Solar Hotspot Detection Using VHDL-Simulated Fixed-Point SVM: A Methodology Toward FPGA Realization

Nayomi Fernando[a], Lasantha Seneviratne[b], Nisal Weerasinghe[c], Namal Rathnayake[d]* and Yukinobu Hoshino[e]

Department of Electrical and Electronic Engineering, Faculty of Engineering, Sri Lanka Institute of Information Technology, Malabe [a,b,c] , Advanced Institute for Marine Ecosystem Change (WPI-AIMEC), Japan Agency for Marine-Earth Science and Technology (JAMSTEC), Yokohama 236-0001[d] , School of Systems Engineering, Kochi University of Technology, 185 Miyanokuchi, Tosayamada, Kami 782-8502, Kochi, Japan[e]
nayomi.f@sliit.lk, lasantha.s@sliit.lk, nisal.w@sliit.lk, namalr@jamstec.go.jp, hoshino.yukinobu@kochi-tech.ac.jp

ABSTRACT

Early detection of thermal hotspots in photovoltaic modules is critical to ensuring their efficiency, safety, and longevity. This study presents a complete end-to-end methodology for implementing a fixed-point Medium Gaussian Support Vector Machine classifier using VHDL for a Field Programmable Logic Array. The approach begins with feature extraction from thermal images of healthy and defective solar panels, which focuses on MPEG-7 descriptors. The study shows that high impact for hotspot detection comes from blue chrominance contrast. A medium Gaussian SVM model is trained in MATLAB and converted to a fixed-point Q1.15 format for hardware compatibility. Key parameters, including support vectors, Lagrange multipliers, bias, and kernel scale, are extracted and verified in a custom Python environment to ensure numerical alignment with MATLAB results. The validated model is then implemented in synthesizable VHDL. It is verified using GHDL and the GNU Tool Kit waveform viewer, confirming bit-accurate hardware behaviour. Results show classification accuracy exceeding 99.3% with negligible performance loss due to quantization. The design achieves deterministic latency through an FSM-based structure and parallel feature processing for a 300-support vector and 222-feature system. This method enables low-power, real-time inference on a UAV-based edge platform, primarily focusing on drones.

CCS CONCEPTS Hardware ~ Electronic design automation ~ Logic synthesis; Computing methodologies ~ Machine learning ~ Learning paradigms ~ Supervised learning; Hardware ~ Integrated circuits ~ Reconfigurable logic and FPGAs; Computer systems organization ~ Real-time systems

KEYWORDS: *Deep Learning, FPGA, Machine Learning, MPEG-7, Solar PV, SVM, VHDL*

1 INTRODUCTION

Global demand for renewable energy has driven rapid growth in the deployment of solar photovoltaic (PV) systems, both in large-scale commercial solar farms and in domestic rooftop installations. However, maintaining their efficiency over time is not always straightforward. One major issue that frequently arises in PV panels is the formation of thermal "Hotspots"[1]. Hotspots are localized regions where temperatures exceed those of the surrounding area for several reasons. Factors contributing to hotspot occurrence include partial shading, dust accumulation, cell mismatch, and microcrack defects. In practice, hotspots not only reduce power output but also accelerate material degradation, eventually causing permanent damage to the PV modules. Anomaly detection using thermal signatures in PV modules is advancing with the development of thermal imaging, computer vision, and Machine Learning (ML).

This makes them particularly useful for image-based inspection tasks. The recent study [1], which extracts handcrafted features and trains a Medium Gaussian Support Vector Machine (SVM),

* Corresponding author.

https://doi.org/10.54389/FNSB7045

demonstrates strong performance on drone-captured thermal images. Further, among the evaluated features, the blue chrominance contrast extracted from MPEG-7 feature descriptors has a substantial impact on solar hotspot identification. The model achieves 99.3% accuracy with an inference time of 18 seconds. This demonstrates that this model maintains a balance between complexity and computational cost. However, the challenging aspect of the workflow arises in hardware implementation. This is because these models, typically developed in MATLAB, rely on floating-point operations and high-level abstractions. As a result of these characteristics, which make them less suitable for low-power, real-time edge deployment, this raises the challenge of optimizing the model for drone platforms. Therefore, we identified a clear need for hardware-efficient solutions for model implementation on a UAV platform. Considering the factors such as parallel processing, predictable timing, and relatively low power consumption, we identified that Field Programmable Gate Arrays (FPGAs) are well-suited for this application.

However, this is not a direct process for converting a high-level ML model into an FPGA-compatible design. The first significant challenge encountered is replacing floating-point computations with fixed-point representations to achieve efficient synthesis with minimal resource utilisation. In this work, we propose a complete fixed-point implementation of the SVM model developed in the Q1.15 format, providing a practical trade-off between numerical precision and hardware efficiency. The approach is initiated by extracting key parameters of the MATLAB-trained model, namely support vectors, Lagrange multipliers, class labels, kernel scale, and bias. Next, these parameters are evaluated using a Python-based fixed-point simulation, and numerical consistency is checked before proceeding with the hardware implementation. The model is described in synthesizable VHDL using Q1.15 arithmetic after validation. The VHDL design is structured into three main parts: SVM decision logic, constants package, and test bench. GHDL for logic-level execution performs simulation, while GTKWave examines the waveform behaviour. This allows for a closer inspection of timing and state transitions, confirming that the decision function behaves as expected. The design is synthesised to integrate easily into FPGA toolchains such as Intel Quartus or Xilinx Vivado for future hardware implementation.

This study, which is an extension of the previous work [1], focuses on hardware-oriented implementation. The main contribution is that we integrate previous work on MPEG-7 feature extraction using the Automatic Content Extractor (ACE), followed by Medium Gaussian SVM-based classification, and synthesise the resulting FPGA-ready VHDL. To utilise this methodology, we convert the trained SVM parameters to the Q1.15 fixed-point format to reduce FPGA resource usage without significantly reducing accuracy. The second contribution is the verification of the VHDL implementation to ensure bit-level agreement with MATLAB predictions, resulting in a design ready for FPGA synthesis.

The paper is structured as follows. The literature reviews the recent work on solar hotspot detection along with feature extraction and FPGA-based classification methods. The methodology outlines the proposed approach, including SVM training in MATLAB, fixed-point conversion, and VHDL implementation. The results section presents the simulation outcomes, and the discussion section analyses performance, hardware efficiency, and suitability for real-time UAV deployment. The paper concludes by summarising the findings for an implementation-ready design and outlining possible directions for future work in real-world system integration.

