in the supervised case.

E. Baseline Encoding Methods

We additionally prepare three different baselines for comparison with the density matrix method: using the raw packet data features, using a flow aggregation software, CICFlowMeter [19], to generate flows, and using statistical aggregates as features.

- *Raw Data Features*: We extract a comprehensive set of features from raw pcap files using Wireshark, detailing

packet sizes, timing, and header information. Specifically, our extracted features include frame length, IP networking features (protocol, time-to-live, flags, total length), TCP and UDP protocol features (source and destination ports, TCP flags, sequence number, segment length, window size), and ICMP and ARP protocol features (type, code). Categorical fields such as IP protocol, TCP flags, etc. are one-hot encoded, giving a total feature length of 32. These extracted features are used as-is for training our machine learning models, resulting in a dataset of 764,137 packets.

- *CICFlowMeter Flows*: We use the CICFlowMeter flow aggregation tool to generate bidirectional flow records from our raw pcap files. CICFlowMeter extracts 81 traffic features for each flow, covering packet length statistics, inter-arrival times, flags, header information, and more. Key features include flow duration, total forward/backward packets and bytes, min/max/mean/std of packet lengths and inter-arrival times in each direction, TCP flags, header lengths, and various derived metrics like bytes per bulk, packets per second, and download/upload ratio. We run CICFlowMeter with the default feature set and flow timeout, generating one flow record per bidirectional connection in our pcaps, resulting in a dataset of 36,133 flows.
- *Statistical Aggregation*: Using our raw data features, we apply statistical aggregation to describe sequences of packets as network flows. We consider a flow as a timed sequences of packets passing through a particular point in a network [20]; for our analysis, we define each 1-second interval in our pcap dataset as a flow, encompassing a sequence of packets within that time-frame. We then compute summary statistics for each raw data feature across these packets, using the following metrics: the mean, variance, median, minimum, and maximum, and obtain 7137 flows. Each flow thus contains $32 \times 5 = 160$ features.

F. Experimental Setup

All experiments were conducted on a MacBook Pro M2, equipped with a 10-core CPU, and 16GB of unified memory. For software, we used Python 3.12, along with popular data science libraries such as Pandas and NumPy. Machine learning tasks were performed using Scikit-learn and Tensorflow Keras.

V. Results

A. Performance

We present the results of our two experiments below. Table I contains the results for the supervised experiments, while Table II contains those for the unsupervised experiments. Since we perform classification at the aggregate level for the density matrix encoding method "DM", the statistics-based method "STATS" and the flow-based method "FLOW" methods, which map multiple network packets to a single classification, we map the aggregate classification results back to the individual packets to report performance metrics at the packet level. The bold values indicate the best performance for the given metric. In Table I, we see that the density matrix encoding method DM shows superior performance compared to all the other methods, with near-perfect scores across all metrics. We see that the statistics-based method STATS exhibits competitive performance with DM, closely following it in terms of accuracy, recall, precision, and F1 score. This is followed by the raw packet data method "RAW", which does well, showing a high accuracy and F1 score. Finally, the flow-based method FLOW performs comparatively poorly, with low recall and F1 scores. This implies that FLOW struggles to effectively identify positive cases and has a high rate of false positives and false negatives. Conversely, the RAW and STATS encoding methods perform well in identifying malicious packets while minimising false positives, while the DM method demonstrates an improved ability to distinguish between benign and malicious packet representations, resulting in better overall performance.

