Received 28 December 2025, accepted 28 April 2026, date of publication 30 June 2026
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.25.2.4663

VHDL-Implemented Neuro-Fuzzy Cascade for Renewable Energy Defect Classification

LESIA DUBCHAK[1], BOHDAN RUSYN[2]
[1]Department of Computer Engineering, West Ukrainian National University, Ternopil, 46009, Ukraine
[2]Institute of Physics and Mechanics of the National Academy of Sciences of Ukraine, 79060, Lviv, Ukraine

Corresponding author: Lesia Dubchak (e-mail: dlo@wunu.edu.ua).

This work was supported in part by the Ministry of Education and Science of Ukraine under the grant "An intelligent system for recognizing defects in green energy facilities using UAVs" September 2024–August 2026

ABSTRACT This paper presents a hardware-oriented neuro-fuzzy cascade for the real-time classification of wind turbine blade defects. A modified SqueezeNet model generates an informative feature vector, which is processed by Hypersector Fuzzy LVQ, Fuzzy BSB, and a modified Wang–Mendel classifier. Hypersector Fuzzy LVQ performs primary classification, whereas the switching logic evaluates the confidence margin and a hardware-efficient feature-variability indicator to select an auxiliary classifier for uncertain samples. The complete classifiers are trained offline, while their compact inference representations and switching logic are implemented in VHDL. Functional verification in Active-HDL demonstrated correct sample-dependent routing and synchronous decision generation. On the test subset, the cascade correctly classified samples, corresponding to an accuracy of 98.33%. The proposed architecture provides low decision latency after CNN feature extraction and can serve as a basis for embedded defect-monitoring systems. Its current limitations include empirical threshold selection, a relatively small test subset, and the absence of physical FPGA synthesis.

KEYWORDS renewable energy facilities; defect classification; neuro-fuzzy system; cascade architecture; SqueezeNet; Hypersector Fuzzy LVQ; Fuzzy BSB; Wang–Mendel network; FPGA; real-time monitoring.

I. INTRODUCTION AND RELATED WORK

THE rapid development of renewable energy systems has increased the need for reliable and automated monitoring of their technical condition. Wind turbine blades and photovoltaic panels operate under variable environmental conditions and are exposed to mechanical damage, thermal stress, corrosion, erosion, and other defects that may reduce energy efficiency and cause equipment failure.

Traditional inspection methods are often time-consuming, labour-intensive, and dependent on expert assessment. Therefore, intelligent approaches combining computer vision, deep learning, fuzzy logic, and embedded computing are increasingly being investigated to enable accurate, interpretable, and real-time defect detection.

Current research on the automated detection of defects in renewable energy facilities is developing primarily along three directions: the application of deep CNN/YOLO/Mask R-CNN models, the use of multisensor RGB/thermal/UAV data, and the development of ensemble or fuzzy decision-making schemes [3].

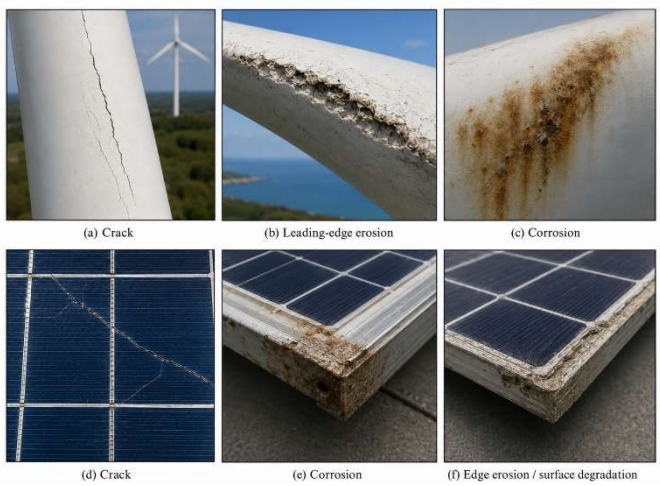


Figure 1. Examples of common defects in wind turbine blades and photovoltaic panels: (a) crack, (b) leading-edge erosion, (c) corrosion of a wind turbine blade, (d) crack, (e) corrosion, and (f) edge erosion/surface degradation of a photovoltaic panel [1,2]

In [3], the authors investigated Xception, ResNet-50, AlexNet, VGG-19, and a custom CNN for anomaly detection on small wind turbine blades and demonstrated very high accuracy for transfer-learning models. However, such solutions largely remain black boxes and do not provide sufficient interpretability of their decisions.