2 LITERATURE REVIEW

The integration of infrared (IR) thermography, ML algorithms, and embedded hardware platforms results in thermal hotspot detection. This section delves into the state-of-the-art literature across three main domains. They are image-based hotspot detection techniques, the application of Support Vector Machines for PV fault classification, and FPGA-based implementation of ML algorithms, respectively. This section highlights the evolution of anomaly-detection methodologies

driven by new technological advancements. It identifies the key limitations in previous related studies that motivate the proposed pipeline.

1.1 Image-Based Hotspot Detection in PV Panels

Conventional photovoltaic system monitoring has relied on irregularities in electrical parameters such as voltage and current. However, the drawback is that the studies [2-3] reveal these approaches fail to detect anomalies at early stages. The use of infrared thermography from drone-mounted platforms primarily facilitates the identification of localised thermal variations. This signals the primary indication of potential hotspots [4], [5].

The study in [1] proposes a framework combining MPEG-7 texture descriptors, such as edge histograms, with colour-based features to identify fault regions in PV panels. The experiment, conducted across five solar PV datasets, demonstrated that the Medium Gaussian SVM outperformed other classifiers, such as Random Forests and Binary GLM (Generalised Linear Model) Logistic Regression, in terms of classification accuracy and robustness to feature variation. This paper serves as a basis for identifying discriminative features in thermal images, and its findings motivate the use of Medium Gaussian SVM as a core classifier in hardware deployment scenarios [1].

1.2 Support Vector Machines for Fault Detection

Support Vector Machines have been widely favoured for classification tasks where feature space dimensionality is high, and nonlinearity is prevalent [6]. Their ability to define optimal hyperplanes with maximal margin allows them to generalise well, even on limited datasets. The Radial Basis Function (RBF) or Gaussian kernel used in SVMs adds further flexibility by allowing the mapping of inputs into higher-dimensional spaces, where linearly inseparable data become separable [7].

A wide range of works have utilized SVMs in PV diagnostics. For example, study [6] employed an SVM for string-level fault detection in PV arrays using electrical signatures. Recent approaches utilise thermal image features with machine learning models, such as SVM, for fault detection, as stated in the study [9]. Furthermore, enhanced real-time applicability for edge-based systems has been demonstrated by FPGA-accelerated implementations of lightweight deep learning models [8]. While software implementations are widely studied, the hardware integration of SVM-based models remains relatively unexplored due to challenges in representing floating-point operations and nonlinear kernel functions [10], [11].

1.3 FPGA-Based Machine Learning and Fixed-Point Design

FPGAs offer a reconfigurable, power-efficient platform for deploying ML inference models. Their parallel processing capabilities make them suitable for latency-sensitive applications such as real-time image classification [12]. However, due to hardware constraints, floating-point arithmetic is avoided in favour of fixed-point implementations that enable fast logic synthesis, lower resource usage, and faster execution [13].

Existing literature shows that deploying ML on an FPGA requires either of the following. One is high-level synthesis tools like Xilinx Vitis or Intel HLS, which abstract HDL generation from C/C++ code [14]. The other is manual HDL coding with fixed-point models, which offer better control and efficiency [10].

The study [14] proposed a fixed-point neural network implementation on an FPGA for object detection, showing that quantisation has the least impact on accuracy while drastically improving performance metrics. Similarly, research [13] implemented a hardware-efficient kernel SVM classifier for gesture recognition using VHDL, successfully validating the feasibility of kernel-based models on hardware.

Despite these advances, few studies have focused on deploying SVMs with Gaussian kernels, especially for solar hotspot detection using VHDL implementations. The complexity of exponential function evaluation and vector distance computations poses challenges for hardware implementation.

This identified gap is addressed in this proposed work by translating a MATLAB-trained Medium Gaussian SVM into a fixed-point, synthesizable VHDL model. The design is simulated using GHDL, and its functional correctness is verified through waveform analysis in GTKWave. This simulation step ensures that the algorithm behaves identically to its floating-point MATLAB counterpart before it is synthesised for real-time FPGA deployment [10], [19].

1.4 Significance of the Study

The reviewed literature confirms that most PV monitoring solutions rely on software-based ML models with limited focus on edge deployment. While studies highlight the motivation and complexity of implementing Gaussian kernel SVMs in hardware [15], [16], existing works on DNN and SVM hardware flows [17], [18] lack a complete fixed-point MATLAB-to-VHDL pipeline. Furthermore, there is a clear gap in the implementation of fixed-point, nonlinear SVMs on FPGA platforms [19]. By addressing these gaps, the proposed work contributes a novel end-to-end approach that combines ML with hardware readiness, facilitating an energy-efficient, real-time PV health-monitoring system.

In summary, the literature demonstrates substantial progress in image-driven PV fault detection using SVM classifiers, particularly with Gaussian kernels. While many software-based approaches have shown promising results, limited attention has been paid to fixed-point SVM implementations on an FPGA for real-time applications. The reviewed studies emphasise both the potential and the challenges of hardware implementation for nonlinear models. This justifies the current research focus on translating a MATLAB-trained Medium Gaussian SVM model into a VHDL-based implementation. This bridges the gap between ML accuracy and the feasibility of embedded hardware for PV monitoring systems.

2 METHODOLOGY

This section presents the design and validation process for a synthesis-ready Medium Gaussian SVM classifier. The classifier is developed for real-time solar hotspot detection on FPGA-based hardware implementation. This approach includes simulation and pre-synthesis verification. Aligned with the feature selection rationale in the literature, the workflow starts with thermal-image feature extraction using MPEG-7 descriptors and blue chrominance (YCbCr), followed by model training in MATLAB. This process includes fixed-point conversion, VHDL development, and simulation using GHDL. The main design steps are floating-to-fixed-point adaptation in Q1.15, Finite-State Machine (FSM)- based control logic, and a hardware-friendly Gaussian kernel approximation. This shows the mechanism for translating high-level ML models into synthesizable hardware architectures suitable for UAV-based PV panel monitoring. The pipeline explicitly extracts SVM parameters, namely support vectors, Lagrange multipliers, labels, bias, and kernel scale. Further, this validates a fixed-point MATLAB reference and implements dot products and kernel logic in VHDL, with cycle-accurate verification in GHDL. Furthermore, GTKWave ensures bit-accurate parity before FPGA synthesis.

