

# Towards Quantum Machine Learning in Ransomware Detection

Francesco Mercaldo<sup>4,2</sup><sup>a</sup>, Giovanni Ciaramella<sup>1,2</sup><sup>b</sup>,  
Fabio Martinelli<sup>3</sup><sup>c</sup> and Antonella Santone<sup>4</sup><sup>d</sup>

<sup>1</sup>IMT School for Advanced Studies Lucca, Lucca, Italy

<sup>2</sup>Institute for Informatics and Telematics, National Research Council of Italy (CNR), Pisa, Italy

<sup>3</sup>Institute for High Performance Computing and Networking, National Research Council of Italy (CNR), Rende, Italy

<sup>4</sup>University of Molise, Campobasso, Italy


**Keywords:** Ransomware, Malware, Machine Learning, Quantum Computing, Security.


**Abstract:** Ransomware represent one of the most aggressive malware, due to their capability to prevent access to data and, as a consequence, totally paralyze the activity of any organization, such as companies, but also hospitals or banks. Considering the inadequacy of the signature-based approach, mainly exploited by free and commercial current antimalware, researchers are proposing new ransomware detection techniques based on deep learning. Recently, with the introduction of quantum computing, there is the possibility to introduce quantum principles into machine learning. In this paper, we propose an approach for ransomware detection through a quantum machine learning model aimed to analyse images obtained from the application opcodes. In particular, a hybrid model is proposed, composed of quantum and convolutional layers to discern between ransomware, generic malware, and trusted applications. To demonstrate that quantum machine learning is promising in ransomware detection, a real-world dataset composed by 15,000 applications is evaluated, by showing that the proposed hybrid quantum model obtains promising performances if compared to (fully) convolutional models (i.e., Alex\_Net, MobileNet, and a convolutional model developed by authors).


## 1 INTRODUCTION


Ransomware is malicious software employed by threat actors to penetrate a network. This process entails encrypting discovered data and withholding the decryption key until a specified ransom is paid to the hackers. With the advent of the digital age, ransomware groups find it increasingly advantageous to operate covertly on the internet and typically request ransom payments in Bitcoin to evade detection. In 2021, over one-third of organizations worldwide experienced attempted ransomware attacks. There were 623.3 million ransomware attacks globally in 2021, marking a 105% increase compared to 2020 figures. This surge can be partly attributed to businesses encountering difficulties adapting their networks and

supply chains for remote and hybrid work<sup>1</sup>. It is important to note that not all of these attacks were successful; instead, they were attempted breaches. However, in 2022, the volume of ransomware attacks decreased by 23%, indicating that increased government scrutiny and growing awareness of the associated risks are starting to have an impact<sup>2</sup>. Nevertheless, attack methods are evolving. In the past, "traditional" ransomware techniques involved encrypting target data and demanding a fee for the decryption key. Now, cybercriminals are resorting to "double-extortion" schemes, threatening to release or sell the data if the ransom is not paid. Other methods of extortion include Denial of Service attacks and harassment through email or phone communications. According to Fortinet security experts, as cybercriminals utilize data infiltrations and the threat of data leaks, attacks continue to surge, exerting more significant pressure

<sup>a</sup> <https://orcid.org/0000-0002-9425-1657>

<sup>b</sup> <https://orcid.org/0009-0002-9512-0621>

<sup>c</sup> <https://orcid.org/0000-0002-6721-9395>

<sup>d</sup> <https://orcid.org/0000-0002-2634-4456>

<sup>1</sup><https://www.sangfor.com/blog/cybersecurity/ransomware-attacks-2022-overview>