A separate research direction involves YOLO- and Mask R-CNN-based approaches to defect localization. The authors of [4] compared YOLO and Mask R-CNN for detecting blade cracks, holes, and erosion, with YOLOv9-C achieving the best mAP50 performance. These methods are effective for localizing defective regions, but they require large volumes of annotated data and may be difficult to deploy on resource-constrained edge platforms.

In [5], the authors proposed an RGB and thermal image fusion approach for detecting anomalies in wind turbine blades. This approach improves the informativeness of the input data; however, it increases the requirements for sensing equipment, synchronization, and computational processing.

For solar energy facilities, research on detecting PV panel defects using CNN, YOLO, and attention-based models is particularly relevant. Specifically, the authors of [6] considered a deep-learning pipeline for detecting defects and dust on solar panels, while [7] applied deep learning to electroluminescence images of solar cells. These approaches demonstrate high accuracy, but they are generally focused on a single object type and do not provide a universal architecture for different renewable energy facilities.

Particular attention should be paid to [8], which proposed an attention-based deep-learning model for detecting surface defects on both solar PV panels and wind turbine blades. This approach is closely related in terms of the application domain because it also covers two types of renewable energy facilities. However, it is based on a Vision Transformer, whereas the present study focuses on compact features, interpretable neuro-fuzzy classification, and a hardware-suitable cascade implementation.

Neuro-fuzzy approaches are less widely represented in this domain. In [9, 10], an adaptive neuro-fuzzy system was proposed for detecting wind turbine blade defects to reduce computational complexity and improve accuracy. This direction can be regarded as the methodological basis for further development in the form of a cascade architecture, Hypersector Fuzzy LVQ, Fuzzy BSB, and adaptive cascade bagging.

An important analogue is the study in [11], which proposed combining Mask R-CNN with fuzzy voting to classify wind turbine blade defects. This approach is the closest prototype of the proposed system because it also uses fuzzy aggregation of the outputs of several models. However, fuzzy logic is employed mainly as a voting mechanism on top of deep models, where as in the proposed approach, the neuro-fuzzy modules are full-featured classifiers operating within cascade and adaptive bagging architectures.

A related adaptive cascade-bagging architecture was previously presented in [12]. That study focused on the adaptive weighting and aggregation of classifier outputs. In contrast, the present work investigates conditional sample-dependent routing. The primary FLVQ decision is accepted for sufficiently confident samples, whereas only one specialized auxiliary decision is selected for uncertain samples according to the confidence margin and feature-variability indicator. Therefore, the contribution of the present study concerns hardware-oriented switching logic and conditional decision selection rather than adaptive weighted aggregation.

Despite the progress achieved in the previous studies, the available solutions address only individual components of the defect-classification problem. The modified SqueezeNet model was previously used to form informative feature representations, whereas the Fuzzy BSB, Wang–Mendel, and Hypersector Fuzzy LVQ methods were investigated as separate classifiers. However, the problem of dynamically selecting an appropriate neuro-fuzzy classifier according to the reliability and characteristics of each input sample has not been sufficiently addressed. Existing ensemble approaches generally process every sample using all component models and subsequently aggregate their outputs. Consequently, they do not provide sample-dependent routing between classifiers with different functional properties, nor do they explicitly consider the suitability of such routing for implementation as digital logic.

This study aims to develop and investigate a hardware-oriented neuro-fuzzy cascade architecture for reliable real-time classification of defects in renewable energy facilities using feature representations. The main objectives of the research are:

- To analyse existing methods for automated defect detection and classification in renewable energy facilities.
- To form an informative feature space using a modified SqueezeNet convolutional neural network.
- To integrate Hypersector Fuzzy LVQ, Fuzzy BSB, and a modified Wang–Mendel network into a unified cascade architecture.
- To develop hardware-oriented switching logic for selecting an auxiliary classifier according to the confidence margin of the primary decision and a feature-variability indicator.
- To implement the proposed cascade decision-making architecture in VHDL.
- To verify the architecture using the Active-HDL simulation environment.