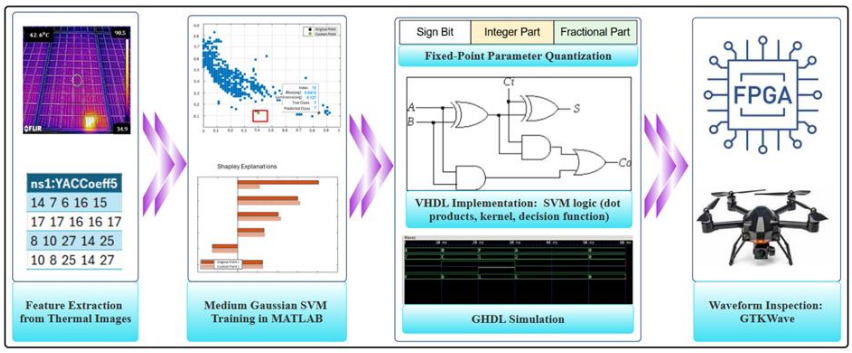


Figure 1. End-to-end workflow for implementing the Medium Gaussian SVM classifier on FPGA, which starts from feature extraction and training in MATLAB through fixed-point parameter quantisation, VHDL coding, simulation with GHDL, and leads to FPGA synthesis and deployment-ready integration on UAV platforms for real-time hotspot detection

The complete workflow for implementing the Medium Gaussian SVM classifier in an FPGA-compatible format for real-time PV hotspot detection is illustrated in Figure 1. The design begins with feature extraction from thermal images using YCbCr descriptors and training a Medium Gaussian SVM model in MATLAB. Then we explicitly extract support vector coefficients, labels, Lagrange multipliers, bias, and kernel scale from the trained model. A custom MATLAB implementation of the SVM decision function is then built to replicate the prediction behaviour, ensuring an exact software reference. These parameters and computations are converted to fixed-point Q1.15 format. The format includes an explicit sign, an integer, and a fractional part for hardware compatibility. The SVM logic is implemented in VHDL. It includes dot products, Gaussian kernel approximation, and decision, followed by cycle-accurate simulation in GHDL. This generates waveforms for GTKWave inspection to confirm bit-accurate agreement with MATLAB. The verified design is delivered as a synthesis-ready module, compatible with standard FPGA synthesis toolchains such as Intel Quartus or Xilinx Vivado, for future deployment on UAV platforms.

The subsequent discussion builds on these stages. It provides a detailed explanation of how MATLAB-SVM models are bridged to synthesizable VHDL, of logic gate-level simulation for solar hotspot classification, and of the implementation of the Gaussian kernel and decision logic intended for FPGA deployment.

Bridging MATLAB SVM Models to Synthesizable VHDL from Floating-Point SVM to Fixed-Point Hardware Friendly Implementation

Built-in SVM functions, such as *fitcsvm* and *predict*, in MATLAB use floating-point arithmetic and object-oriented syntax. It makes them incompatible with the HDL Coder and the FPGA pipeline. To overcome this gap, we implement a custom SVM decision function in MATLAB using extracted parameters. The extracted parameters are alpha, gamma, bias, and support vectors. This allows conversion to fixed-point using *the fi()* function in MATLAB, enabling hardware-friendly computation. The manual approach provides precise control over word length, precision, and overflow behaviour, while supporting debugging, validation, and matching with the software model. It also simplifies the creation of HDL test benches and serves as a bridge toward synthesizable VHDL modules for solar PV hotspot detection.

2.1 Fixed-Point Arithmetic for FPGA Compatibility

FPGAs lack native support for floating-point arithmetic. Moreover, the usage of floating-point IP cores introduces significant resources and latency overhead. Therefore, the enhanced design uses a fixed-point representation, expressing numbers as scaled integers, overcoming this challenge. This procedure reduces computational complexity. It aligns with FPGA architecture, making it the preferred choice for efficient, real-time machine learning implementations on FPGAs.

FPGA hardware compatibility is ensured by adopting a fixed-point format of Q1.15 to maintain numerical precision. There, the total word length is 16 bits comprising 1 sign bit, 1 integer bit, and 15 fractional bits. This configuration allows representation of values in the approximate range of −1 to 0.99997 with a high precision of 0.00003. Among standard 16-bit fixed-point formats such as Q4.11 or Q8.8, Q1.15 is preferred when high precision is critical, and the dynamic range of values is limited, as is frequently the case in embedded AI applications like solar hotspot classification.

2.2 Logic Gate-Level Simulation for Solar Hotspot Classification

To evaluate the feasibility of real-time deployment on an Unmanned Aerial Vehicle (UAV) platform, the trained SVM model was translated into a hardware-compatible version using VHDL. The logic implements binary classification to detect the presence of solar panel hotspots (Class 1) or their absence (Class 0) from MPEG-7 feature vectors. The VHDL module is structured around a three-state FSM: IDLE, COMPUTE, and FINISH, and tested using a testbench.

In the IDLE state, the control signals, index counters (i, j), and accumulator registers are initialised, and the system waits to assert the start signal. Once the start signal is activated, the FSM transitions to the COMPUTE state. It iteratively calculates the squared Euclidean distance between the normalised input feature vector and each stored support vector using fixed-point signed arithmetic. This distance value is used in a hardware-friendly Gaussian RBF kernel approximation implemented as a rational function to avoid floating-point operations. The resulting kernel values are multiplied by the corresponding Lagrange multipliers (α_i) and class labels $(label_i)$, and the products are accumulated into a decision variable (fx_{accum}) according to $fx_{accum}{+}\alpha_i * label_i * kernel_i$. Upon completing all iterations, the FSM enters the FINISH state. The bias term b is added to the accumulated decision value. Then the sign of the result is evaluated. A binary prediction is generated on the predicted signal. All state transitions are synchronised to the rising edge of the clock signal. This ensures deterministic and cycle-accurate execution.

2.3 Fixed-Point Arithmetic and Kernel Logic

The use of Q1.15 signed fixed-point representation avoids the overhead of floating-point units. This makes the implementation resource-efficient for FPGA targets. Accumulators and kernel computations are implemented using a `signed 47 downto 0` format.

SVM Decision Function Gaussian (RBF) Kernel: The classifier computes the decision function as:

$$\boldsymbol{f(x) = \sum_{i=1}^{N} \alpha_i . y_i . K(x_i . x) + b} \qquad \textbf{(1)}$$

where x_i: support vector, y_i: label of support vector, α_i: Lagrange multiplier, b: bias

$$K(x_i . x) = \exp\left(-\gamma . \left|\left|x - x_i\right|\right|^2\right) \qquad (2)$$

where γ: kernel scaling factor, $K(x_i . x)$: Gaussian kernel value between the input vector x and the i^{th} support vector (x_i), approximated for hardware implementation.