<sup>2</sup><https://aag-it.com/the-latest-ransomware-statistics/>

on companies to pay the ransom, and this trend will continue in 2023<sup>3</sup>. Even if a company can restore its data from backups, the leaked data from organizations that refuse to comply with ransom demands may surface on database websites operated by threat actors. Cryptocurrency is commonly used for ransomware payments due to its anonymity. According to Chainalysis, over \$602 million in ransom payments for attacks were made using cryptocurrency. Cybersecurity Ventures estimates that global ransomware damage will witness a 30% year-over-year growth over the next decade. By 2031, the damages are projected to exceed \$265 billion annually, with a new attack occurring approximately every two seconds. Traditional malware detection struggles with new threats as it relies on signature-based methods. To address this, researchers are exploring machine learning and deep learning for improved detection. With the rise of quantum computing, there is growing interest in applying quantum machine learning to enhance malware detection capabilities. A quantum computer leverages quantum mechanical phenomena to perform computations. Unlike classical computers, which use bits, quantum computers use qubits that can exist in superposition, allowing for parallel processing. This enables them to solve certain problems exponentially faster than classical computers, with potential applications in encryption breaking and complex simulations. However, quantum computing is still largely experimental. Quantum algorithms are designed to exploit quantum properties like wave interference to achieve efficient computations. Recently, researchers introduced the concept of quantum computing in the malware detection fields, using full quantum and hybrid approaches (Ciamarella et al., 2022) (Mercaldo et al., 2022) (Ciaramella et al., 2023) (He et al., 2024).

Starting from these considerations, we propose a method for ransomware detection by exploiting quantum machine learning in this paper. In detail, we propose a model aimed at discriminating between ransomware, (generic) malware, and legitimate applications by exploiting a network composed of a layer to simulate quantum convolutions by considering transformations in circuits to simulate a quantum computer's behavior.

The remaining of the paper proceeds as follows: the next Section provides the state-of-the-art literature related to malware detection, while the proposed quantum machine learning model for ransomware detection is described in Section 3; experimental results obtained by evaluating a real-world dataset composed of ransomware, malware, and le-

gitimate applications are presented in Section 4 and, finally, conclusion and future research lines are drawn in the last Section.

## 2 RELATED WORK

Authors in (Ferrante et al., 2017) presented a method able to detect ransomware using a static and dynamic approach. The first takes into account the frequency of opcodes, while the second takes into account CPU use, memory consumption, network usage, and system call data. The results suggest that this combination of techniques performs well. It can detect ransomware with 100% precision and a false positive rate of less than 4%.

In (Chen et al., 2017), researchers proposed a dynamic ransomware detection method for identifying known and undiscovered malware utilizing data mining techniques such as Random Forest, Support Vector Machine, Simple Logistic, and Naive Bayes. To deceive the most recent concealment strategies, they utilized a dynamic approachable to monitor program actions and build an API call flow graph. Results obtained good performances.

Authors in (Jeng et al., 2022) proposed a malware classification method based on the static feature in the Android environment. In particular, they extract the opcode from the static feature of the applications selected to build the dataset. Jeng *et al.* also applied an attention mechanism that improved the accuracy of the CNN model for the classification performed.

In (Xing et al., 2022), authors introduced a deep learning method to recognize malware using an autoencoder network. To train the latter Xing *et al.* employed a dataset composed of gray-scale images retrieved from Android applications. The detection method proposed achieved a good accuracy of 96%. Despite them, who obtain images from applications by converting bytecodes, we create a script to detect the opcodes present in each executable program. We transformed each element stored in the list of opcodes into a triple of integers to create an RGB picture.

## 3 METHOD

In this section, we describe the proposed method for ransomware detection by means of quantum machine learning. Thus, we introduce a model combining quantum and convolutional layers, and several (fully) convolutional models exploited for direct comparison with the quantum model, in terms of performance, are

<sup>3</sup><https://www.fortinet.com/resources/cyberglossary/ransomware-statistics>

```

let filter_words as a set of opcode
let newLines as list
let data as list

for file in dir:
    subprocess.run(objdump -b binary -D -m \
        i386 file > application.txt)

for file in dir:
    if file.endswith('.txt'):
        open(file, 'r') as f:
            for line in f:
                words = line.split()

                for word in words:
                    if word in filter_words:
                        newLines.append(word)

        open(file, 'w') as f:
            for line in newLines:
                f.write(line)
            f.close

for file in dir:
    open(file, 'r') as f:
        for line in f:
            for key, values in \
                dictionary.items():
                    if key == item:
                        data.append(values)

        open(file, 'w') as f:
            for line in newLines:
                f.write(line)
            f.close

```

Listing 1: The script converts the applications into images, by invoking the *objdump* disassembler to obtain the opcodes and then convert each opcode into an RGB pixel.

presented. The proposed method is depicted in Figures 1 and 2: in particular in Figure 1 we show how we represent an application as an image (by obtaining the opcode sequences through a reverse engineering process), while Figure 2 is related to the hybrid quantum model building and evaluation.

