Improving Robustness Against Adversarial Attacks with Deeply Quantized Neural Networks

Ferheen Ayaz‡, Idris Zakariyya*, José Cano*,
Sye Loong Keoh*, Jeremy Singer*, Danilo Pau†, Mounia Kharbouche-Harrari†
*University of Glasgow, UK †STMicroelectronics ‡University of Sussex, UK

2023 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-8867-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/IJCNN54540.2023.10191429

***Abstract*—Reducing the memory footprint of Machine Learning (ML) models, particularly Deep Neural Networks (DNNs), is essential to enable their deployment into resource-constrained tiny devices. However, a disadvantage of DNN models is their vulnerability to adversarial attacks, as they can be fooled by adding slight perturbations to the inputs. Therefore, the challenge is how to create accurate, robust, and tiny DNN models deployable on resource-constrained embedded devices. This paper reports the results of devising a tiny DNN model, robust to adversarial black and white box attacks, trained with an automatic quantization-aware training framework, i.e. QKeras, with deep quantization loss accounted in the learning loop, thereby making the designed DNNs more accurate for deployment on tiny devices. We investigated how QKeras and an adversarial robustness technique, Jacobian Regularization (JR), can provide a co-optimization strategy by exploiting the DNN topology and the per layer JR approach to produce robust yet tiny deeply quantized DNN models. As a result, a new DNN model implementing this co-optimization strategy was conceived, developed and tested on three datasets containing both images and audio inputs, as well as compared its performance with existing benchmarks against various white-box and black-box attacks. Experimental results demonstrated that on average our proposed DNN model resulted in 8.3% and 79.5% higher accuracy than MLCommons/Tiny benchmarks in the presence of white-box and black-box attacks on the CIFAR-10 image dataset and a subset of the Google Speech Commands audio dataset respectively. It was also 6.5% more accurate in the presence of black-box attacks on the SVHN image dataset.**

***Index Terms*—Deep Neural Networks (DNNs), QKeras, Jacobian Regularization (JR), Adversarial Attacks.**

I. Introduction

Deep Neural Networks (DNNs) demonstrate remarkable performance in various tasks such as natural language processing, cybersecurity, computer vision, intelligent applications and many more [1]. However, DNN models are resource intensive with large memory footprint and computational requirements. Moreover, the increasing requirements of edge intelligence have given rise to new optimization strategies in Machine Learning (ML) which strive to reach optimal accuracy while shrinking the DNN model architectures at the same time [2]. The specific sub-discipline of ML that generates constrained ML workloads to be deployed on a target edge device, such as Microcontroller Units (MCUs) and sensors, is called Deeply Quantized Machine Learning (DQML). However, DNN models are often vulnerable to adversarial attacks causing changes to the input imperceptible to the human eye [3]. These vulnerabilities are critical, as they restrict the deployability of DNN models as an effective solution for real world applications such as autonomous cars, smart cities, intelligent applications and responsive Artificial Intelligence (AI) [4]. Two widely used classes of adversarial attacks are *white-box* and *black-box* attacks. In white-box, the attacker has full knowledge of the DNN model, its structure and parameters, whereas in the black-box paradigm, the attacker is unaware of the used DNN model. In both scenarios, these attacks aim to cause deliberate mis-classifications or to disrupt the model performance. As such, there is an urgent need of balancing the trade-off between reducing the memory footprint of DNN models for tiny embedded devices while making them robust against adversarial attacks.

DQML offers a promise in terms on executing very low bit depth ML models on resource-constrained MCUs with μW power and a memory of only a few MBytes [5]. Various frameworks such as TensorFlow Lite (TFLite) [6] for post-training optimization and Larq [7], Brevitas [8] and QKeras [9] for deep quantization-aware training, are used to optimize ML models resources. Most of these frameworks use quantization to optimize the utilized ML models based on data precision.

QKeras is designed to offer quantization as low as a single-bit, and at the same time retaining the model accuracy through introducing quantization error in the form of random noise and learning to overcome it during training [10]. It is based on drop-in replacement functions for Keras, thus providing the freedom to add a quantizer and choose quantization bit-depth separately for activations, biases and weights per layer. This is useful for efficient training of quantized DNN models. Among various deep quantization strategies offered by QKeras, there is stochastic quantization [11], which instead of quantizing all elements (parameters) of a DNN model, quantizes a portion of the elements with a stochastic probability inversely proportional to the quantization error, while keeping the other portion unchanged in full-precision (FP). The quantized portion is gradually increased at each iteration until potentially the entire DNN is quantized. This procedure greatly and incrementally compensates the quantization error and thus yields better accuracy for very low-bit-depth DNN.

In parallel, exploring the robustness of DNN models is critical to be integrated within DQML. In fact, enhancing the security while enabling the model's deployment in resource-constrained MCUs is a key challenge. Various defensive techniques and models for DNNs are present in the literature

This work was done when Ferheen Ayaz was with University of Glasgow.

Authorized licensed use limited to: City St Georges University of London. Downloaded on June 12,2025 at 18:15:09 UTC from IEEE Xplore. Restrictions apply.

to provide robustness against adversarial attacks [12]. However, such defensive mechanisms may result in an increased model size or accuracy drop for clean sets. Considering DQML models, which require extensive learning computation to reach optimal size and accuracy, with possible vulnerability to adversarial attacks, any addition to the model size or drop in accuracy can affect the deployment performance. In view of that, recent studies [13], [14] have demonstrated that quantization can reduce computational requirements while granting robustness to a certain level of white-box adversarial attacks.

Motivated by such an observation, this paper investigates the following hypothesis: *Can a per-layer hybrid quantization scheme inherit robustness against white-box and black-box attacks while maintaining the trade-off between clean set accuracy performance and limited-resources requirements of tiny embedded devices?* We propose a DQML model that is deployable on tiny devices and highly robust to adversarial attacks, trained using the QKeras framework. Our theoretical investigation shows that QKeras utilizes Jacobian Regularization (JR) as an adversarial attack defensive mechanism. Based on this, we use QKeras to propose a Stochastic Ternary Quantized (STQ) DNN model with accuracy performance suitable for deployment in tiny MCUs, and potentially for image and audio in the same sensor package embodiment. Its ability to provide robustness against various adversarial attacks has been proven. The contributions of this paper are as follows:

- Development of an STQ-based model that is less complex and can be deployed on MCUs with minimal memory footprint requirements and an improved accuracy performance on clean sets (Section III).
- Analysis of the robustness of the STQ-based model according to its similarity with the JR technique, and demonstration of its resilience against various white and black box adversarial attacks (Sections III and IV).
- A comprehensive comparison of the resilience of the STQ-based model to industry endorsed MLCommons/Tiny benchmarks for both image and audio inputs while investigating its performance with K-fold cross validation (Section IV).