It is required to normalise the input feature vector x such that $x' = \frac{x-\mu}{\sigma}$ by Z-score normalisation, where x': Standardised value, x: Original data value, μ: Mean of the data, σ: Standard deviation of the data. Support vectors, labels, and Lagrange multipliers are essential in computing the kernel sum. γ defines the kernel function shape. μ and σ must be used to preprocess input features before classification.

Distance squared:

$$dist_{sq} = \sum_{j=0}^{N} (x_j - sv_{i,j})^2 \quad (3)$$

is calculated in pipelined j iterations. $dist_{sq}$ is the Squared Euclidean distance between the input feature vector x and $the\ i^{th}$ support vector. x_j is the j^{th} element of the normalised input feature vector. $sv_{i,j}$ is $the\ j^{th}$ feature of the ith support vector in the trained SVM model. N is the total number of

features in the input feature dimension vector. i is the index of the current support vector being processed, and j is the index of the current feature element in the input vector.

Gaussian kernel approximation:

$$K(x, x_i) \approx \frac{10}{1+\gamma.dist_{sq}} \qquad (4)$$

where $K(x_i.x)$: Gaussian kernel value is implemented in the exponential approximation function using division-based rational approximation, suitable for hardware. $dist_{sq}$ is the Squared Euclidean distance, and γ is the kernel scaling factor. Decision value is accumulated as in equation (1) across all support vectors and compared to a fixed threshold for a binary decision.

The final stage of the workflow is hardware simulation and validation. This is to ensure deployment readiness. We have implemented the fixed-point VHDL of the SVM classifier, and simulated it using GHDL, with the signal waveforms analysed in GTKWave. This verifies correct state transitions, timing behaviour, and numerical equivalence with the MATLAB floating-point model. The validation confirms that the FPGA design reliably performs real-time solar hotspot classification while allowing integration into UAV-based PV inspection systems.

Overall, the methodology bridges the gap between MATLAB-trained SVM classifiers and their FPGA implementation through a structured process that includes parameter extraction, fixed-point conversion to Q1.15 format, and custom VHDL coding. The classifier is implemented as a state machine-based logic module. It performs kernel evaluations and accumulates decision values using resource-efficient arithmetic. Simulation results validate the equivalence of fixed-point hardware behaviour with the original floating-point mode. It ensures deployment readiness for embedded solar hotspot detection applications.

3 RESULTS

This section assesses the experimental evaluation of the Medium Gaussian SVM model trained on MPEG-7 feature descriptors for detecting thermal anomalies in PV modules. We have tested the model on five datasets of varying sizes and image quality to understand better how its performance scales with dataset size [1]. Performance is evaluated based on accuracy, F1 score, precision, recall, and execution time. A logic-level hardware simulation of the fixed-point SVM classifier is performed to assess its feasibility for FPGA implementation in real-time UAV-based PV inspection systems. We present the findings across five focused subsections. The first is the multi-metric evaluation across datasets used to assess how consistently the model performs at different dataset sizes. Second, the execution time of the models during the training and testing phases has been analysed to determine whether the approach supports real-time deployment. Next, the precision-recall tradeoff and overall model consistency are examined to understand better, the balance between false positives and false negatives in hotspot detection. After that, a logic-gate-level analysis of the SVM classifier was performed to ensure that the software model is correctly translated into hardware. Finally, the simulation results are analysed using timing waveforms and state transitions. This ensures the classifier operates as expected in an FPGA-based environment.

3.1 Multi-Metric Evaluation Across Datasets

Table 1 summarises the training and testing performance of the Medium Gaussian SVM classifier across five datasets [1]. The classifier maintained high accuracy and generalisation throughout the five datasets.

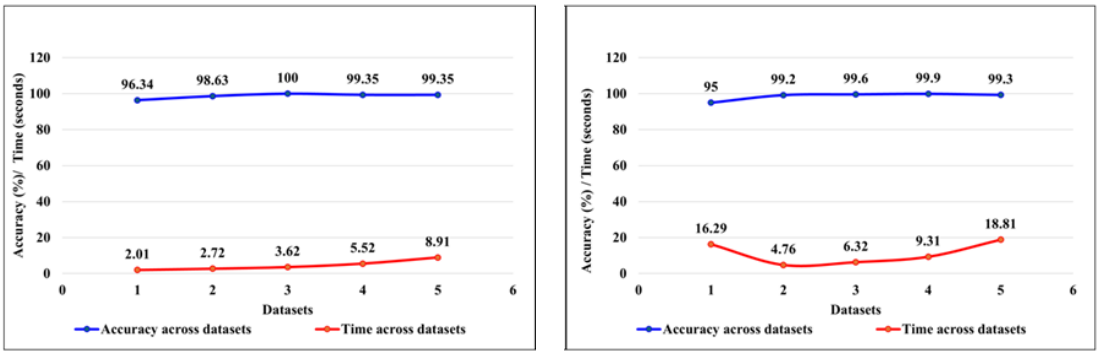
Table 1: Training and Testing Performance across Multiple Datasets

Phase	Dataset Size	Time (seconds)	Accuracy (%)	F1-Score	Precision	Recall
Training						
Dataset 1	724	16.29	95.00	0.9490	0.9570	0.9419
Dataset 2	1302	4.76	99.20	0.9920	0.9908	0.9938
Dataset 3	1836	6.32	99.60	0.9950	1.0000	0.9920
Dataset 4	2746	9.31	99.90	0.9984	1.0000	0.9970
Dataset 5	4114	18.81	99.30	0.9920	0.9920	0.9930
Testing						
Dataset 1	82	2.01	96.34	0.9640	0.9520	0.9760
Dataset 2	146	2.72	98.63	0.9860	0.9860	0.9860
Dataset 3	204	3.62	100.00	1.0000	1.0000	1.0000
Dataset 4	306	5.52	99.35	0.9930	1.0000	0.9870
Dataset 5	458	8.91	99.35	0.9930	0.9910	0.9960

It is observed that the Medium Gaussian SVM achieved up to 100% testing accuracy and F1-scores greater than 0.99 for larger datasets. It demonstrates the model's robustness and generalisation ability.