II. NEURO-FUZZY METHODS OF DEFECTS CLASSIFICATION

The proposed intelligent system for classifying defects in renewable energy facilities combines deep-learning methods and neuro-fuzzy approaches to improve diagnostic accuracy and robustness. The overall architecture consists of four interconnected stages: feature-space formation, neuro-fuzzy classification, fuzzy associative analysis, and decision-making.

At the first stage, a modified SqueezeNet convolutional neural network is used to automatically extract informative features from defect images acquired by an unmanned aerial vehicle (UAV). SqueezeNet is a compact deep-learning architecture based on Fire modules that combine squeeze and expand operations in the feature space. By using 1x1 convolutions, the network substantially reduces the number of parameters and computational cost without a significant loss of accuracy. This is especially important for UAV-based inspection, where the acquired images are characterized by high data variability caused by changes in illumination, viewing angle, flight altitude, camera-to-object distance, background conditions, defect scale, surface texture, and partial occlusions. Such variability leads to differences in the

visual representation of the same defect class and may increase the overlap between classes in the feature space. Therefore, the use of a compact convolutional feature extractor enables the formation of stable and discriminative descriptors that preserve the structural and textural properties of defects under variable acquisition conditions [9].

The extracted features are supplied to the input of a modified Wang-Mendel neuro-fuzzy network. The classical Wang-Mendel method is an algorithm for automatically generating fuzzy rules from training data [13-15]. In the proposed study, it is modified for multiclass classification by using a multi-output structure, Gaussian membership functions, a softmax output, and a cross-entropy training criterion. This enables interpretable classification under uncertainty and partial overlap between defect classes [9].

To substantiate the position of the proposed modified Wang–Mendel method among modern heuristic and metaheuristic approaches, it is appropriate to compare it with methods based on genetic algorithms, swarm optimization, bio-inspired algorithms, and ensemble models. In most such studies, the heuristic algorithm does not directly perform defect classification but is used as an auxiliary optimization mechanism: for selecting informative features, tuning ensemble weights, reducing the dimensionality of the input space, or accelerating neural network training. In contrast, the proposed Wang–Mendel method forms a neuro-fuzzy decision-making model in which classification is based on fuzzy rules, membership functions, and an adaptive multi-output mechanism for class evaluation.

In [16], a Genetic Algorithm + Neural Network (GA+NN) approach is proposed for fault diagnosis in grid-connected photovoltaic systems. The genetic algorithm is used for feature selection, while the neural network performs classification. For the complete set of PV system faults, the GA+NN method achieved Accuracy equal 88.48%, Recall equal 88.48%, and Precision equal 90.65%; at the same time, the number of features was reduced from 8 to 4, and the computation time was 0.40 s. Compared with this result, the proposed modified Wang–Mendel method demonstrates higher test accuracy: 94% versus 88.48%, corresponding to an improvement of +5.52 percentage points. If the refined value of 96.88% is used in the dissertation text, the advantage over GA+NN increases to +8.40 percentage points. The proposed method must achieve this result without population-based optimization or an additional genetic feature-selection stage, since it operates on an already informative feature vector.

In [17], a weighted ensemble learning-based method for PV system fault diagnosis using a genetic algorithm is proposed. In this case, GA optimizes the weights of individual classifiers in the ensemble, which includes Logistic Regression, Support Vector Machine (SVM), and k-Nearest Neighbors (KNN). The authors report an accuracy of approximately 99% for diagnosing open-circuit and line-line faults in PV arrays. According to the detailed experimental data, the validation accuracy is 98.61%, while class-wise Precision values are 99%, 100%, and 97%, and Recall values are 97%, 100%, and 99%. Formally, this result exceeds the accuracy of the proposed Wang–Mendel method by 4.61 percentage points when compared with Accuracy equal 94%. However, the approach by [17] uses an ensemble of several models and genetic optimization of weights, whereas the proposed method is based on a single compact neuro-fuzzy structure. In addition, authors use 16 selected features out of 30, whereas the proposed method uses only 4 features; thus, the number of input parameters is four times smaller. This is a significant advantage for implementation on edge devices and in hardware-oriented systems.