II. Background and Related Work

A. Adversarial Attacks Against Machine Learning Models

Security considerations are important for sensitive applications like intelligent transportation systems, healthcare and financial systems which utilize image and voice recognition AI-based models [15]. A disadvantage of ML and AI models developed for such tasks is that they can be vulnerable to attacks which can compromise their integrity, confidentiality and privacy in real world applications. In particular, adversarial attacks involve adding a small perturbation to the input to maximize the loss function of a model under a constrainted norm [16]. Equation (1) expresses such a procedure of introducing a perturbation into an input data, where: $\mathcal{L}(\theta, x', y)$ is the loss function, θ denotes the model parameters, x' is the perturbed input, y is the model output, δ denotes the perturbation and p is the perturbation norm [17]. This work considers the two types of attacks defined in Section I, i.e. white-box and black-box attacks.

A common adversarial attack technique is the Fast Gradient Sign Method (FGSM) [18]. This is a white-box attack which uses a single-step iteration to estimate the gradient of the model training loss function based on the inputs. An FGSM [18] attack procedure is expressed in (2), where Δ represents the gradient and ϵ denotes a small constant value that restricts the perturbation. A variation of FGSM is the Projected Gradient Descent (PGD) [18]. This is a more computationally expensive multi-step threat model which runs several iterations to find an adversarial input with the lowest possible δ, as expressed in (3), where i is the iteration index, α denotes the gradient step size and S represents the perturbation set.

$$\max_{||\delta||_p} \mathcal{L}(\theta, x', y) \tag{1}$$

$$x' = x + \epsilon \cdot sign(\Delta_x \mathcal{L}(\theta, x, y)) \tag{2}$$

$$x'_{i+1} = \prod_{x+S} (x + \alpha \cdot sign(\Delta_x \mathcal{L}(\theta, x, y))) \tag{3}$$

An efficient black-box attack is the Square attack [19], which is based on a random search optimization technique with multiple iterations. In each iteration, it changes a small fraction of the input shaped into square at random positions. Similar to gradient-based optimizations, it also relies on step-size reduction, where the size refers to the dimensions of the square [19]. Another form of black-box attack is the Boundary attack [20], where the queries are used to estimate the decision boundaries of the output classes. Starting with a clean image, gradient estimation is performed with queries, moving along the estimated direction in each iteration and projecting a new perturbation until the model decision is changed [20].

Overcoming the white-box and black-box attack methods requires a suitable defensive mechanism. Therefore, it is important to enhance the model performance against different attacks variations to ease its deployment.

B. Robustness against Adversarial Attacks

Various defense methods have been proposed to increase the robustness of DNNs against adversarial attacks [21]. Some strategies aim at detecting adversarial inputs [22] or performing transformations to remove perturbations [23] [24] through an additional network or module. Adversarial training introduces adversarial inputs during the model's learning so that it learns not to misinterpret them [25]. However, these approaches do not guarantee robustness against attacks which are not introduced to the model during training, as they are not learned by the model. Other methods such as [26] decrease the model's sensitivity to small perturbations by adding a regularization term in the loss function. Equation (4) expresses this joint loss function, where $\mathcal{L}_{reg}$ denotes the regularization term and λ is a hyper-parameter used to allow the adjustment between the regularization and the actual loss.

$$\mathcal{L}_{joint} = \mathcal{L}(\theta, x, y) + \lambda L_{reg}, \tag{4}$$

Authorized licensed use limited to: City St Georges University of London. Downloaded on June 12,2025 at 18:15:09 UTC from IEEE Xplore. Restrictions apply.

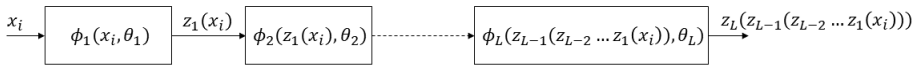


Fig. 1: Transformation of input x_i into output z_L by DNN.

C. Jacobian Regularization

Jacobian Regularization (JR) is a technique to provide adversarial robustness, where the input gradient regularization normalizes the gradient of the cross-entropy loss, such that $\mathcal{L}_{reg} = \sum_{k=1}^{K} ||\Delta_x z_k^L(x_i)||_2^2 = ||J(\mathbf{x}_i)||_F^2$ and $||J(\mathbf{x}_i)||_F^2$ is the Frobenius norm of the model's Jacobian matrix evaluated on the input data [27]. To reduce computational complexity, per-layer JR is proposed in [28].

To demonstrate the robustness of per-layer JR, let's consider a D-dimensional network input X consisting of N training samples. As shown in Figure 1, the network contains $l = 1, 2, ..., L$ layers. z_l denotes the output at layer l and z_l^k is the output of the kth neuron of the lth layer. Consider the softmax operation at the output of the network; the predicted final output for computing the top-1 accuracy for an input x_i is $f(x_i) = argmax\{z_L^1, z_L^2, .., z_L^K\}$, where K is the dimension of the output vector. The Jacobian matrix of a DNN is computed at L^{th} layer, i.e. $J_L(x_i) = \nabla_x z_L(x_i)$, and is defined as:

$$J_L(x_i) = \begin{bmatrix} \frac{\partial z_L^1 x_1}{\partial x_1} & \frac{\partial z_L^1 x_1}{\partial x_2} & \dots & \frac{\partial z_L^1 x_1}{\partial x_D} \\ \frac{\partial z_L^2 x_2}{\partial x_1} & \frac{\partial z_L^2 x_2}{\partial x_2} & \dots & \frac{\partial z_L^2 x_2}{\partial x_D} \\ . & . & . & . \\ . & . & . & . \\ \frac{\partial z_L^K x_K}{\partial x_1} & \frac{\partial z_L^K x_K}{\partial x_2} & \dots & \frac{\partial z_L^K x_K}{\partial x_D} \end{bmatrix} \epsilon \mathbb{R}^{K \times D}. \quad (5)$$