3.2 Execution Time Analysis

The total inference runtime assesses deployment feasibility. Although the dataset sizes have been increased, the inference times have remained within practical limits. This makes the model appropriate for onboard FPGA-based inference in real-time UAV inspections.



(a) (b)

Figure 2. Dataset-wise tradeoff between accuracy and time (a) training, (b)testing

As illustrated in Figure 2(a), training time increases gradually as the dataset size grows. However, it remains under 20 seconds for the most extensive dataset as well. This indicates efficient scalability during training. Similarly, as illustrated in Figure 2(b), testing time increased linearly with dataset size, demonstrating a predictable computational load. It should be noted that no significant tradeoff was observed between computational speed and classification accuracy. The model maintained high performance across all five datasets, with no drop in efficiency. The scalability and low computational overhead make the fixed-point SVM model feasible for real-time solar hotspot detection.

3.3 Precision-Recall Tradeoff and Model Alignment

As observed, despite variations in the dataset, the model maintained a high balance between precision and recall. The precision is consistently greater than 95.00% with minimal false positives. Recall up to 99.70%, indicating rare cases of missed detection. F1-score peaked at 1.0000 on dataset 3 with a perfect harmonic mean between precision and recall.

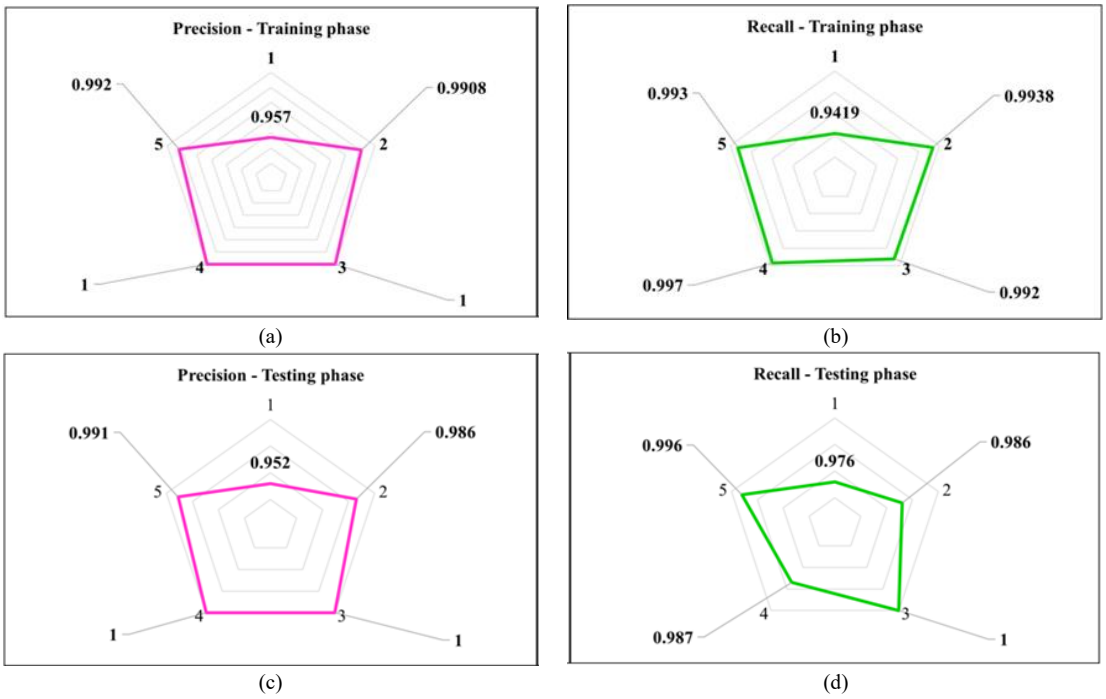


Figure 3. Radar plot of precision and recall across five datasets (a) and (b) training, (c) and (d) testing

Figure 3(a) and (b) show the radar plots of precision and recall for the training phase across five datasets. As the dataset size increases, the training precision generally improves. This can be seen in dataset 1, where 0.9570, reaching 1.0000 in datasets 3 and 4. This indicates that false positives have not been observed. We can see a similar trend in recall, which increases from 0.9419 in dataset 1 to 0.9970 in dataset 4. This improvement is reflected in the F1-score and overall accuracy for dataset 4 as well. As expected, training time increases with dataset size, with the most extensive dataset taking 18.81 seconds.

Figure 3(c) and (d) present the precision and recall during the testing phase. The model maintains strong performance even with relatively small datasets, suggesting good generalisation capability. Precision values range from 0.9520 to 1.0000, while recall ranges from 0.9760 to 1.0000. The best performance is observed for dataset 3, where the model achieves perfect accuracy and an ideal F1-score, with no misclassifications. Datasets 4 and 5 also show consistent results, with high precision (0.9910) and recall (0.9960). These observations indicate the performance reliability of the fixed-point SVM model.

In summary, Q1.15 fixed-point Medium Gaussian SVM gives high detection accuracy. Furthermore, it maintains low execution time across datasets of multiple sizes. MPEG-7 feature descriptor embedding further improves the model's ability to distinguish hotspot regions. These results support the feasibility of deploying the model on FPGA platforms for real-time onboard PV inspection.

3.4 Logic Gate-Level Analysis and Simulation of SVM Classifier

The trained Medium Gaussian SVM model is converted into a deployable embedded form. It is designed for use in UAV edge devices. A hardware-level implementation is developed using VHDL.

The module *svm_core.vhd* simulates the classification process. This is based on finite-state control logic. Furthermore, the design relies on signed fixed-point arithmetic.

The binary classifier of solar hotspot detection combines arithmetic operations with state-machine-based control. At its core, the architecture is constructed from basic logic gate components. They are addition, subtraction, squaring, and multiplication, all facilitated using an FSM with states IDLE, COMPUTE, and FINISH. The inner product calculation, which serves as the kernel distance, is performed using repeated subtraction and squaring. The accumulation of weighted kernel contributions follows it. The decision logic uses a final comparison gate to determine whether the output exceeds the bias threshold that assigns the prediction class. The design computes the RBF kernel, ensuring efficient SVM evaluation without the memory overhead of precomputed exponential values.

3.5 Insights from Simulation

The testbench mimics real-world activation by applying start and reset signals. Simulation output provides visibility into internal signals via report statements.