From a (ransomware, generic malware, or legitimate) application under analysis, we obtain the application opcodes through a Python script developed by authors (shown in Listing 1).

From the Listing 1 we observe that to obtain the opcode for each application we exploited to *objdump* disassembler, natively available on Linux operating systems<sup>4</sup>. Thus, the extracted opcode sequence is

<sup>4</sup><https://man7.org/linux/man-pages/man1/objdump.1.html>

stored into a text file (from row 6 to row 8 in Listing 1).

Once stored the opcode code sequence into a text file (named *application.txt* in Listing 1), from row 10 to row 24 in Listing 1, each opcode is filtered i.e., we consider opcodes without argument, and we save the opcode list in a text file (we overwrite the *application.txt* file).

From row 26 to row 37 in Listing 1 we set a unique color for each opcode found previously in RGB (i.e., a triple of integers) and save them overwriting the previous *application.txt* file. Obviously, the same opcode will be coded with the same color for all applications. Subsequently, by storing each pixel, we compose the image for the application under analysis.

Figures 3, 4 and 5 report three examples of images obtained by applying the steps depicted in Figure 1 related to two different ransomware applications (shown in Figure 3 and 4) and a legitimate one, shown in Figure 5. The sizes of the output images are not equal to each other, this happens because the proportions are related to the number of opcodes contained in the analyzed file.

In particular in Figure 3 we show the image obtained from the ransomware identified by the 2d7ded792db11781276f2e5f42b29b1ae37055b1f306c5bc247448275ea6b40f hash.

This ransomware is belonging to the Shade family, diffused through malicious websites and infected email attachments. Once infiltrated, the ransomware belonging to the Shade family are able to encrypt most files stored on the infected machine<sup>5</sup>. The VirusTotal report of this sample is available<sup>6</sup>.

In Figure 4 a second example of an image obtained from a ransomware sample (identified by the fbeb235e9a95e4cfef6d6fe2201e3bf9350eb23f79a98c449613951c2384c4671 hash) is shown.

The image shown in Figure 4 is related to a ransomware sample belonging to the Ransom-NB family delivered, similarly to the Shade ransomware, as attached to phishing emails but also as a repercussion of user winding up on a repository that hosts a harmful software. This ransomware too is able to cipher files on the victim's computer and it is also able to show a ransom money note<sup>7</sup>.

The Virustotal report of this sample is available<sup>8</sup>.

<sup>5</sup><https://www.malwarebytes.com/blog/news/2019/03/s-potlight-troldesh-ransomware-aka-shade>

<sup>6</sup><https://www.virustotal.com/gui/file/2d7ded792db11781276f2e5f42b29b1ae37055b1f306c5bc247448275ea6b40f>

<sup>7</sup><https://howtofix.guide/win32ransom-nb-trj/>

<sup>8</sup><https://www.virustotal.com/gui/file/fbeb235e9a95e4cfef6d6fe2201e3bf9350eb23f79a98c449613951c2384c4671>

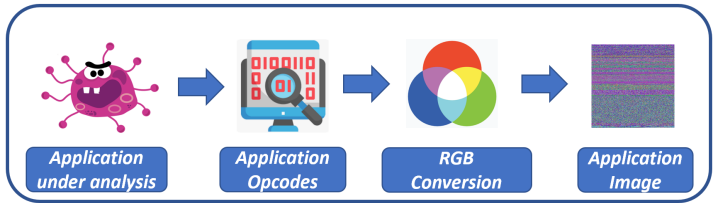


Figure 1: Steps to convert an application into an image through the application opcodes.

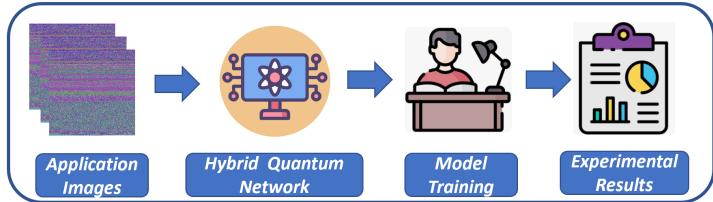


Figure 2: Steps for ransomware detection model building with the hybrid quantum network.