TABLE I
PERFORMANCE METRICS FOR NEURAL NETWORK MODELS (SUPERVISED)

Encoding	Accuracy	Precision	Recall	F1 Score
RAW	0.87066	0.90236	0.94893	0.92506
FLOW	0.62668	0.69712	0.05216	0.09705
STATS	0.98810	0.99801	0.96905	0.98332
DM	**0.99986**	**0.99963**	**1.00000**	**0.99982**

TABLE II
PERFORMANCE METRICS FOR AUTOENCODER MODELS (UNSUPERVISED)

Encoding	Accuracy	Precision	Recall	F1 Score
RAW	0.89854	0.98949	0.88877	0.93643
FLOW	0.61556	0.49154	0.07550	0.13089
STATS	0.99643	**1.00000**	0.99575	0.99787
DM	**0.99907**	0.99992	**0.99897**	**0.99945**

The unsupervised experiments (Table II) also showed similar results; the DM method outperformed all other methods in accuracy, recall, and F1 score, while STATS achieved a slightly higher precision (0.99992 vs. 1.0). These findings suggest that density matrix encodings can perform competitively when compared to other encoding methods for IDS.

The ROC curve plots (Figures 2 and 3) confirm the strong performance findings from the tables. In both supervised and unsupervised settings, the DM method demonstrates superior performance, achieving a perfect or near-perfect area-under-the-curve (AUC), which indicates its ability to effectively separate benign and malicious packet representations. The STATS method also exhibits excellent performance, closely following the DM method. Both DM and STATS methods show a sharp increase at the beginning of their ROC curves, suggesting they can achieve high true positive rates while maintaining low false positive rates. This is a highly desirable characteristic for an intrusion detection system. The elbow shape of their curves indicates robust class separation and a scarcity of intermediate points.

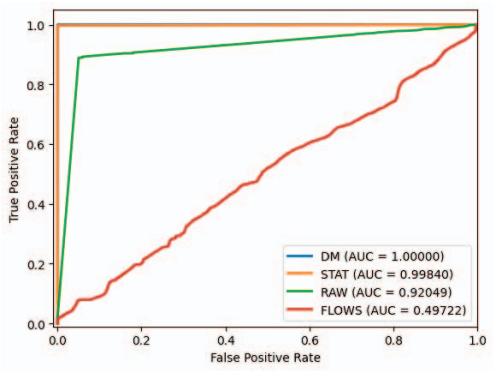

Fig. 2. ROC Curves for Different Data Encoding Methods (AE)

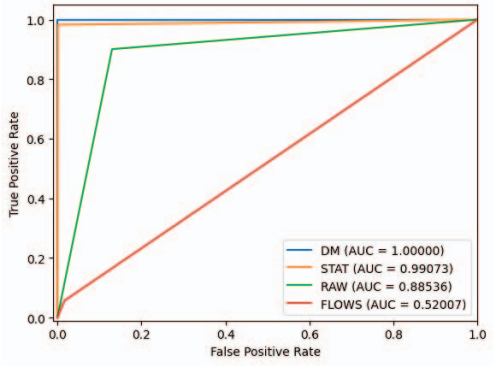

Fig. 3. ROC Curves for Different Data Encoding Methods (NN)

B. Comparing Encoding, Inference and Training Times

TABLE III
COMPARISON OF ENCODING, INFERENCE, AND TRAINING TIMES FOR SUPERVISED AND UNSUPERVISED MODELS.

	Encoding Time (s)	Supervised Times (s)		Unsupervised Times (s)	
		Inference	Training	Inference	Training
Raw	-	2.30e−4	533.62	2.30e−4	140.10
Flow	3.54e−4	5.77e−4	149.85	5.77e−4	25.92
Stats	1.55e−2	1.58e−2	31.09	1.58e−2	8.59
DM	2.48e−2	2.50e−2	17.17	2.50e−2	9.96

Table III compares the encoding, inference, and training times for supervised and unsupervised models across different encoding methods. The encoding time represents the average time to encode a single instance, while the inference time is the sum of the encoding and model's inference times for one instance, across all test instances. Training times are averaged over 5 trials. The RAW encoding method requires no additional encoding time but has the longest training times among all methods. This is expected, as the other encoding methods reduce the dimensionality of the data, resulting in fewer data points for the model to process. FLOW encoding has a low encoding time of 3.54×10^{-4} seconds and significantly reduces training times compared to RAW. The STATS and DM methods have higher encoding times, and thus inference times, of 1.55×10^{-2} and 2.48×10^{-2} seconds, respectively. The DM encoding is relatively time-intensive, incurring a time complexity of $O(nm^2)$ as it requires encoding and summing n packets, within a given time window, of m features into matrices of m^2 dimensions. The STATS method incurs a time complexity of $O(nm)$, stemming from the aggregation operations over the given flow. Despite having the highest encoding and inference times, the DM method offers slightly improved performance compared to the other methods.