In [27], the JR term for an input x_i is defined as

$$||J(x_i)||_F^2 = \sum_{d=1}^{D} \sum_{k=1}^{K} \left(\frac{\partial}{\partial x_d} z_L^k(x_i) \right)^2 = \sum_{k=1}^{K} ||\nabla_x z_L^k(x_i)||_2^2, \quad (6)$$

The standard loss of the training is added with the regularization term in (6) to improve the robustness of the DNN. It is proposed as a post-training, in which the network is re-trained for fewer iterations with the new loss function [27]. As $i\epsilon\{1, 2, ..., N\}$, JR requires the computation of N gradients, whereas in per-layer JR, the Jacobian matrix is computed on only one random i at each layer [28]. The basic idea is to reduce the Frobenius norm of the Jacobian matrix which results in the expansion of the classification margin, i.e. the distance between an input and the decision boundary induced by a network classifier.

D. Adversarial Robustness of Quantized Models

Model quantization techniques are widely used in various fields [29]. In [30], it is used for anomaly detection and thwarting cyber attacks in Internet-of-Things (IoT) networks. However, the limitation of a quantized model is the shift of the FP model classification boundary, which may influence how vulnerable the model is to adversarial perturbations [31]. As such, the authors in [31] investigated the use of a boundary-based retraining method to reduce adversarial and quantization losses with the usage of non-linear mapping as a defensive mechanism against white-box adversarial attacks. Other previous studies explored the impact of perturbations on different models. Table I compares some existing works that investigated the robustness of quantized DNN models. These studies investigated the resilience of quantized DNN models without examining their deployment feasibility on resource-constrained devices, such as MCUs or sensors, and most of them considered only white-box attacks.

TABLE I: Comparison of adversarially quantized DNN models.

Works		[32]	[33]	[34]	[35]	Ours
Compress. technique	Method	N bit-width quantization				SQ*
	on weights	✓	✓	✓	✓	✓
	on activat.	✓	✓	✓	✗	✓
Inputs	Images	✓	✓	✓	✓	✓
	Audio	✗	✗	✗	✗	✓
Datasets		MNIST C-10*	MNIST SVHN	MNIST C-10* TinyI*	MNIST Spiral	C-10* SVHN GSC
Attacks	White box	FGSM PGD C&W	FGSM BIM C&W	✗	FGSM	FGSM PGD C&W
	Black box	✗	SPSA ZOO	✗	✗	Square Boundary ZOO
	Random	✗	✗	Gaus. noise	✗	✗
Memory footprint		✗	✗	✗	✗	410 KB

*C-10 denotes CIFAR-10 dataset, *TinyI: TinyImageNet, *SQ: Stochastic Quantization

III. Proposed DNN model

This section introduces a new robust and less complex DNN model based on the Stochastic Ternary Quantization (STQ) approach which is conceived to be deployable on tiny MCUs.

A. Adversarial Robustness in QKeras

QKeras [9] is a DNN framework targeting quantization based on Keras [36], which provides a productive methodology to build and train quantized neural networks, either fractional or integer spanning from 1 to 32 bits. QKeras performs quantization aware training [37]. The background algorithm to train neural networks with q bit-width weights, activations and gradient parameters is conceptualized in [38]. In particular, the network training in QKeras includes a backward propagation where parameter gradients are stochastically quantized into low bit-width numbers. Figure 2 shows the process flow of training each layer in QKeras. Each training iteration involves a forward propagation step to quantize weights, find output and add quantization error into the output of each layer. Then it includes a backward propagation step where gradients are stochastically quantized into low bit-width numbers. Finally, unquantized weights and gradients are updated for the next iteration. The training process of deeply quantized networks,

Authorized licensed use limited to: City St Georges University of London. Downloaded on June 12,2025 at 18:15:09 UTC from IEEE Xplore. Restrictions apply.

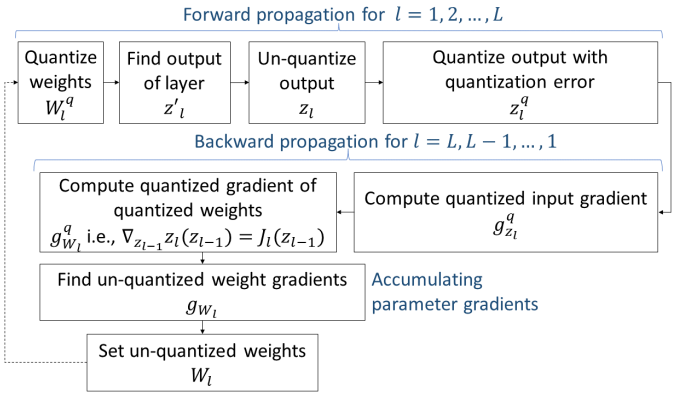


Fig. 2: QKeras quantization aware training flow.

through QKeras, make them robust to adversarial attacks due to the two following reasons: i) a noise function is introduced during quantization of gradients to overcome quantization error during training, which makes a DNN robust to noise and perturbation effect; ii) QKeras involves JR features that are explained below.

Theorem: *QKeras introduces per-layer JR and therefore increases the robustness of DNNs against adversarial attacks.*

Proof: Considering the per-layer structure of DNNs, as shown in Figure 1, the output z_L at the last layer L is

$$z_L = \phi_L((\phi_{L-1}(...\phi_1(x_i, \theta_1), ...\theta_{L-1}), \theta_L)), \tag{7}$$

where $\phi_l(., \theta_l)$ represents the function of the lth layer, θ_l denotes the model parameters at layer l, and $z_0 = x_i$ [28]. The Jacobian matrix of the lth layer is defined as

$$J_l(z_{l-1}) = \frac{dz_l}{dz_{l-1}}, \tag{8}$$

which is back-propagated during QKeras training (Step 10-16 of Algorithm 1 in [38]). The derivation expressed in (8) is the Jacobian matrix [28] of the lth block layers. Thus, QKeras with such patterns incorporates per-layer JR during training.