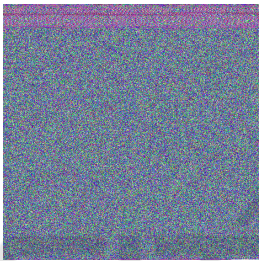


Figure 3: An example of an image obtained from a ransomware sample belonging to the Shade family.

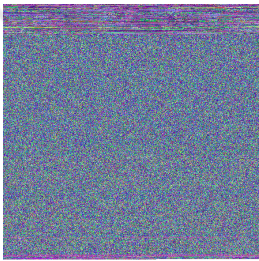


Figure 4: An example of an image obtained from a ransomware sample belonging to the Ransom-NB family.

The third example of image is obtained from a legitimate application i.e., WordPad, a word processing built-in in Microsoft Windows operating systems, which allows to create simple texts with basic formatting.

In particular, we consider the 22H2 version of WordPad included in the Microsoft Windows 10 operating system, which hash is a70d52eda892edc073932b462cc367cdbfbace3f4196857d8d4fa869a13de792. The related VirusTotal report is available also for this application<sup>9</sup>.

<sup>9</sup><https://www.virustotal.com/gui/file/a70d52eda892edc073932b462cc367cdbfbace3f4196857d8d4fa869a13de7>

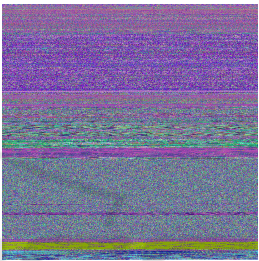


Figure 5: An example of an image obtained from the WordPad application, a legitimate application included in the Microsoft Windows 10 operating system.

The hybrid quantum model building and evaluation is shown in Figure 2. Once obtained a set of images belonging to (ransomware, generic malware and trusted) applications, we need a quantum deep learning model.

We call the proposed quantum machine learning model **Hybrid Quantum Convolutional Neural Network** i.e., Hybrid-QCNN. The proposed quantum model is considered hybrid because it considers quantum and classical neural network-based function blocks composed with one another in the topology of a directed graph. We consider this a rendition of a hybrid quantum-classical computational graph where the inner workings (variables, component functions) of various functions are abstracted into boxes. The edges represent the flow of classical information through the metanetwork of quantum and classical functions(Broughton et al., 2020).

The Python code we developed for the Hybrid-QCNN is shown in Listing 2: in rows 2,3,4 and 5 there is the quantum layer (i.e., QConv), while from rows 6 to 11 there are the convolutional layers, in particular a Conv2D, a Flatten and two Dense layers.



---

```

model = models.Sequential()
model.add(QConv(filter_size=2, depth=8,
    activation='relu', name='qconv1',
    input_shape=(self.input_width_height,
        self.input_width_height, \
        self.channels)))
model.add(layers.Conv2D(16, (2, 2),
    activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(32, activation='relu'))
model.add(layers.Dense(self.num_classes,
    activation='softmax'))

```

---

Listing 2: The Hybrid Convolutional Neural Network for ransomware detection.

With the aim to compare to Hybrid-QCNN model with (full) convolutional models, the following architectures are exploited in this paper: a Convolutional Neural Network (i.e., called Standard CNN) model designed by authors, *AlexNet* (Alom et al., 2018), *LeNet* and *MobileNet*.

---

```

model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3),
    activation='relu',
    input_shape=(self.input_width_height,
        self.input_width_height,
        self.channels)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), \
    activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), \
    activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dropout(0.5))
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(256, activation='relu'))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(self.num_classes,
    activation='softmax'))
model.compile(loss='categorical_crossentropy',
    optimizer=Adam(self.learning_rate),
    metrics=['acc', Precision(name="prec"),
    Recall(name="rec"), AUC(name="auc")])

```

---

Listing 3: The developed *Classic-CNN* network.

Due to hardware limitations, we need to consider small images for the Hybrid-QCNN model: in order to make a fair and direct comparison we try to set the same parameters we consider for the Hybrid-QCNN also for the remaining networks. As a matter of fact, for the Hybrid-QCNN model, an image size equal to 25x25x1 is considered. Once built the Hybrid-QCNN and the remaining models, the evaluation results are analysed in order to perform the model comparison for ransomware detection.