C. Effect of Changing Probabilities p_i

TABLE IV
PERFORMANCE METRICS FOR DIFFERENT PROBABILITY METHODS

Prob. Method	Accuracy	Precision	Recall	F1 Score
IP Protocol	0.99907	0.99992	0.99897	0.99945
Uniform	0.99896	**1.00000**	0.99876	0.99938
Random	**0.99916**	0.99996	**0.99905**	**0.99950**

Table IV compares the performance of using IP protocol as weights for the probabilities, against using uniform or random probabilities; the probabilities are assigned for each flow, such that they sum to one for the flow, for all methods. For the random probabilities, we perform the experiment 5 separate times using 5 different random seeds. We then compute and report the average performance metrics in the table. We observe that using uniform or even random probabilities for the protocol can lead to enhanced performance compared to using the protocol information directly. This suggests that the protocol information may not successfully encode valuable information for optimally informing the probabilities. It may be sufficient to simply use uniformly assigned probability values, or additional exploration may be required to determine the best method of assigning probabilities to maximise performance.

D. Effect of Changing Time Intervals

TABLE V
CLASSIFICATION METRICS FOR DIFFERENT TIME INTERVALS

Intervals (s)	Acc.	Prec.	Rec.	F1	# Flows	Ave. Pkts/Int.
Unsupervised (AE)						
0.1	0.832	0.739	0.940	0.828	51748	14.8
0.5	0.831	0.781	0.763	0.772	14253	53.6
1	0.999	0.999	0.998	0.999	7137	107.1
2	0.990	1.000	0.974	0.987	3596	214.1
3	0.990	1.000	0.974	0.987	2230	342.7
4	0.999	1.000	0.997	0.999	1693	451.4
5	0.999	1.000	0.998	0.999	1428	535.1
10	0.998	1.000	0.996	0.998	714	1070.2
Supervised (NN)						
0.1	0.871	0.770	1.000	0.870	51748	14.8
0.5	0.995	0.988	1.000	0.994	14253	53.6
1	1.000	1.000	1.000	1.000	7137	107.1
2	1.000	1.000	1.000	1.000	3596	214.1
3	1.000	1.000	1.000	1.000	2230	342.7
4	1.000	1.000	1.000	1.000	1693	451.4
5	1.000	1.000	1.000	1.000	1428	535.1
10	0.997	1.000	0.993	0.996	714	1070.2

The effect of changing the time (flow) interval, which determines the number of packets in each density matrix, is

examined in both unsupervised (AE) and supervised (NN) settings. In the AE setting, a performance peak is observed at the 1-second flow interval, with accuracy, precision, recall, and F1 score all reaching 0.998 or higher. The metrics slightly dip at 2 and 3-second intervals before recovering to near-perfect levels for intervals of 4 seconds and above. As the flow interval increases, the number of flows decreases substantially, from 51,748 at 0.1s to only 714 at 10s, while the average packets per flow increases proportionally. The high performance at longer intervals with much smaller dataset sizes could potentially indicate over-fitting, requiring further analysis. In the NN setting, a performance peak is observed at a flow window of 1, which is maintained throughout flow windows 2-5, with most metrics consistently achieving 1.000. A small dip is observed at a flow window of 10. This may suggest a relationship between performance and the amount of data compressed into each density matrix, affecting the model's ability to accurately classify or predict outcomes. However, more exploration is needed to confirm this trend.