To prove that per-layer JR enhances adversarial robustness, consider a clean input x_c and an adversarial input x_p, both close to an input x_i and all belonging to the same class k. Since $f(x_i) = f(x_c) \neq f(x_p)$, then the ℓ_2 distance metric of the input and output of the network, as defined in [27], is

$$\frac{||x_p - x_i||_2}{||x_c - x_i||_2} \approx 1, \tag{9}$$

$$\frac{||z_L(x_p) - z_L(x_i)||_2}{||z_L(x_c) - z_L(x_i)||_2} > 1, \tag{10}$$

Combining (9) and (10) and using the Mean Value Theorem [39], it is justified that a lower Frobenius norm makes a network less sensitive to perturbations, i.e.,

$$\frac{||z_L(x_p) - z_L(x_i)||_2^2}{||x_p - x_i||_2^2} \leq ||J(x')||_F^2, \tag{11}$$

where $x' \epsilon [x_i, x_p]$. Similar to (10), for each layer l, we have

$$\frac{||z_l(x_p) - z_l(x_i)||_2}{||z_l(x_c) - z_l(x_i)||_2} > 1. \tag{12}$$

The misclassification error is propagated at each layer, thus

$$\frac{||z_l(z_{l-1}(x_p)) - z_l(z_{l-1}(x_i))||_2}{||z_l(z_{l-1}(x_c)) - z_l(z_{l-1}(x_i))||_2} > 1. \tag{13}$$

This error is back-propagated and then adjusted at each layer. Therefore the learning optimization process increases robustness of a DNN model trained by QKeras since it discriminates the error due to the clean versus the perturbed input. Consequently, it can be concluded that QKeras deep quantization-aware process introduces per-layer JR with respect to the previous layer's input that is back propagated, as shown in Figure 2 and Step 11 to 15 of Algorithm 1 in [38]. Although back propagation is intended to quantize parameters, it ultimately results in more robust networks. Its learning process increases the training computation time but its advantage is two-fold, i.e. deep quantization and robustness to adversarial perturbations.

B. Stochastic Ternary Quantized QKeras Architecture

As the QKeras [37] API serves as an extension of Keras, initially a 32-bit FP DNN model was built with [36]. The FP model consists of six convolutional layers including depth-wise, separable, and three fully connected layers. The former layers are used for capturing channel-wise correlations and can provide more features with less parameters, particularly with image inputs [40]. A multi-branch topology was used with residual connections to refine the feature maps. For better accuracy, batch normalization [41], ReLU [42] and sigmoid [43] activation functions are considered in the hidden layers, while softmax is used in the output layer. The FP model contains 998,824 parameters, of which 997,460 are trainable and 1,364 are non-trainable, as returned by `model.summary()`. This DNN model is not integrated with JR by default, and is therefore vulnerable to adversarial attacks. However, the model cannot be deployed on tiny MCUs due to its large size (4 MB). Since our target device is the STM32H7, which has a maximal RAM memory of 1 MB, we applied the STQ method to this model using QKeras.

Figure 3 shows the architecture of the STQ model visualized using Netron [44] to render the DNN graph along with its hyperparameters. As we can see, the convolutional, depth-wise, separable layers and activation are appended with *Q*, which indicates the quantization version of the FP Keras [36] model, while the *quantized_relu* represents the quantized activation function versions of Keras. The proposed STQ model used the heterogeneous quantization features of QKeras, which supports independent quantization of each layer in the DNN [10]. This is useful in reducing the model memory footprint and complexity with increased performance accuracy.

IV. Evaluation

This section describes the evaluation procedure of the proposed STQ model, using image and audio datasets for clean and adversarial samples. Moreover, it compares the top-1 accuracy of our STQ model with other benchmarks for three black-box and three white-box adversarial attacks.

Authorized licensed use limited to: City St Georges University of London. Downloaded on June 12,2025 at 18:15:09 UTC from IEEE Xplore. Restrictions apply.

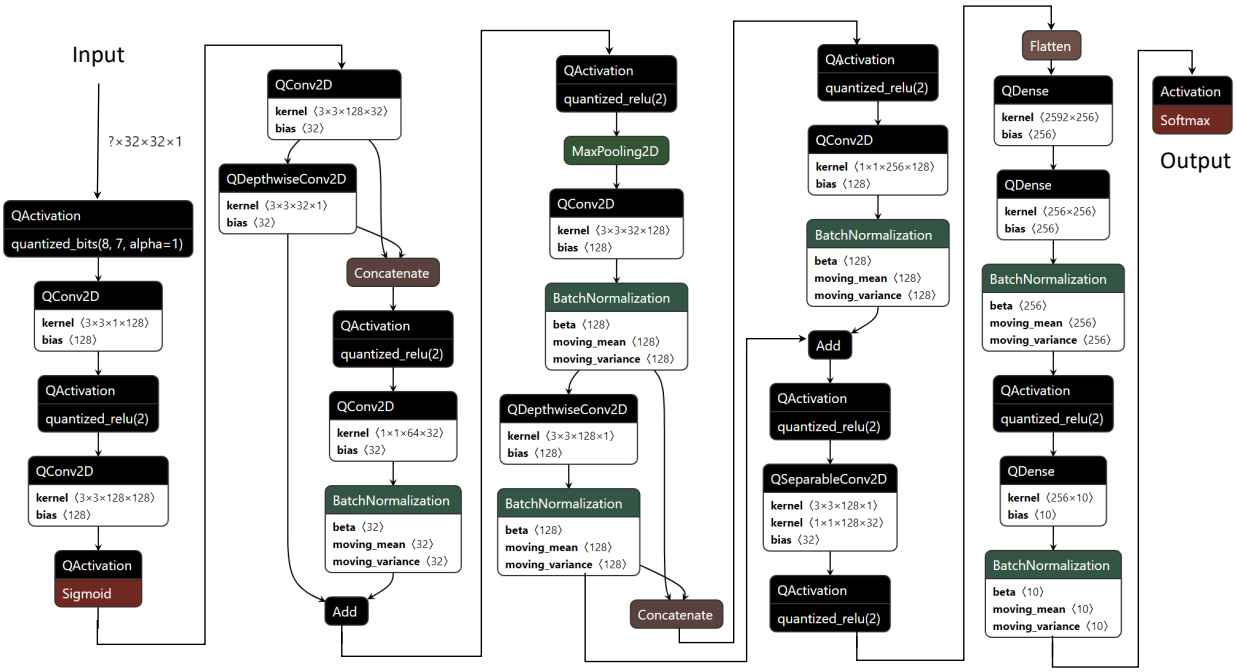


Fig. 3: Proposed architecture of the STQ-based DNN model developed using QKeras.