In [18], fault diagnosis in wind energy conversion systems is considered using Particle Swarm Optimization + Neural Network (PSO-NN) and Reduced Particle Swarm Optimization + Neural Network (RPSO-NN). Particle swarm optimization is used to select the most informative features and reduce data redundancy. RPSO-NN achieved Accuracy 98.52% on the test set with a testing time of 1.19 s, whereas a conventional NN achieved 93.70%, SVM achieved 92.14%, and KNN achieved 88.30%. Compared with RPSO-NN, the proposed method is inferior in terms of absolute accuracy by 4.52 percentage points; however, it still outperforms the conventional NN reported in the same study: 94% versus 93.70%. This is an important argument, since it shows that even without swarm optimization, the proposed neuro-fuzzy method reaches or exceeds the level of baseline neural-network approaches. An additional advantage is that the Wang–Mendel method forms interpretable fuzzy rules, whereas PSO-NN and RPSO-NN remain predominantly black-box models.

In [19], a Convolutional Neural Network + Bitterling Fish Optimization (CNN+BFO) approach is proposed for detecting defects in PV panels. The authors combined five pre-trained CNN architectures — GoogleNet, SqueezeNet, ResNet-50, VGGNet-16, and AlexNet — with Bitterling Fish Optimization for feature selection. The best result was obtained for VGGNet-16+BFO: Accuracy 98.75%, Sensitivity 98.72%, Specificity 98.78%, Precision 98.76%, and F1-score 98.74%. This approach is the strongest among the considered analogues in terms of accuracy; however, it has a significantly more complex architecture, since it requires a deep CNN model and an additional bio-inspired optimization algorithm. In the proposed approach, the classification block operates with a compact vector $z \in R^4$, requires a small amount of memory, and, in the edge implementation of the Wang–Mendel core, the reported execution time is approximately 0.065 s with a throughput of 5–6 FPS on Raspberry Pi 5. Therefore, although CNN+BFO exceeds Wang–Mendel in terms of accuracy by 4.75 percentage points, the proposed method has advantages in compactness, transparency, and practical suitability for embedded implementation.

To improve the system's robustness to noise and distortion, the Fuzzy Brain-State-in-a-Box (Fuzzy BSB) model is used [20]. Unlike the classical BSB associative-memory network, the proposed approach incorporates fuzzy membership functions and a geometric representation of data in a multidimensional hypercube. The method can simultaneously estimate the degree of membership of a feature vector in several classes, which is important when classifying defects with similar characteristics and overlapping feature regions.

The final decision is made using Hypersector Fuzzy LVQ. This method extends Fuzzy Learning Vector Quantization by normalizing the feature space onto the surface of a unit hypersphere [21]. A direction vector defining the corresponding hypersector is formed for each class. The membership of a new sample in a class is estimated from the angular distance between the feature vector and the class direction using fuzzy membership functions. This approach effectively separates closely spaced classes, reduces the effect of feature overlap, and improves classification reliability.

Thus, SqueezeNet forms a compact feature space, the Wang-Mendel network provides neuro-fuzzy classification and rule generation, Fuzzy BSB performs fuzzy associative data processing, and Hypersector Fuzzy LVQ provides high-accuracy class separation in a hyperspherical feature space. Combining these methods produces an intelligent defect recognition system characterized by high accuracy, interpretable decisions, and the low computational complexity required for real-time operation.

It should be distinguished between the complete algorithmic models used during software-based training and their hardware-oriented inference representations used in the proposed cascade. The parameters of Hypersector Fuzzy LVQ, Fuzzy BSB, and the modified Wang–Mendel model are estimated offline using the training data. The obtained prototypes, weight matrices, rule parameters, and scaling coefficients are subsequently converted into fixed numerical constants and stored in the VHDL parameter package. Therefore, the hardware modules do not perform model training or rule-base generation during operation; they implement only the inference stage required to classify a new feature vector.

To reduce hardware complexity and decision latency, computationally expensive nonlinear operations are transformed or approximated before hardware deployment. The resulting VHDL modules preserve the decision functions of the trained classifiers within the examined compact feature space but should be regarded as hardware-oriented inference representations rather than complete implementations of the original training algorithms. This separation between offline training and online inference enables the use of the developed neuro-fuzzy models in resource-constrained digital systems.

III. IMPLEMENTATION OF THE INTELLIGENT SYSTEM CASCADE ARCHITECTURE IN ACTIVE-HDL

To improve the accuracy of defect classification for renewable energy facilities, a cascade architecture is proposed in which the modified SqueezeNet network forms an informative feature space and subsequent classification is performed sequentially by several intelligent modules (Figure 2). In the first stage, the hypersector-based Fuzzy LVQ method rapidly recognizes primary defects in a hyperspherical feature space. In difficult cases, when classification confidence is insufficient or classes overlap, the switching-logic block is activated and directs the data to either the Fuzzy BSB module or the modified Wang-Mendel method. Fuzzy BSB improves robustness to noise and distortions, whereas the Wang-Mendel method ensures decision interpretability by generating IF-THEN fuzzy rules. This approach achieves a balance among processing speed, accuracy, and classification explainability.