VI. Discussion

A. Performance of Density Matrix Encodings

The results presented in Section V-A demonstrate the strong performance of the density matrix encoding method for network intrusion detection on the Mirai BotNet dataset. In both supervised and unsupervised settings, this representation outperformed other methods including statistics-based features, raw packet data, and flow-based features. The density matrix approach achieved exceptional accuracy, precision, recall, and F1 score in the supervised experiments, and obtained the highest scores on all metrics except precision in the unsupervised tests. The corresponding ROC curves further highlight the superiority of this encoding, showing it can well separate benign and malicious activity at various thresholds.

B. More Robust Consideration of Flow Windows

The current method determines the flow interval statically, without considering the dynamic nature of network traffic. This static approach may be vulnerable to attacks that spread network activity over a long time frame, as the attack packets would have less influence on the density matrix. To improve robustness against delayed attacks, dynamically deciding the flow window based on traffic patterns, or using communication channels as flows instead of static time windows could be considered. Exploring these methods can help make the system more resilient to adversaries attempting to exploit knowledge of the static flow windows.

C. Possibilities for Use in Real-Time Detection

The suitability of density matrix encodings for real-time detection depends on the speed at which the inference and encoding can be performed. This encoding process can add greater computational complexity compared to other methods, which may subsequently lead to an extra cost in inference time, as discussed in Section V-B. The time-intensive nature of density matrix encoding might pose challenges for real-time detection, which can be highly time-sensitive. Further research and optimisation efforts may be necessary to mitigate the computational overhead and make density matrix encodings more feasible for real-time applications.

D. Other Future Directions

While these findings are promising, further exploration of the density matrix methodology in different contexts is necessary. The Mirai BotNet dataset used here is relatively small and simple, containing only a single attack type; this simplicity is reflected in Table V, where all results show very strong performance. Experimenting on more complex datasets with diverse attack patterns could provide deeper insights into the strengths and limitations of this approach and enable its extension to problems like multi-class classification. Due to space constraints, such evaluations are not present in this work; however, we plan to explore these aspects in our future research to better understand the effectiveness and generalisability of the proposed technique.

VII. Conclusion

We investigate the application of density matrix encodings for network intrusion detection. This quantum-inspired representation aims to capture the complex relationships and dependencies present in network traffic data. We evaluated the effectiveness of this approach on the Mirai BotNet dataset in both supervised and unsupervised settings, comparing against other commonly used network traffic representations. The results obtained demonstrate the strong performance of the density matrix encoding for this task. In the supervised experiments, it achieved strong accuracy, precision, recall, and F1 scores, outperforming all other methods tested. For unsupervised intrusion detection, the density matrix approach obtained an accuracy, recall and F1 score (in percentages) of 99.91%, 99.90% and 99.95% respectively, staying competitive with the next best methods. These findings suggest that the density matrix representation is able to effectively capture the relevant patterns and distinguish between benign and malicious network activity. While this exploration is in its nascent stages, the outcomes of this research highlight the potential of quantum-inspired machine learning techniques, specifically the density matrix encoding, for enhancing network security. By providing a more expressive and powerful representation of traffic data, this methodology could enable the development of more accurate and robust intrusion detection systems.

Acknowledgment

This work was supported by Commonwealth Scientific and Industrial Research Organisation (CSIRO)'s Quantum Technologies Future Science Platform. Ajmal Mian is the recipient of an Australian Research Council Future Fellowship Award (project number FT210100268) funded by the Australian Government.

REFERENCES

[1] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, 2019.

[2] A. Thakkar and R. Lohiya, "A review on machine learning and deep learning perspectives of ids for iot: recent updates, security issues, and challenges," *Archives of Computational Methods in Engineering*, vol. 28, no. 4, pp. 3211–3243, 2021.