A. Experimental Setup

1) Datasets and Pre-processing: To evaluate the effectiveness of the devised STQ model, the following audio and image benchmark datasets are considered:

- **CIFAR-10** consists of 60,000 images belonging to 10 different classes. Each class is divided into 50,000 training images and 10,000 test images [45].
- **Street View House Numbers (SVHN)** is a real-world image dataset obtained from house numbers in Google Street View images. It consists of 10 classes, one for each digit. There are 73,257 and 26,032 digits for training and testing respectively [46].
- **Google Speech Commands (GSC)** consists of 65,000 one-second long utterances of 30 short words by thousands of different people. Its 12 classes comprise words of 'yes', 'no' and digits from 'zero' to 'nine' that were used in the experiment. The number of training and test samples are 31,257 and 15,636 respectively [47].

Note that the input image samples were normalized between 0 and 1 pixel and then converted into gray-scale images before presenting them to the DNN input. This effectively reduced the computational cost of processing the image samples while avoiding the use color which are known to be deceitful. Color processing was out of scope by this work and may be considered in future extensions of it.

2) Model Training Procedure: Table II lists the training parameters used to evaluate the full-precision (FP), the proposed STQ and other tested quantized models. A cosine annealing Learning Rate (LR) function [48] was used with the Adamax optimizer for faster convergence. These training parameters were selected to both fine tune each model and reduce its computational complexity, while maintaining better or state-of-the-art performance.

TABLE II: Training parameters for QKeras DNN

Parameter	Value
Epochs	1000
Batch Size	64
Learning Rate	$[1 \times 10^{-6}, 1 \times 10^{-3}]$
LR Scheduler	Cosine Annealing
Loss	Categorical Cross-entropy
Optimizer	Adamax

Since the proposed STQ model targets images and audio datasets, ResNetv1 and DS-CNN TFLite models were also used and tested for comparison purposes. This is to investigate the robustness of the model against industry adopted MLCommons/Tiny models, which can provide insights into the performance capability of the proposed model across various datasets and other benchmarks.

3) Adversarial Attacks Procedure: The proposed STQ model was evaluated against several adversarial attacks to demonstrate its resilience and robustness. Three white-box attacks, FGSM and PGD [18] and Carlini and Wagner (C&W-L2) [49]; and three black-box attacks, Square [19], Boundary attack [20] and Zero Order Optimization (Zoo) [50] were considered. The perturbed data samples for all attacks were generated with the Adversarial Robustness Toolbox (ART) [51] against the tested datasets. For FGSM and PGD samples, an ϵ value of 0.6 with an L1 norm was used. For the Square attack, an ϵ value of 0.6 with *infinity* norm was used, while for the Boundary attack, an ϵ value of 0.01 with *infinity* norm was used. The maximum

Authorized licensed use limited to: City St Georges University of London. Downloaded on June 12,2025 at 18:15:09 UTC from IEEE Xplore. Restrictions apply.

TABLE III: Performance (top-1 accuracy) comparison on CIFAR-10, SVHN and GSC clean datasets.

Model	Flash (KB)	CIFAR-10 Acc. (%)	SVHN Acc. (%)	GSC Acc. (%)
FP	4496	52.73	67.94	89.50
8-bit	1124	52.46	70.04	85.60
4-bit	527	50.88	64.72	82.06
Ternary	410	77.35	93.11	88.30
STQ	410	**80.57**	**95.53**	**94.76**
2-bit	281	46.35	89.74	81.56
Binary	140	39.03	56.91	77.10
S-Binary	140	52.25	63.72	82.51

number of iterations used for C&W-L2 and Zoo is 10, and a binary search tree of 10 used for Zoo. The C&W-L2 and Zoo are used with CIFAR-10 data samples. The number of samples used to create the perturbations were 100, 240 and 2400 for CIFAR-10, SVHN and GSC datasets respectively.

To further examine the strength of our STQ model against FGSM, PGD [18] and Square [19] attacks, they were crafted under different attack strengths using the 10,000 test samples of CIFAR-10. The FGSM attacks were crafted for ϵ ranging in: $\{0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$ as denoted in [14]. Regarding the PGD attack, an iteration $t = 7$ was used along with a step size $\alpha = 2/255$ and an ϵ ranging in: $\{8/255, 16/255, 32/255\}$. For the Square [19] attacks, the first variation consists of *infinity* norm, ϵ value of 0.05 and maximum iterations of 10,000. The second variation of the square attack uses the ℓ_2 norm and maximum iterations of 10,000 as denoted in [52].

4) K-fold Cross Validation: In order to estimate the variance of the clean and adversarial data samples across each tested model, K-fold cross validation was used. This technique splits the entire dataset into K (fold) equal subsets, of which $K-1$ subsets are used for training and the remaining one subset is used for validation [53]. For each fold, the model is fitted on the training set and predicted on the validation set to estimate the average performance. For implementation purposes, $K = 10$ was considered, because a larger number of folds may increase the predictive performance [54]. In addition, the scikit-learn [55] ML Python API was used.

B. Quantization schemes

Table III shows the performance (top-1 test accuracy) comparison between different quantized models against the tested datasets using clean inputs from each dataset. In addition, it includes the FLASH MCU memory of each quantized model computed based on the weight profile obtained using the `qstats` QKeras library. As we can see, the STQ-based model provides the highest accuracy as compared to the FP, Stochastic Binary (S-Binary), and other quantized models. Moreover, this model is lighter than the FP, 8-bit and 4-bit models and can be deployed on MCU devices. These results motivate further investigations on the robustness of the proposed STQ model against adversarial attacks, since the FP model does not have any integrated defensive mechanism, nor better accuracy performance in classifying clean samples.

TABLE IV: Models robustness (top-1 test accuracy) comparison for CIFAR-10, SVHN, and GSC datasets.