## 4 EXPERIMENTAL ANALYSIS

In this section, we present and discuss the results of the experiments we performed.

Relating to the model implementation, we resort to the Python programming language and we considered two different open-source libraries, Tensorflow<sup>10</sup> and Tensorflow Quantum<sup>11</sup>, to develop the models. TensorFlow (Goldsborough, 2016) is an open-source machine learning software library that provides tested and optimized modules for building deep learning models, while Tensorflow Quantum (Broughton et al., 2020) is a library aimed to offer high-level abstractions for the design and training of both discriminative and generative quantum, compatible with the TensorFlow library, along with quantum circuit simulators.

The machine considered for the experiments is an Intel Xeon Gold 6140M CPU at 2.30GHz, 64GB RAM equipped with the Ubuntu 22.04.01 LTS operating system.

A dataset composed by 15000 real-world applications is considered: in particular we take into account 5000 ransomware, 5000 generic malware, and 5000 trusted applications. The generic malware and the ransomware samples were obtained from the VirusShare<sup>12</sup> web site, a repository of malware samples to provide security researchers, incident responders and forensic analysts, while the legitimate samples were collected from libraries (for example, DLL files) and executable files obtained from a Microsoft Windows 10 machine. To confirm the maliciousness or the trustworthiness of the samples we obtained, we submitted the ransomware, (generic) malware and legitimate samples to the Virustotal<sup>13</sup>, and the Jotti<sup>14</sup> web services. In particular, we confirm that all the ransomware considered in the experiment are crypto-ransomware, i.e., malicious files with the ability to hunt for and encrypt targeted files (for example, image, video, and document files), for instance, belonging to the TeslaCrypt, Wawnnacry and Petya families.

We exploited the code snippet shown in Listing 1 to obtain the related images from ransomware, (generic) malware and trusted applications.

We split the dataset into training, validation and testing, in particular, 12000 applications are related to the training dataset (i.e., 4000 ransomware, 4000 generic malware and 4000 trusted), 1500 applications (i.e., 500 ransomware, 500 generic malware and 500 trusted) are reserved for the validation and the remain-

<sup>10</sup><https://www.tensorflow.org/>

<sup>11</sup><https://www.tensorflow.org/quantum>

<sup>12</sup><https://virusshare.com/>

<sup>13</sup><https://www.virustotal.com/>

<sup>14</sup><https://virusscan.jotti.org/>

Table 1: Hyper-parameters setting.

Model	Epochs	Batch size	Learning rate	Image size
STANDARD_CNN	10	16	0.01	100x3
ALEX_NET	10	16	0.01	100x3
MobileNet	10	16	0.01	50x3
Hybrid-QCNN_1	10	64	0.0001	25x3
Hybrid-QCNN_2	20	64	0.0001	25x3
Hybrid-QCNN_3	20	16	0.001	25x3
Hybrid-QCNN_4	20	32	0.0001	25x3
Hybrid-QCNN_5	20	64	0.001	25x3
Hybrid-QCNN_6	20	16	0.0001	25x3
Hybrid-QCNN_7	15	16	0.0001	25x3
Hybrid-QCNN_8	15	16	0.001	25x3
Hybrid-QCNN_9	15	64	0.0001	25x3
Hybrid-QCNN_10	15	32	0.0001	25x3
Hybrid-QCNN_11	15	16	0.001	25x3
Hybrid-QCNN_12	20	32	0.001	25x3
Hybrid-QCNN_13	10	32	0.001	25x3
Hybrid-QCNN_14	10	64	0.001	25x3
Hybrid-QCNN_15	10	32	0.0001	25x3
Hybrid-QCNN_16	10	16	0.0001	25x3
Hybrid-QCNN_17	10	16	0.001	25x3
Hybrid-QCNN_18	15	32	0.001	25x3

ing 1500 (i.e., 500 ransomware, 500 generic malware and 500 trusted) as testing dataset.

In Table 1 are shown the hyper-parameters for the models considered in the experimental evaluation.

As shown in Table 1 we perform several experiments, with the aim to tune the Hybrid-QCNN with the best parameters: we consider 18 different configurations for the Hybrid-QCNN (each configuration is different from the other ones in terms of epochs, batch size or learning rate).