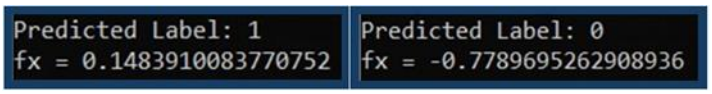


Figure 4. Prediction 1 and 0

Figure 4 shows that the model correctly identifies a hot spot = '1' and '0' when fx_{total} exceeds the threshold. This aligns with expectations from software SVM classification using the same support vectors. Figure 5 and Figure 6 represent the respective waveforms from the GTKWave interface.

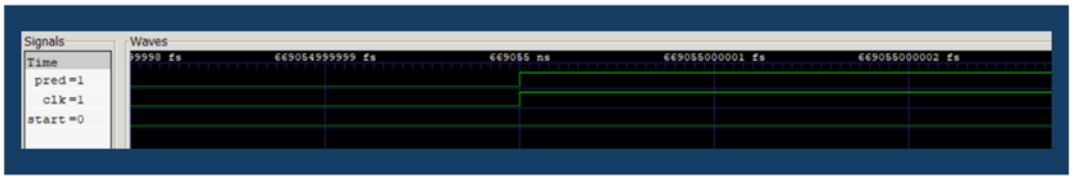


Figure 5. GTK View Output Prediction 1

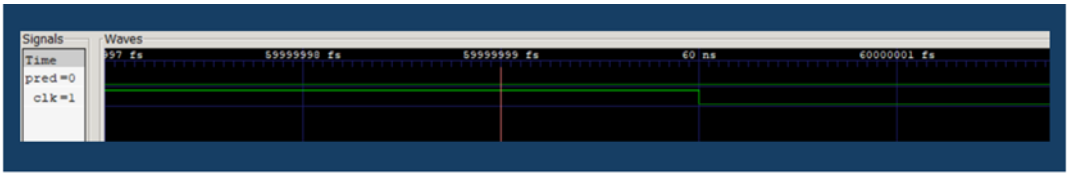


Figure 6. GTK View Output Prediction 0

A medium Gaussian SVM classifier achieved high detection accuracy reaching 100% testing accuracy. Q1.15's fixed-point implementation demonstrated low execution time and robust generalisation, while validating its readiness for real-time FPGA deployment. Hardware-level simulation verified the classifier's accuracy using VHDL. The logic gates and FSM-based control executed Gaussian kernel operations without requiring a look-up table (LUT). Not using a LUT computes values directly, which typically results in higher numerical accuracy, especially for fine-grained inputs. These results ensure the feasibility of the proposed model for embedded solar hotspot detection.

4 DISCUSSION

This section discusses the performance implications and deployment feasibility of the fixed-point Medium Gaussian SVM model for this specific solar PV application. In this section, we highlight the necessity of a custom MATLAB implementation for hardware translation. It evaluates the impact of dataset size on model performance. Further, it examines generalisation and discusses hardware-software coherence for UAV deployment. Additionally, the statistical consistency of the performance metrics across datasets is examined to ensure robustness and reliability, and to validate the quantitative significance of observed performance trends. The discussion is organised under five subsections. They are listed as: *Rationale for Developing a Custom SVM Function in MATLAB Prior to HDL Conversion; Impact of Dataset Size on Training Performance, Generalisation, and Testing Accuracy; Hardware-Software Coherence and Deployment Viability on UAVs;* and *Statistical Analysis of Classifier Performance.* Each subsection interprets the results presented with clarifications on why the observed metrics are meaningful.

4.1 Rationale for Developing a Custom SVM Function in MATLAB Prior to HDL Conversion

The custom SVM function implemented in MATLAB replicates the core decision function behaviour through fixed-point arithmetic. Built-in SVM models and prediction methods in MATLAB are not directly compatible with HDL Coder because they are object-oriented. They depend on floating-point operations. The model has been tailored for fixed-point arithmetic by manually reconstructing the SVM using extracted parameters. It allows accurate translation to VHDL. This approach ensures full control over precision and word length. Further, it facilitates simulation and verification. It bridges the gap between high-level training and hardware implementation, a critical step for FPGA-based machine learning applications. The SVM logic is converted into fixed-point arithmetic to achieve hardware compatibility. Based on this, synthesizable VHDL code is written, including both the main predictor module and a corresponding test bench for simulation. GHDL is used to verify the functional behaviour of the VHDL implementation. Once validated, the design can be synthesised and implemented using FPGA toolchains such as Intel Quartus. The final bitstream is deployed to the FPGA hardware for real-time solar hotspot classification.

4.2 Impact of Dataset Size on Training Performance

The radar plots and tabulated metrics indicate that increasing the dataset size enhances both precision and recall during training. This is attributed to the improved representation of data distribution, allowing the Medium Gaussian SVM to learn more generalizable decision boundaries. The jump from 95.00% accuracy in dataset 1 to nearly 100.00% in dataset 4 highlights this trend. Furthermore, the Q1.15 fixed-point arithmetic does not introduce noticeable quantisation errors as evidenced by the consistent alignment between fixed-point and floating-point results. This confirms that the model retains its discriminative power even under hardware-friendly numeric constraints.

4.3 Generalisation and Testing Accuracy

The model sustains high performance across all five datasets, validating its robustness and transferability from training to deployment. The consistency in recall across test sets, particularly the perfect scores in dataset 3 and high scores in the other datasets, demonstrates the model's ability to detect hotspots without omission. Minor dips in precision for datasets 1 and 4, to 0.9520 and 1.0000, respectively, suggest rare false positives due to edge cases in smaller test samples. Nonetheless, the GHDL-verified VHDL simulation maintains high fidelity with MATLAB predictions, reinforcing that the implemented architecture is both accurate and efficient. These outcomes strongly advocate for its FPGA realisation in real-time solar inspection systems.

4.4 Hardware-Software Coherence and Deployment Viability on UAVs

The VHDL-implemented SVM classifier replicates the behaviour of its MATLAB-trained floating-point counterpart, confirming the correctness of the fixed-point hardware design. Its deterministic latency is a direct consequence of the FSM structure and parallel feature processing. The number of clock cycles depends on the number of support vectors and features, yielding 2702 cycles for 300 SVs and 222 features in the current design. The Gaussian RBF kernel is approximated using a rational function to avoid costly exponentials and look-up tables. This makes the design lightweight and feasible for resource-constrained FPGAs. The model supports further optimisations such as pipelining, early termination, and precision scaling, enhancing efficiency and reducing power consumption. Overall, the design is compact, predictable, and ideal for real-time inference on mid-tier FPGAs in UAV-based PV hotspot detection.