Dataset	Model	Clean Acc(%)	FGSM Acc(%)	PGD Acc(%)	Square Acc(%)	Boundary Acc(%)
CIFAR-10	ResNetv1	85.0	65.6	64.1	65.5	82.9
	STQ	80.6	**80.8**	**80.8**	**73.6**	**83.8**
SVHN	ResNetv1	94.8	**94.0**	**94.0**	82.1	94.0
	STQ	**95.5**	93.1	93.1	**91.7**	**97.3**
GSC	DS-CNN	89.1	15.0	15.4	15.1	14.7
	STQ	**94.8**	**94.3**	**94.2**	**94.3**	**95.4**

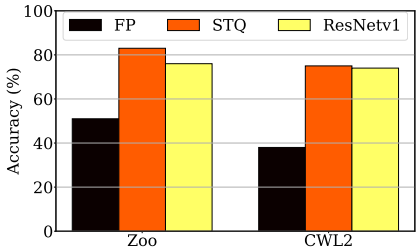


Fig. 4: Models robustness (top-1 test accuracy) comparison against Zoo and C&W-L2 attacks for the CIFAR-10 dataset.

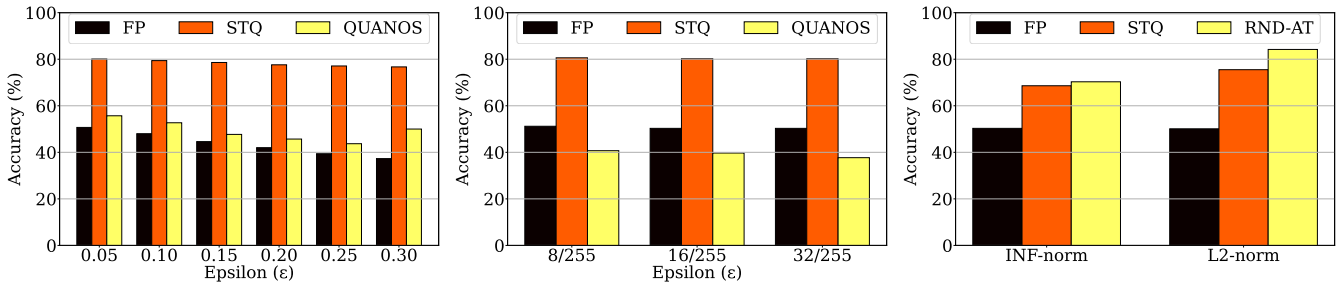
C. Robustness evaluation against adversarial attacks

Table IV shows the robustness (top-1 test accuracy) of our STQ model against two white-box attacks (FGSM, PGD) and two black-box attacks (Square, Boundary). The results are compared with two MLCommons/Tiny benchmarks: ResNetv1 for image classifications trained on CIFAR-10 and SVHN datasets; and DS-CNN TFLite for Keyword Spotting trained on GSC dataset [56]. In addition, Figure 4 shows the robustness against another white-box attack (C&W-L2) and a black-box attack (Zoo). The results are compared with ResNetv1 for image classifications trained on the CIFAR-10 dataset.

As we can see in Table IV, there is a drop in accuracy for the two MLCommons/Tiny benchmarks against all adversarial attacks, as no defense mechanism is included, which makes STQ with integrated JR an interesting scheme. On average, adversarial attacks have caused an accuracy drop of 15.5%, 3.8% and a massive 74.1% with CIFAR-10, SVHN and GSC datasets respectively. Related to the C&W-L2 and Zoo attacks, we can see in Figure 4 that the average drop of ResNetv1 for the CIFAR-10 dataset is 10.5%. All the previous accuracy drops highlight the importance of robust tiny models. Note that Figure 4 also includes the robustness of our FP model, which is clearly lower than STQ.

However, the accuracy of the proposed STQ model was either improved or slightly decreased in the presence of attacks. The largest accuracy drops of 7.0% and 5.6% were observed for the Square and C&W-L2 attacks, respectively, and the CIFAR-10 dataset, although both outperformed the ResNetv1 benchmark. Note that ResNetv1 performs 0.9% better than our STQ model only in the case of FGSM and PGD perturbations for the SVHN dataset. Finally, our model outperforms the DS-CNN model with better accuracy for both clean and adversarial samples of the GSC dataset, for all the experiments.

Authorized licensed use limited to: City St Georges University of London. Downloaded on June 12,2025 at 18:15:09 UTC from IEEE Xplore. Restrictions apply.



(a) FGSM (b) PGD (c) Square

Fig. 5: Robustness comparison of our FP and STQ models against Quanos [14] and RND-AT [52] for various FGSM and PGD perturbation strengths and variations of Square attacks.

These results demonstrate that the resilience and robustness of the proposed STQ model against several adversarial attacks is better than the MLCommons/Tiny benchmarks tested models. As such, the per-layer JR integration of our STQ model has an advantage of reducing model complexity and in parallel enhancing the robustness of the model, producing an effective performance in comparison with traditional benchmark models. This is an interesting finding when looking at the requirements of MCUs, and other tiny devices that are memory and computational resource-constrained.

Table V shows the variance reports of K-fold cross validation, where $K = 10$. The MLC/T column represents ResNetv1 model for CIFAR-10 and SVHN datasets and DS-CNN model for the GSC dataset. The variance of the benchmark for the STQ model is lower in many samples of CIFAR-10 and all instances of the GSC dataset. For the SVHN, the variance of the ResnetV1 model is lower than that of STQ. Therefore, the K-fold cross validation results are varying with datasets and types of attacks, although the overall average results demonstrate general consistency of STQ, with categorical cross entropy loss. Particularly for the GSC dataset, at which STQ tends to outperform the MLC/T with both clean and attacks samples. These results demonstrate the performance capability of STQ as a robust and effective model suitable to be deployed into a resource-constrained environment.

Finally, Figure 5 shows a comparison between our FP and STQ models against: i) QUANOS [14] baseline model for various FGSM and PGD perturbation strengths; and ii) RND-AT [52] for variations of Square [19] attacks. All attacks used 10,000 testing samples of the CIFAR-10 dataset. As we can see, STQ clearly outperforms QUANOS in detecting FGSM and PGD attacks for various strengths, and is slightly worse than RND-AT for Square attacks. We did not find any previous work for the Boundary attack which we could fairly compare against, and we leave Zoo and C&W-L2 for future work.

TABLE V: Variance as a result of K-fold cross validation.