The image size is fixed due to the huge request for computational resources: it is limited for the Hybrid-QCNN to 25x3 i.e., we resize the images with a width and a height of 25 pixels for 3 channels (i.e., RGB). Relating to the STANDARD\_CNN, Alex\_Net and MobileNet networks, we consider an image size respectively equal 100x3 for the STANDARD\_CNN and Alex\_Net networks and equal to 50x3 for the MobileNet model, by considering for these models a number of epochs of 10. We consider these parameters, for the (fully) convolutional models to make a comparison as direct as possible with the Hybrid-QCNN by considering, when possible, the same hyper-parameters.

In Table 2 we present the experimental analysis results for each configuration considered in Table 1.

As shown from Table 2 the Hybrid-QCNN\_1 model reaches an accuracy equal to 0.73, a precision of 0.75, and a recall equal to 0.7: this is symptomatic that the proposed hybrid quantum convolutional model is able to discriminate between ransomware, (generic) malware and trusted applications. Also the other Hybrid-QCNN configurations generally obtain interesting performances, for instance, the Hybrid-QCNN\_2 model reaches an accuracy equal to 0.712, the Hybrid-QCNN\_4 accuracy is of 0.728. The worst accuracy is obtained by the Hybrid-QCNN\_12

Table 2: The results of the experimental analysis.

Model	Loss	Accuracy	Precision	Recall
STANDARD_CNN	1.0	0.33	0	0
ALEX_NET	1.10	0.33	0	0
MobileNet	1.17	0.39	0.47	0.25
Hybrid-QCNN_1	0.71	0.73	0.75	0.70
Hybrid-QCNN_2	0.745	0.712	0.735	0.686
Hybrid-QCNN_3	2.411	0.713	0.713	0.709
Hybrid-QCNN_4	0.713	0.728	0.754	0.705
Hybrid-QCNN_5	1.01	0.707	0.713	0.694
Hybrid-QCNN_6	0.713	0.731	0.749	0.712
Hybrid-QCNN_7	0.735	0.723	0.748	0.702
Hybrid-QCNN_8	1.502	0.712	0.715	0.708
Hybrid-QCNN_9	0.790	0.678	0.693	0.649
Hybrid-QCNN_10	0.736	0.726	0.747	0.700
Hybrid-QCNN_11	2.543	0.679	0.682	0.677
Hybrid-QCNN_12	1.098	0.332	0.0	0.0
Hybrid-QCNN_13	0.946	0.703	0.714	0.688
Hybrid-QCNN_14	0.779	0.722	0.739	0.712
Hybrid-QCNN_15	0.708	0.727	0.749	0.705
Hybrid-QCNN_16	0.746	0.713	0.744	0.690
Hybrid-QCNN_17	0.933	0.707	0.715	0.698
Hybrid-QCNN_18	1.098	0.333	0.0	0.0

configuration, where an accuracy equal to 0.332 is reached, probably due to the extended number of epochs (equal to 20, as shown from Table 1) that conduct the network in overfitting, with the related performance decay. Relating to the remaining configurations, as shown from the results obtained in Table 2, the performances obtained by the Hybrid-QCNN are satisfactory for the ransomware detection task. With regard to the (fully) convolutional models, built with the set of parameters shown in Table 1, obtain an accuracy equal to 0.33 (for the STANDARD\_CNN, Alex\_Net) models and equal to 0.39 for the MobileNet one, confirming that with a similar set of hyper-parameters, the Hybrid-QCNN is able to obtain better performances in the discrimination of ransomware, (generic) malware and legitimate applications.

This result is symptomatic of how promising hybrid networks can be, in fact considering that due to the limitations of the hardware on which the experiments were performed it was possible to evaluate a hybrid model with only one convolutional layer, superior performances were obtained from convolutional networks composed of many more layers. Therefore, probably, when it will be possible to test quantum networks with different convolutional layers and with a greater number of epochs, but also with a greater size of images, higher performances than those of the current convolutional networks will be obtained.