The cascade architecture of the intelligent system for detecting and classifying defects in renewable energy facilities was implemented in VHDL using Active-HDL 9.1 [22]. The architecture implements a conditional cascade with switching logic. The high-speed base classifier always generates the primary decision, while auxiliary modules are engaged only when this decision is insufficiently reliable. This organization reduces the average processing time, avoids excessive computational load, and maintains high accuracy.

At the Active-HDL simulation level, the convolutional feature-extraction module, namely the SqueezeNet network itself, was not implemented. Instead, its output was modeled as a ready-made vector of four features supplied to the cascade architecture. This approach is technically valid because, in a real embedded system, the CNN module and the decision-making module can be implemented at different levels: the former as a software-based or hardware-accelerated feature extractor and the latter as a high-speed digital classifier implemented in FPGA or HDL logic.

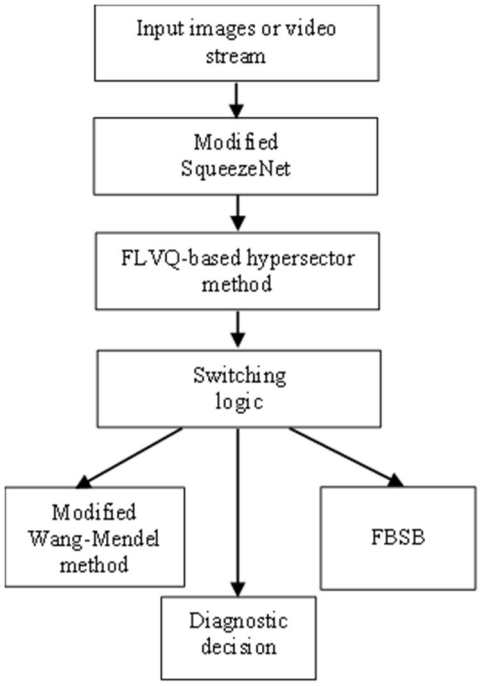


Figure 2. General diagram of the intelligent system cascade architecture

The software-level implementation of the modified SqueezeNet feature extractor is justified by the functional separation between image-level feature extraction and hardware-level decision making. The convolutional part of the system processes high-dimensional image data and would require substantial FPGA resources for convolution operations, feature-map buffering, weight storage, memory-bandwidth control, and fixed-point optimization. Since the main objective of this study is the design and verification of a hardware-oriented neuro-fuzzy cascade rather than the development of a dedicated CNN accelerator, SqueezeNet is used as a software-based module that generates a compact four-dimensional feature vector. This vector is then supplied to the VHDL cascade, where low-latency classification and conditional routing are performed. Such hardware–software partitioning preserves flexibility for CNN retraining and adaptation to new acquisition conditions, while allowing the neuro-fuzzy decision stage to be implemented as compact, deterministic, FPGA-suitable digital logic.

Within the developed project, the architecture consists of a package containing parameters and training (test) samples, the base Hypersector-FLVQ classifier, the auxiliary FBSB and Wang-Mendel modules, the switching-logic module, the top-level cascade-system module, and a testbench. All modules have clearly defined functions and interact through unified data and control signals.

The entire VHDL model is built around a parameter package that contains global constants, data types, weight matrices, class prototypes, and training and test datasets. The package defines the number of features, number of classes,

scaling factors, switching-logic thresholds, and the argmax4, second_max4, clamp01k, and abs_int functions. This unifies all modules because they use the same parameters and operate with a common data format. The TRAIN_X, TRAIN_Y, TEST_X, and TEST_Y feature and label sets are used for training and testing the cascade architecture.

The first functional module of the cascade system is hyper_flvq_core, which implements the base classification level. Its task is to rapidly generate the primary decision through geometric fuzzy analysis of the feature vector. Four feature components, x0, x1, x2, and x3, are supplied to the module inputs. The outputs are four numerical scores, s0, s1, s2, and s3, representing the input sample's membership in the corresponding classes, along with the pred_class signal containing the index of the class with the maximum score.