Dataset	Procedure	STQ	MLC/T
CIFAR-10	No Attack	**3.04**	26.25
	FGSM	**5.81**	6.01
	PGD	3.7	3.35
	Square	**11.78**	15.58
	Boundary	**1.4**	5.69
SVHN	No Attack	0.19	0.1
	FGSM	4.27	1.99
	PGD	1.64	1.40
	Square	**4.2**	5.36
	Boundary	**0.39**	0.45
GSC	No Attack	**0.37**	0.80
	FGSM	**1.57**	4.45
	PGD	**3.17**	5.45
	Square	**2.96**	3.64
	Boundary	**0.24**	0.34
	Average	**2.98**	5.39

V. Conclusion

This paper investigated the robustness of a Deeply Quantized Machine Learning (DQML) model against various white-box and black-box adversarial attacks. The deep quantization facilities of QKeras were considered to create a memory optimized, accurate and adversarially robust quantized model. This is due to its similarities with a defense technique, Jacobian Regularization (JR), that was integrated into it. As demonstrated, the proposed Stochastic Ternary Quantized (STQ) model, with quantization-aware training procedure introducing per-layer JR, was more robust than industry adopted MLCommons/Tiny benchmarks when facing several adversarial attacks. In fact, the stochastic quantization scheme was effective for model compression with support for robustness against adversarial attacks. This robustness was experimentally proved by observing its accuracy under various adversarial attacks utilizing two image (CIFAR-10 and SVHN) datasets and one audio (GSC) dataset. This is relevant in the context of deploying efficient and effective models in resource-constrained environments, with limited capabilities. Our initial results suggest further exploration of other sophisticated white-box and black-box attacks with different attack strengths. Future work will further assess the effectiveness of the proposed STQ QKeras model against new and latest attacks.

Acknowledgment

This work has been supported by the PETRAS National Centre of Excellence for IoT Systems Cybersecurity, funded by the UK EPSRC under grant number EP/S035362/1.

Authorized licensed use limited to: City St Georges University of London. Downloaded on June 12,2025 at 18:15:09 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] W. Liu, Z. Wang, X. Liu *et al.*, "A Survey of Deep Neural Network Architectures and their Applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[2] J. Turner, J. Cano, V. Radu *et al.*, "Characterising Across-Stack Optimisations for Deep Convolutional Neural Networks," in *2018 IEEE International Symposium on Workload Characterization (IISWC)*, 2018.

[3] I. Rosenberg, A. Shabtai, Y. Elovici, and L. Rokach, "Adversarial Machine Learning Attacks and Defense Methods in the Cyber Security Domain," *ACM Comput. Surv.*, vol. 54, no. 5, pp. 108:1–108:36, 2021.

[4] Y. Lin, H. Zhao, X. Ma *et al.*, "Adversarial Attacks in Modulation Recognition with Convolutional Neural Networks," *IEEE Trans. Reliab.*, vol. 70, no. 1, pp. 389–401, 2020.

[5] B. Sudharsan, J. G. Breslin, M. Tahir *et al.*, "OTA-TinyML: Over the Air Deployment of TinyML Models and Execution on IoT Devices," *IEEE Internet Comput.*, vol. 26, no. 3, pp. 69–78, 2022.

[6] R. David, J. Duke, A. Jain *et al.*, "TensorFlow Lite Micro: Embedded Machine Learning for TinyML Systems," in *Proc. of the Machine Learning and Systems*, vol. 3, 2021.

[7] L. Geiger and P. Team, "Larq: An Open-Source Library for Training Binarized Neural Networks," *J. Open Source Softw.*, vol. 5, no. 45, p. 1746, 2020.

[8] A. Pappalardo, "Xilinx/brevitas," 2022. [Online]. Available: https://doi.org/10.5281/zenodo.3333552

[9] F. Loro, D. Pau, and V. Tomaselli, "A QKeras Neural Network Zoo for Deeply Quantized Imaging," in *Proc. of the Int. Forum on RTSI*, 2021.

[10] L. N. C. Jr, A. Kuusela, S. Li *et al.*, "Automatic Heterogeneous Quantization of Deep Neural Networks for Low-latency Inference on the Edge for Particle Detectors," *Internet of Things*, 2021.

[11] Y. Dong, R. Ni, J. Li *et al.*, "Learning Accurate Low-Bit Deep Neural Networks with Stochastic Quantization," *arXiv*, 2017.

[12] S. H. Silva and P. Najafirad, "Opportunities and Challenges in Deep Learning Adversarial Robustness: A Survey," *arXiv*, 2020.

[13] J. Lin, C. Gan, and S. Han, "Defensive Quantization: When Efficiency Meets Robustness," *arXiv*, 2019.

[14] P. Panda, "Quanos: Adversarial Noise Sensitivity driven Hybrid Quantization of Neural Networks," in *Proc. of the ACM/IEEE ISPLED*, 2020.

[15] A. K. Sikder, G. Petracca, H. Aksu *et al.*, "A Survey on Sensor-based Threats and Attacks to Smart Devices and Applications," *IEEE Commun. Surv. Tutor.*, vol. 23, no. 2, pp. 1125–1159, 2021.

[16] J. Su, D. V. Vargas, and K. Sakurai, "One Pixel Attack for Fooling Deep Neural Networks," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, 2019.

[17] Y. Jiang, G. Yin, Y. Yuan, and Q. Da, "Project Gradient Descent Adversarial Attack Against Multisource Remote Sensing Image Scene Classification," *Security Commun. Netw.*, vol. 2021, 2021.

[18] T. Huang, V. Menkovski, Y. Pei, and M. Pechenizkiy, "Bridging the Performance Gap between FGSM and PGD Adversarial Training," *arXiv preprint arXiv:2011.05157*, 2020.

[19] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square Attack: A Query-Efficient Black-Box Adversarial Attack via Random Search," in *Proc. of the ECCV*, 2020.

[20] H. Li, X. Xu, X. Zhang *et al.*, "QEBA: Query-Efficient Boundary-based Blackbox Attack," in *Proc. of the IEEE/CVF CVPR*, June 2020.

[21] H. Xu, Y. Ma, H.-C. Liu *et al.*, "Adversarial Attacks and Defenses in Images, Graphs, and Text: A Review," *Int. J. Autom. Comput.*, vol. 17, pp. 151–178, 2020.

[22] G. Cohen, G. Sapiro, and R. Giryes, "Detecting Adversarial Samples Using Influence Functions and Nearest Neighbors," in *Proc. of the IEEE/CVF CVPR*, June 2020, pp. 14 453–14 462.