4.5 Statistical Analysis of Classifier Performance

The statistical analysis has been conducted by standard evaluation metrics, namely Mean, Standard Deviation, Coefficient of Variation (CV) and Range, as illustrated in Figure 7. It quantitatively validates the consistency and significance of the performance of Medium Gaussian SVM across different dataset sizes. A strong generalisation is evident across the training and testing phases in the analysis.

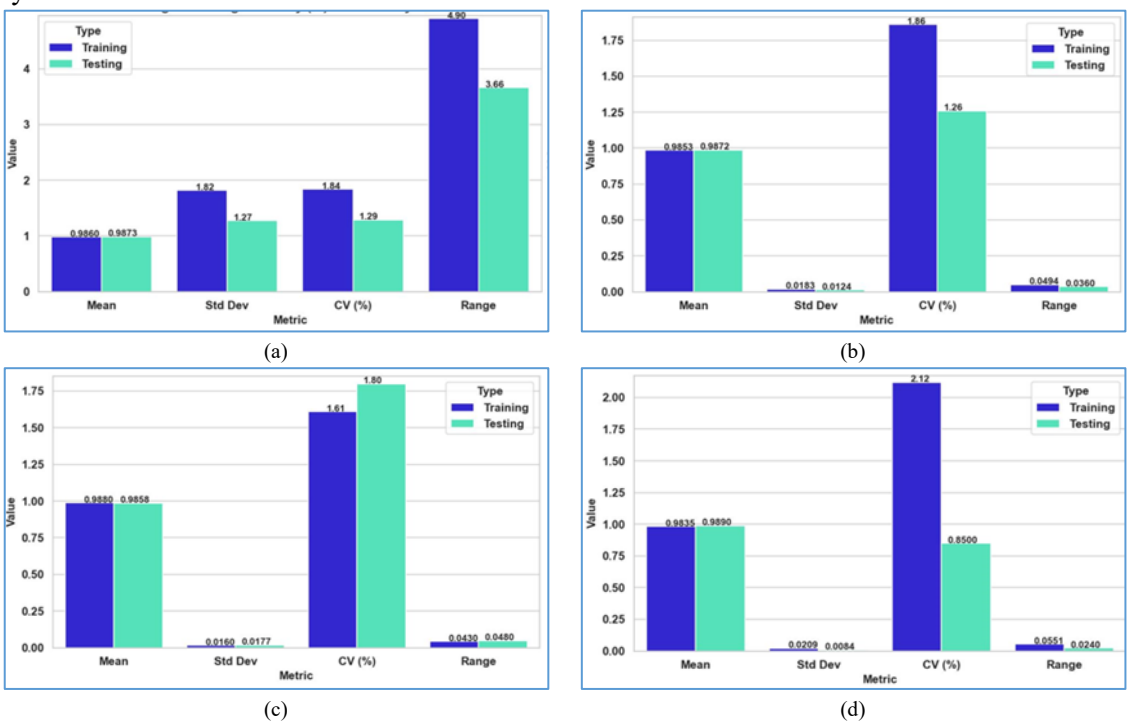


Figure 7. Training and testing consistency metrics for (a) accuracy, (b) f1-score, (c) precision, (d) recall

Figures 7(a) to 7(d) respectively illustrate the accuracy, F1 score, precision, and recall for the training and testing phases. The model achieves a higher test-set mean accuracy (98.73%) than the training-set accuracy (98.60%), as shown in Figure 7(a). Figure 7(b) shows that the range of F1 scores is narrower for testing than for training, suggesting better balance and stability in precision and recall on unseen data.

Figure 7(c) presents precision metrics, with the mean values high for both training and test sets, though the test set shows slightly more variability. Figure 7(d) shows a significantly lower CV in the testing phase than in the training phase. These results suggest the model not only captures more true positives in the test set but does so more consistently. This reinforces its reliability for real-world solar hotspot detection.

Collectively, the fixed-point SVM synthesis confirms the feasibility of hardware deployment, enabling efficient VHDL synthesis. The model demonstrates consistent, high performance across datasets with minimal variance, confirming its generalisation capability. Finally, these findings support the model's potential for real-time, resource-constrained UAV operations in solar inspection tasks.

5 CONCLUSION

This research presents a design pipeline for a fixed-point Medium Gaussian SVM classifier directed for solar hotspot detection in PV modules using MPEG-7 descriptors. We have created a comprehensive pipeline that spans from MATLAB-based preprocessing and Medium Gaussian SVM training to logic-level VHDL implementation for FPGA deployment. Accordingly, the Medium Gaussian SVM classifier consistently achieved high accuracy and F1 scores across five thermal image datasets of varying quality and image counts. To guarantee compatibility with hardware platforms with limited resources, mainly UAV-based FPGA, the conversion from floating-point to Q1.15 fixed-point arithmetic was crucial. As demonstrated by simulation, kernel output validation, and prediction agreement, the logic-based SVM implementation successfully reproduced the MATLAB model's behaviour.

Overall, the proposed approach demonstrates that a handcrafted feature-based, fixed-point SVM classifier preserves software-level accuracy while also satisfying the demanding performance requirements of edge devices. This work provides a reliable and hardware-efficient solution for intelligent fault detection in solar energy systems. This contributes to enhanced PV module maintenance, reduced energy loss, and improved system lifespan.

5.1 Directions for Future Work

There are several opportunities for future enhancement available for this research context. One area is dynamic reconfiguration and online retraining mechanisms within a hybrid FPGA-CPU architecture. Another perspective lies in integration with lightweight deep learning models to further improve the detection of complex anomalies. In addition, deploying the system on actual UAVs with real-time thermal imaging sensors will allow field validation and calibration against diverse irradiance and angle-of-view conditions. For further research, power-optimisation techniques and resource-aware design can be investigated to extend the endurance of UAVs during continuous inspection missions. Furthermore, as noted in [20], emerging paradigms such as TinyML and Transformer variants, along with architectures such as Mamba, provide directions for balancing accuracy and efficiency in real-time applications.

ETHICS DECLARATIONS

Declaration of generative AI and AI-assisted technologies in the writing process

The author used ChatGPT to check for grammatical and spelling errors with the prompt "Proofread this part, focusing on grammatical and spelling errors." After using this tool, the author reviewed and edited the content as needed and takes full responsibility for the content of the published article. Grammarly was also used to ensure clarity and correctness.