Figure 6 shows the confusion matrix (considering only the testing dataset) for the Hybrid-QCNN\_1 configuration. We show the confusion matrix of this specific configuration considering that with this set of hyper-parameters we obtained the best performances

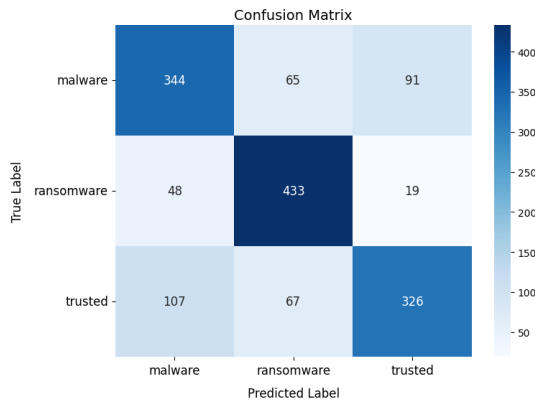


Figure 6: Confusion matrix for the Hybrid-QCNN\_1 model.

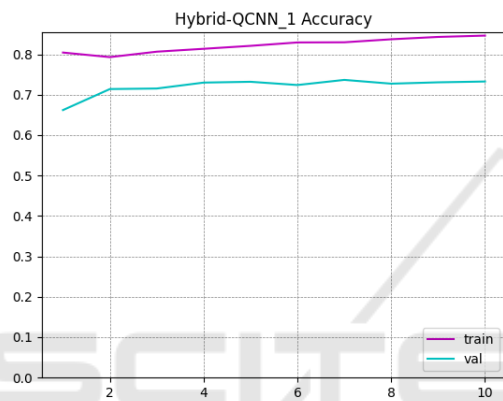


Figure 7: Training and validation accuracy trend for the Hybrid-QCNN\_1 model.

in terms of accuracy, as shown from the experimental analysis results shown in Table 2.

From the confusion matrix in Figure 6 we can note that most of the applications were detected as belonging to the right category: as a matter of fact, 344 (generic) malware (on 500), 433 ransomware (on 500) and 326 trusted applications (on 500) were rightly recognised. Relating to the misclassifications, 65 ransomware and 91 trusted applications were wrongly labelled as (generic) malware, 48 (generic) malware and 19 trusted application were wrongly labelled as belonging to the ransomware category and 107 malware and 67 ransomware were wrongly classified as trusted applications.

To understand how the Hybrid-QCNN\_1 is learning through the epochs, in the follow we consider the trends of accuracy and loss in both training and validation.

Figure 7 shows the accuracy trend of the training and the validation for the Hybrid-QCNN\_1 model.

From the trends shown in Figure 7 it emerges that the network is training during the 10 epochs, and for each epoch is learning more information aimed to

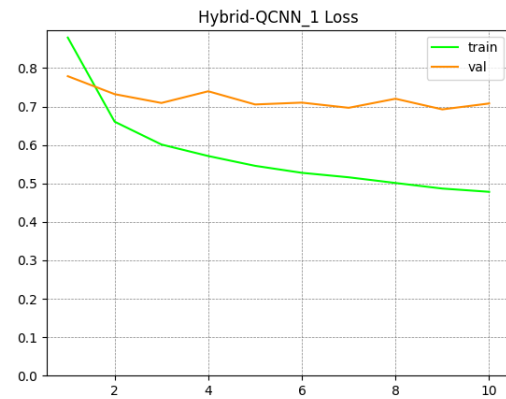


Figure 8: Training and validation loss trend for the Hybrid-QCNN\_1 model.

rightly discriminate between ransomware, malware, and legitimate applications.

Figure 8 shows the trend of the training and the validation loss for the Hybrid-QCNN\_1 model.

From the loss trend shown in Figure 8, we can note that the loss is decreasing, and this is expected behaviour: this happens for both the training and the validation. In particular, the decreasing trend is more accentuated for the training, but also in the validation we can note this behaviour.

The trend of loss that decreases, when accuracy increases simultaneously with increasing epochs is a symptom of the fact that the model is correctly learning the information to distinguish between ransomware, (generic) malware, and legitimate applications, thus confirming the experimental analysis results and that quantum machine learning is promising in ransomware detection.