[23] S. Hussain, P. Neekhara, S. Dubnov *et al.*, "WaveGuard: Understanding and Mitigating Audio Adversarial Examples," in *Proc. of the 30th USENIX Security Symposium*, 2021.

[24] R. Bernhard, P.-A. Moëllic, and J.-M. Dutertre, "Luring Transferable Adversarial Perturbations for Deep Neural Networks," in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–8.

[25] R. A. Khamis and A. Matrawy, "Evaluation of Adversarial Training on Different Types of Neural Networks in Deep Learning-based IDSs," in *Proc. of the ISNCC*. IEEE, 2020, pp. 1–6.

[26] M. Picot, F. Messina, M. Boudiaf *et al.*, "Adversarial Robustness via Fisher-rao Regularization," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022.

[27] D. Jakubovitz and R. Giryes, "Improving DNN Robustness to Adversarial Attacks using Jacobian Regularization," in *Proc. of the ECCV*, September 2018, pp. 1–16.

[28] J. Sokolić, R. Giryes, G. Sapiro, and M. R. D. Rodrigues, "Robust Large Margin Deep Neural Networks," *IEEE Trans. Signal Processing*, vol. 65, no. 16, pp. 4265–4280, 2017.

[29] A. Gholami, S. Kim, Z. Dong *et al.*, "A Survey of Quantization Methods for Efficient Neural Network Inference," *arXiv:2103.13630*, 2021.

[30] D. Pau, A. Khiari, and D. Denaro, "Online Learning on Tiny Microcontrollers for Anomaly Detection in Water Distribution Systems," in *Proc. of the ICCE*, vol. 11. IEEE, 2021, pp. 1–6.

[31] C. Song, E. Fallon, and H. Li, "Improving Adversarial Robustness in Weight-quantized Neural Networks," *arXiv e-prints*, 2020.

[32] A. Galloway, G. W. Taylor, and M. Moussa, "Attacking Binarized Neural Networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[33] R. Bernhard, P.-A. Moellic, and J.-M. Dutertre, "Impact of Low-bitwidth Quantization on the Adversarial Robustness for Embedded Neural Networks," in *Proc. of the Int. Conference on Cyberworlds*, 2019.

[34] Y. Kim, J. Lee, Y. Kim, and J. Seo, "Robust Quantization of Deep Neural Networks," in *Proc. of the 29th International Conference on Compiler Construction*, 2020, pp. 74–84.

[35] M. Gorsline, J. Smith, and C. Merkel, "On the Adversarial Robustness of Quantized Neural Networks," in *Proc. of the GLSV*, 2021, pp. 189–194.

[36] Google, "Keras Framework: Python Deep Learning API," 2017. [Online]. Available: https://keras.io/

[37] ——, "QKeras: Quantization Extension to Keras," 2017. [Online]. Available: https://github.com/google/qkeras

[38] S. Zhou, Y. Wu, Z. Ni *et al.*, "DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients," *arXiv preprint*, vol. arXiv:1606.06160, 2018.

[39] E. Shishkina, "Mean-Value Theorem for B-Harmonic Functions," *Lobachevskii J. Math.*, vol. 43, no. 6, pp. 1401–1407, 2022.

[40] Y. Guo, Y. Li, L. Wang, and T. Rosing, "Depthwise Convolution is all you need for Learning Multiple Visual Domains," in *Proc. of the AAAI Conference on Artificial Intelligence*, 2019.

[41] E. Wang, J. J. Davis, D. Moro *et al.*, "Enabling Binary Neural Network Training on the Edge," in *Proc. of the 5th Int. Workshop on EMDL*, 2021.

[42] K. Eckle and J. Schmidt-Hieber, "A Comparison of Deep Networks with ReLU Activation Function and Linear Spline-type Methods," *Neural Netw.*, vol. 110, pp. 232–242, 2019.

[43] N. Papernot, A. Thakurta, S. Song *et al.*, "Tempered Sigmoid Activations for Deep Learning with Differential Privacy," in *Proc. of the AAAI Conference on Artificial Intelligence*, 2021.

[44] L. Roeder, "Netron," 2022. [Online]. Available: https://netron.app/

[45] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," University of Toronto, Toronto, ON, Canada, Tech. Rep., 2009.

[46] Y. Netzer, T. Wang, A. Coates *et al.*, "Reading Digits in Natural Images with Unsupervised Feature Learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[47] G. S. Commands, "Google's Speech Commands Dataset," 2016. [Online]. Available: https://pyroomacoustics.readthedocs.io/en/pypi-release/pyroomacoustics.datasets.google_speech_commands.html

[48] I. Loshchilov and F. Hutter, "SGDR: Stochastic Gradient Descent with Warm Restarts," in *ICLR*, 2017.

[49] N. Carlini and D. Wagner, "Towards Evaluating the Robustness of Neural Networks," in *2017 IEEE Symposium on Security and Privacy (SP)*, 2017.

[50] P.-Y. Chen, H. Zhang, Y. Sharma *et al.*, "Zoo: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models," in *Proceedings of the 10th ACM workshop on artificial intelligence and security*, 2017, pp. 15–26.

[51] M.-I. Nicolae, M. Sinn, M. N. Tran *et al.*, "Adversarial Robustness Toolbox v1. 0.0," *arXiv preprint arXiv:1807.01069*, 2018.

[52] Z. Qin, Y. Fan, H. Zha, and B. Wu, "Random Noise Defense Against Query-based Black-box Attacks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 7650–7663, 2021.

[53] D. Berrar, "Cross-Validation," in *Encyclopedia of Bioinformatics and Computational Biology*, 2019, pp. 542–545.

[54] T.-T. Wong and P.-Y. Yeh, "Reliable Accuracy Estimates from k-fold Cross Validation," *IEEE Trans. Knowl.*, vol. 32, no. 8, pp. 1586–1594, 2019.

[55] "Scikit-learn: Machine Learning Python API," 2019. [Online]. Available: https://scikit-learn.org/stable/

[56] C. Banbury, V. J. Reddi, P. Torelli *et al.*, "MLPerf Tiny Benchmark," vol. arXiv:2106.07597, 2021.

Authorized licensed use limited to: City St Georges University of London. Downloaded on June 12,2025 at 18:15:09 UTC from IEEE Xplore. Restrictions apply.