Before hardware inference, the input feature vectors and class prototypes are normalized during the software-preprocessing stage. Consequently, the dot product computed in the VHDL module represents the cosine similarity between the normalized input vector and the corresponding class prototype:

$$\cos\,\theta_k = \frac{\mathrm{x}^T \mathrm{v}_k}{\|\,\mathrm{x}\,\|\|\,\mathrm{v}_k\,\|}.$$

Because $\|\,\mathrm{x}\,\|=\|\,\mathrm{v}_k\,\|=1$, this expression is reduced to $\cos\,\theta_k = \mathrm{x}^T\mathrm{v}_k$. Maximizing the dot product is therefore equivalent to minimizing the angular distance θ_k. This transformation eliminates the need to calculate the inverse cosine function in hardware while preserving the class-ranking result of the original Hypersector Fuzzy LVQ model.

The module implements four parallel computational channels that can be interpreted as four hypersectors in the normalized feature space. Each channel computes the dot product of the input vector and a class prototype stored in the FLVQ_PROTOS matrix. The resulting d0, d1, d2, and d3 values are scaled by SCALE_X to produce normalized output scores s0_i, s1_i, s2_i, and s3_i. These signals are supplied to the module outputs and used by the subsequent switching logic.

Thus, hyper_flvq_core implements the inference decision of the trained Hypersector Fuzzy LVQ model using cosine-equivalent scores. Prototype adaptation and hyperspherical normalization are performed offline and are not included in the RTL inference path. Structurally, this is the simplest and fastest module in the entire system. It contains no iterative processes, fuzzy-inference rules, or multistage aggregation procedures (Figure 3). From an engineering perspective, this makes it an ideal primary classifier with minimum latency and the ability to process real-time data streams.

The block diagram clearly shows that each component of the input vector is supplied in parallel to the d0-d3 computation blocks, followed by normalization and maximum-value selection using the argmax4 function. This organization explains the module's high processing speed and predictable timing behavior.

The second architectural component is fbsb_core, which implements a hardware-oriented inference representation of the trained Fuzzy BSB classifier. The complete Fuzzy BSB model, including the estimation of fuzzy class prototypes and membership parameters, is formed during offline training. The resulting parameters are represented by the FBSB_W matrix and stored as constants in the VHDL package. During online operation, the module performs only the matrix-based mapping and bounded activation required to obtain the class scores.

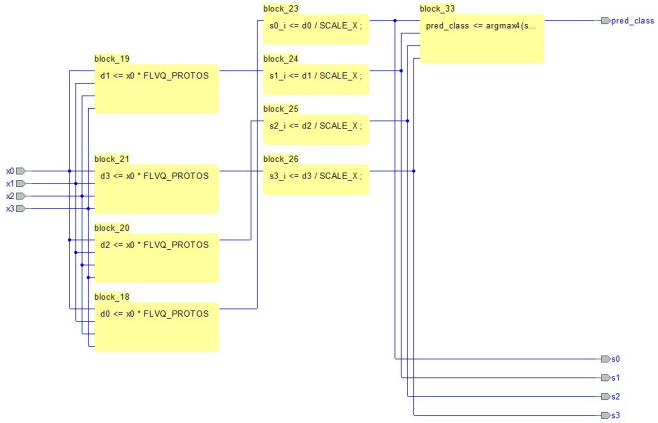


Figure 3. Block diagram of the Hypersector-FLVQ classifier module

The same four features, x0-x3, are supplied to the module inputs. Four parallel computation paths, a0, a1, a2, and a3, perform matrix multiplication of the input vector by the corresponding columns of FBSB_W. Each value is then constrained by the clamp01k function, which maps the result to the admissible range. The resulting s0_i-s3_i signals are fuzzy class-membership scores. As in the preceding module, argmax4 determines the index of the maximum value, which is output as pred_class (Figure 4).