[3] K. He, D. D. Kim, and M. R. Asghar, "Adversarial machine learning for network intrusion detection systems: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 538–566, 2023.

[4] Y. Liu, Q. Li, B. Wang, Y. Zhang, and D. Song, "A survey of quantum-cognitively inspired sentiment analysis models," *ACM Computing Surveys*, 2023.

[5] M. Ahmad, Q. Riaz, M. Zeeshan, H. Tahir, S. A. Haider, and M. S. Khan, "Intrusion detection in internet of things using supervised machine learning based on application and transport layer features using unsw-nb15 data-set," *EURASIP Journal on Wireless Communications and Networking*, vol. 2021, pp. 1–23, 2021.

[6] T. Talaei Khoei and N. Kaabouch, "A comparative analysis of supervised and unsupervised models for detecting attacks on the intrusion detection systems," *Information*, vol. 14, no. 2, p. 103, 2023.

[7] A. Awajan, "A novel deep learning-based intrusion detection system for iot networks," *Computers*, vol. 12, no. 2, p. 34, 2023.

[8] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," *arXiv preprint arXiv:1802.09089*, 2018.

[9] L. Yang, Y. Song, S. Gao, A. Hu, and B. Xiao, "Griffin: Real-time network intrusion detection system via ensemble of autoencoder in sdn," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2269–2281, 2022.

[10] Y. Song, S. Hyun, and Y.-G. Cheong, "Analysis of autoencoders for network intrusion detection," *Sensors*, vol. 21, no. 13, p. 4294, 2021.

[11] A. Basati and M. M. Faghih, "Apae: an iot intrusion detection system using asymmetric parallel auto-encoder," *Neural Computing and Applications*, vol. 35, no. 7, pp. 4813–4833, 2023.

[12] F. A. González, A. Gallego, S. Toledo-Cortés, and V. Vargas-Calderón, "Learning with density matrices and random features," *Quantum Machine Intelligence*, vol. 4, no. 2, p. 23, 2022.

[13] Y. Zhang, D. Song, P. Zhang, X. Li, and P. Wang, "A quantum-inspired sentiment representation model for twitter sentiment analysis," *Applied Intelligence*, vol. 49, pp. 3093–3108, 2019.

[14] P. Guo, J. Zhang, Y. Hou, X. Gong, P. Wang, and Y. Zhang, "Quantum-inspired dmatt-bigru for conversational sentiment analysis," in *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2019, pp. 1602–1606.

[15] Y. Zhang, D. Song, X. Li, P. Zhang, P. Wang, L. Rong, G. Yu, and B. Wang, "A quantum-like multimodal network framework for modeling interaction dynamics in multiparty conversational sentiment analysis," *Information Fusion*, vol. 62, pp. 14–31, 2020.

[16] Q. Li, D. Gkoumas, C. Lioma, and M. Melucci, "Quantum-inspired multimodal fusion for video sentiment analysis," *Information Fusion*, vol. 65, pp. 58–71, 2021.

[17] H. Hindy, E. Bayne, M. Bures, R. Atkinson, C. Tachtatzis, and X. Bellekens, "Machine learning based iot intrusion detection system: An mqtt case study (mqtt-iot-ids2020 dataset)," in *International Networking Conference*. Springer, 2020, pp. 73–84.

[18] S. Samarakoon, Y. Siriwardhana, P. Porambage, M. Liyanage, S.-Y. Chang, J. Kim, J. Kim, and M. Ylianttila, "5g-nidd: A comprehensive network intrusion detection dataset generated over 5g wireless network," *arXiv preprint arXiv:2212.01298*, 2022.

[19] A. H. Lashkari, G. D. Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features," in *International Conference on Information Systems Security and Privacy*, vol. 2. SciTePress, 2017, pp. 253–262.

[20] M. F. Umer, M. Sher, and Y. Bi, "Flow-based intrusion detection: Techniques and challenges," *Computers & Security*, vol. 70, pp. 238–254, 2017.