ABBREVIATIONS

ACE: Automatic Content Extractor
AI: Artificial Intelligence
BGLR: Binary GLM (Generalised Linear Model) Logistic Regression
CV: Coefficient of Variation
DL: Deep Learning
FPGA: Field Programmable Logic Array
FSM: Finite State Machine
GTKWave: GNU Tool Kit Wave

HDL: Hardware Description Language
IP: Intellectual Property
IR: Infrared
LUT: Look-Up Tables
ML: Machine Learning
MPEG: Moving Picture Experts Group
PV: Photovoltaic
RBF: Radial Basis Function
SVM: Support Vector Machine
UAV: Unmanned Aerial Vehicle
VHDL: VHSIC (Very High-Speed Integrated Circuit) Hardware Description Language

REFERENCES

[1] Nayomi Fernando, Lasantha Seneviratne, Nisal Weerasinghe, Namal Rathnayake, and Yukinobu Hoshino. 2025. Efficient hotspot detection in solar panels via computer vision and machine learning. Information 16, 7, 608. https://doi.org/10.3390/info16070608

[2] Muhammad Umair Ali, Hafiz Farhaj Khan, Manzar Masud, Karam Dad Kallu, and Amad Zafar. 2020.A machine learning framework to identify the hotspot in a photovoltaic module using infrared thermography. Solar Energy 208, 643–651. https://doi.org/10.1016/j.solener.2020.08.027

[3] H. Hussein. 2023. Photovoltaic hotspot detection with image classification deep learning techniques. In Proceedings of the 2023 2nd International Engineering Conference on Electrical, Energy, and Artificial Intelligence (EICEEAI). IEEE, Zarqa, Jordan, 1–6. https://doi.org/10.1109/EICEEAI60672.2023.10590305

[4] Mahmoud Dhimish, Marios Theristis, and Vincenzo d'Alessandro. 2024. Photovoltaic hotspots: A mitigation technique and its thermal cycle. Optik 300 , 171627. https://doi.org/10.1016/j.ijleo.2024.171627

[5] H. Açıkgöz, D. Korkmaz, and Ç. Dandıl. 2022. Classification of hotspots in photovoltaic modules with deep learning methods. TJST 17, 2 (2022), 211–221. https://doi.org/10.55525/tjst.1158854

[6] J. Wang, D. Gao, S. Zhu, S. Wang, and H. Liu. 2019. Fault diagnosis method of photovoltaic array based on support vector machine. Energy Sources, Part A: Recovery, Utilisation, and Environmental Effects 45, 2 , 5380–5395. https://doi.org/10.1080/15567036.2019.1671557

[7] C. J. Burges. 1998. Tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2, 2 , 121–167. https://doi.org/10.1023/A:1009715923555

[8] Fengxi Zhang, Yuying Li, and Zhihao Ye. 2022. Apply YOLOv4-Tiny on an FPGA-based accelerator of a convolutional neural network for object detection. In Proceedings of the 2nd International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA 2022). IOP Publishing, Dali City, China, Article 012032. https://doi.org/10.1088/1742-6596/2303/1/012032

[9] Karuppiah Natarajan, Praveen Kumar B., and Vankadara Kumar. 2020. Fault detection of solar PV system using SVM and thermal image processing. International Journal of Renewable Energy Research 10, 967–977.

[10] X. Song, H. Wang, and L. Wang. 2014. FPGA implementation of a support vector machine-based classification system and its potential application in the smart grid. In Proceedings of the 2014 11th International Conference on Information Technology: New Generations (ITNG). IEEE, Las Vegas, NV, 397–402. https://doi.org/10.1109/ITNG.2014.45

[11] Mateusz Wasala and Tomasz Kryjak. 2022. Real-time HOG+SVM-based object detection using SoC FPGA for a UHD video stream. arXiv:2204.10619 [cs.CV]. https://doi.org/10.48550/arXiv.2204.10619

[12] A. Escobedo and A. Lin. 2016. Tessellation-based multi-block memory-mapping scheme for high-level synthesis with FPGA. In Proceedings of the 2016 International Conference on Field-Programmable Technology (FPT). IEEE, Xi'an, China, 125–132. https://doi.org/10.1109/FPT.2016.7929517

[13] X. A., W. A. Najjar, and A. K. Roy-Chowdhury. 2015. Evaluation and acceleration of high-throughput fixed-point object detection on FPGAs. IEEE Transactions on Circuits and Systems for Video Technology 25, 6, 1051–1062. https://doi.org/10.1109/TCSVT.2014.2360030

[14] H. Brum, M. Véstias, and H. Neto. 2024. LiDAR 3D object detection in FPGA with low bitwidth quantization. In Proceedings of Applied Reconfigurable Computing. RC 2024. Lecture Notes in Computer Science, vol. 14553. Springer, Cham, 90–105. https://doi.org/10.1007/978-3-031-55673-9_7

[15] Muhammad Ihsan Al Hafiz, Naresh Ravichandran, Anders Lansner, Pawel Herman, and Artur Podobas. 2025. A reconfigurable stream-based FPGA accelerator for Bayesian confidence propagation neural networks. arXiv:2503.01561 [cs.AR]. https://doi.org/10.48550/arXiv.2503.01561

[16] Srikanth Ramadurgam and Darshika G. Perera. 2021. An efficient FPGA-based hardware accelerator for a convex optimization-based SVM classifier for machine learning on embedded platforms. Electronics 10, 11, 1323. https://doi.org/10.3390/electronics10111323

[17] Shereen Afifi, Hamid Gholamhosseini, and Roopak Sinha. 2019. A system on chip for melanoma detection using an FPGA-based SVM classifier. Microprocessors and Microsystems 65, 57–68. https://doi.org/10.1016/j.micpro.2018.12.005

[18] Shereen Afifi, Hamid Gholamhosseini, and Roopak Sinha. 2015. Hardware implementations of SVM on FPGA: A state-of-the-art review of current practice. International Journal of Innovative Science, Engineering & Technology (IJISET) 2, 733–752.

[19] Mohammed H. Yacoub, Samar M. Ismail, Lobna A. Said, Ahmed H. Madian, and Ahmed G. Radwan. 2024. Support vector machine reconfigurable hardware implementation on FPGA. Franklin Open 7, 100115. https://doi.org/10.1016/j.fraope.2024.100115

[20] T. Suwannaphong, F. Jovan, I. Craddock, and R. McConville. 2025. Optimising TinyML with quantization and distillation of transformer and mamba models for indoor localization on edge devices. Scientific Reports 15, 10081. https://doi.org/10.1038/s41598-025-94205-9