## 5 CONCLUSION AND FUTURE WORK

One of the most sneaky threats in the malware landscape is represented by the so-called ransomware, a malware aimed to cipher data and user documents asking for a certain amount of money (typically in Bitcoin) to hand data access back to users. Free and commercial antimalware are not adequate to detect novel malicious payloads, this is the reason why ransomware are free to perpetrate damages undisturbed. Considering the recent introduction of quantum computation in machine learning, in this paper we propose a method for ransomware detection exploiting quantum machine learning. We propose a hybrid neural network model composed by quantum and convolutional layers. We experiment several configurations with the proposed model in the evaluation of a dataset

composed by 15000 real-world applications (5000 ransomware, 5000 generic malware, and 5000 legitimate applications), by obtaining an accuracy equal to 0.73, by confirming that quantum machine learning can be promising in ransomware detection.

Future works will consider novel quantum deep learning models, composed of several quantum layers. Moreover, we will investigate whether quantum machine learning can be considered for the detection of malware families and variants, not only categories such as ransomware and malware.

## ACKNOWLEDGMENT

This work has been partially supported by EU DUCA, EU CyberSecPro, SYNAPSE, PTR 22-24 P2.01 (Cybersecurity) and SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the EU - NextGenerationEU projects, by MUR - REASONING: foRmal mEthods for computAtional analySis for diagnOsis and progNosis in imagING - PRIN, e-DAI (Digital ecosystem for integrated analysis of heterogeneous health data related to high-impact diseases: innovative model of care and research), Health Operational Plan, FSC 2014-2020, PRIN-MUR-Ministry of Health, the National Plan for NRRP Complementary Investments D<sup>3</sup> 4 Health: Digital Driven Diagnostics, prognostics and therapeutics for sustainable Health care, Progetto MolisCTe, Ministero delle Imprese e del Made in Italy, Italy, CUP: D33B22000060001, FORESEEN: FORMAL mEthodS for attack dEtEction in autonomous drivINg systems CUP N.P2022WYAEW and ALOHA: a framework for monitoring the physical and psychological health status of the Worker through Object detection and federated machine learning, Call for Collaborative Research BRic -2024, INAIL.

## REFERENCES

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Van Esesn, B. C., Awwal, A. A. S., and Asari, V. K. (2018). The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*.

Broughton, M., Verdon, G., McCourt, T., Martinez, A. J., Yoo, J. H., Isakov, S. V., Massey, P., Halavati, R., Niu, M. Y., Zlokapa, A., et al. (2020). Tensorflow quantum: A software framework for quantum machine learning. *arXiv preprint arXiv:2003.02989*.

Chen, Z.-G., Kang, H.-S., Yin, S.-N., and Kim, S.-R. (2017). Automatic ransomware detection and analysis based on dynamic api calls flow graph. In *Proceedings*

*of the International Conference on Research in Adaptive and Convergent Systems*, pages 196–201.

Ciaramella, G., Iadarola, G., Mercaldo, F., Storto, M., Santone, A., and Martinelli, F. (2022). Introducing quantum computing in mobile malware detection. In *Proceedings of the 17th International Conference on Availability, Reliability and Security*, pages 1–8.

Ciaramella, G., Martinelli, F., Mercaldo, F., and Santone, A. (2023). Exploring quantum machine learning for explainable malware detection. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE.

Ferrante, A., Malek, M., Martinelli, F., Mercaldo, F., and Milosevic, J. (2017). Extinguishing ransomware—a hybrid approach to android ransomware detection. In *International Symposium on Foundations and Practice of Security*, pages 242–258. Springer.

Goldsbrough, P. (2016). A tour of tensorflow. *arXiv preprint arXiv:1610.01178*.

He, H., Yang, H., Mercaldo, F., Santone, A., and Huang, P. (2024). Isolation forest-voting fusion-multioutput: A stroke risk classification method based on the multidimensional output of abnormal sample detection. *Computer Methods and Programs in Biomedicine*, 253:108255.

Jeng, T.-H., Chang, Y.-C., Yang, H.-H., Chen, L.-K., and Chen, Y.-M. (2022). A novel deep learning based attention mechanism for android malware detection and explanation. In *Proceedings of the 10th International Conference on Computer and Communications Management*, pages 226–232.

Mercaldo, F., Ciaramella, G., Iadarola, G., Storto, M., Martinelli, F., and Santone, A. (2022). Towards explainable quantum machine learning for mobile malware detection and classification. *Applied Sciences*, 12(23):12025.

Xing, X., Jin, X., Elahi, H., Jiang, H., and Wang, G. (2022). A malware detection approach using autoencoder in deep learning. *IEEE Access*, 10:25696–25706.