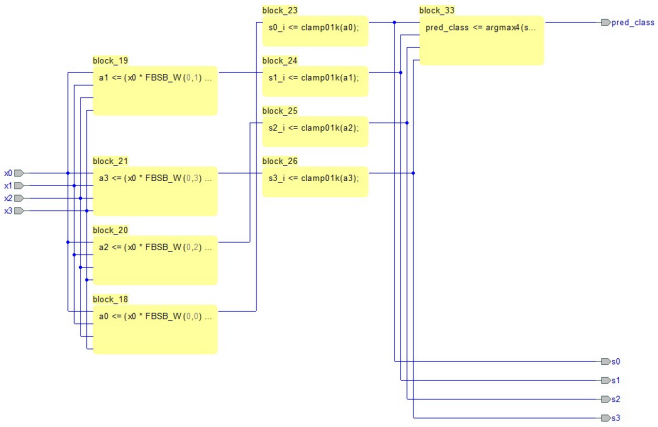


Figure 4. Block diagram of the FBSB module

Compared with Hypersector-FLVQ, this module is less geometric and more associative. This branch is intended to improve robustness in cases characterized by class overlap and perturbations of the feature vector. Its actual contribution is evaluated experimentally by comparing the cascade results with those of the base classifier.

The diagram (see Figure 3) clearly shows that the x0-x3 input signals are supplied to four parallel arithmetic blocks, whose outputs pass through the limiting stage and are forwarded to the class-determination block. This organization confirms that the module retains parallel computation and introduces no substantial additional delay.

The third classification block, wang_mendel_core, implements a hardware-oriented inference approximation of the modified Wang–Mendel classifier. In the complete

software model, Gaussian membership functions are used to determine the activation strengths of the fuzzy rules, while the multiclass outputs are formed using rule aggregation and softmax normalization. Model training, membership-function identification, rule generation, and parameter optimization are performed offline.

For hardware deployment, the responses of the trained rule system within the examined feature domain are transformed into four class-specific discriminant functions $r_k(\mathrm{x}) = \mathrm{w}_k^T\mathrm{x} + b_k, \forall k = \overline{1,K}$, where w_k and b_k are parameters derived from the trained Wang–Mendel model. The hardware class scores are calculated as $s_k = \mathrm{sat}\,(\alpha_k, r_k(\mathrm{x}))$, where α_k is a scaling coefficient and $\mathrm{sat}\,(\cdot)$ denotes bounded activation. The final class is selected using the argmax4 function.

Accordingly, the VHDL module (Figure 5) does not generate fuzzy rules, calculate Gaussian membership functions, or perform softmax normalization during online operation. These operations belong to the offline model-development stage. The module implements a compact approximation of the resulting multiclass decision surface, thereby reducing hardware latency and resource requirements.

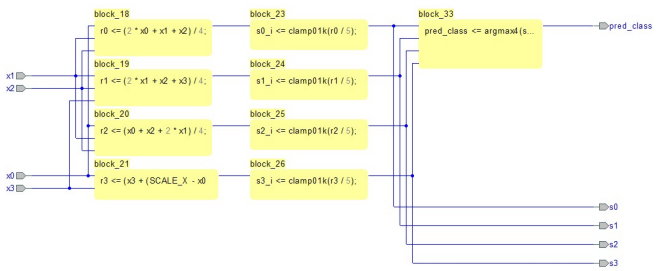


Figure 5. Block diagram of the Wang-Mendel module

Because the rule antecedents are not evaluated directly in the current RTL module, interpretability is provided by the original software-based Wang–Mendel model and its extracted rule base rather than by the internal arithmetic structure of wang_mendel_core. The hardware output can be linked to the corresponding software rule explanation using the selected class and rule identifier. Direct hardware evaluation of fuzzy rules remains a direction for further implementation.

The most important element of the cascade architecture is switch_logic_core because it determines the processing route for the current sample. Whereas the preceding modules generate partial decisions, the switching logic controls the system.

The inputs to this module are the FLVQ scores flvq_s0-flvq_s3, the primary class flvq_class, and the feature vector x0-x3. Its outputs are conf_margin, the confidence margin of the primary classifier; use_aux, a signal indicating whether an auxiliary module must be activated; and route_sel, the processing-route code.

The module logic consists of three consecutive stages. First, the maximum score max1_i corresponding to the current FLVQ decision is determined. Second, the second-largest score max2_i is calculated using second_max4. Their difference forms margin_i, the confidence margin. The larger this margin, the more clearly FLVQ has separated one class from the others.

At the third stage, a hardware-efficient feature-variability indicator is calculated from the feature vector:

$$V(\mathrm{x}) = \mid x_1 - x_2 \mid + \mid x_2 - x_3 \mid + \mid x_3 - x_4 \mid. \qquad (1)$$

Here, $x_1, \ldots, x_4$ are the components of the feature vector represented in a fixed order corresponding to the defect classes. The indicator $V(x)$ is defined as the sum of absolute differences between adjacent components of the compact feature vector. This formulation corresponds to a discrete measure of local variation and makes it possible to estimate feature instability without redundant accounting of global differences between distant components. In particular, the term $\mid x_1 - x_4 \mid$ is not included separately because the global variation between x_1 and x_4 is already partially reflected by the consecutive local differences $\mid x_1 - x_2 \mid$, $\mid x_2 - x_3 \mid$, and $\mid x_3 - x_4 \mid$. This also reduces computational complexity, which is important for hardware-oriented implementation. The calculated value is used as an empirical indicator of the nonuniformity of the feature representation. A larger value indicates stronger differences between adjacent feature components and is treated as a condition requiring the robust auxiliary classification branch.

It should be emphasized that $V(\mathrm{x})$ is not a direct physical estimate of image noise. It is a hardware-oriented heuristic criterion used exclusively for selecting a processing route. Its principal advantage is low implementation complexity because it requires only three subtraction operations, three absolute-value operations, and two additions. The feature-component order remains unchanged during training and testing, ensuring consistent calculation of the indicator for all samples.

The routing mechanism uses two complementary criteria. The confidence margin determines whether the primary Hypersector Fuzzy LVQ decision requires refinement, whereas the feature-variability indicator determines which auxiliary classifier should process the uncertain sample. These thresholds were determined empirically from the training data and remained fixed during final testing.

The main module process is then executed. If margin_i exceeds THRESH_CONF, the primary FLVQ decision is considered sufficiently reliable, the auxiliary module is not activated (use_aux = '0'), and route_sel = 0. If the confidence margin is below the threshold, the system enters refinement mode (use_aux = '1'). It then selects between the two auxiliary modules. If noise_i exceeds THRESH_NOISE, route_sel = 1 is selected, corresponding to FBSB as the module that is more robust to noise disturbances. If the noise level is below this threshold but the FLVQ decision remains uncertain, route_sel=2 is selected, corresponding to the expert Wang-Mendel module.

Thus, switch_logic_core (Figure 6) implements more than simple switching: it provides a conditional decision-making mechanism based on two criteria, namely the discriminative capability of FLVQ and the quality of the input feature vector. This gives the architecture a broader form of adaptivity than a simple combination of modules.

In the VHDL implementation, the feature-variability indicator retains the internal signal name noise_i, while its threshold is denoted by THRESH_NOISE. These identifiers are preserved to maintain consistency with the implemented source code. However, in the methodological interpretation, noise_i represents an empirical feature-variability indicator rather than a direct measurement of physical image noise. The threshold values were selected empirically using the training samples and were not adjusted using the true labels of the final test subset.

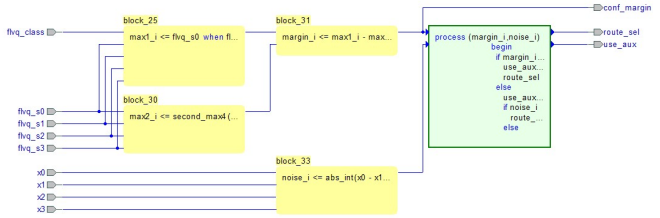


Figure 6. Block diagram of the switching-logic module based on the FLVQ confidence margin and feature-vector variability

The diagram shows that max1_i, max2_i, and margin_i are calculated from the FLVQ score vector, whereas noise_i represents the hardware-level implementation of the feature-variability indicator $V(x)$ (equation (1)). These parameters are supplied to the route-selection process, which generates use_aux, route_sel, and conf_margin.

The cascade_decision_top module performs the integration function and constitutes the top level of the entire project. It combines the three classifiers and the switching-logic module and coordinates their interaction. Its inputs are the synchronization signals clk, rst, valid_in, and train_mode, the feature vector x0-x3, and the true class true_label. Its outputs include the decisions of all three classifiers, the final class, the route code, and auxiliary-logic signals.

The top-level module performs three tasks. First, it supports parallel computation of the partial FLVQ, FBSB, and Wang-Mendel decisions. Second, it transfers the FLVQ scores to the switching module to determine whether the auxiliary branch should be engaged. Third, it generates the final decision final_class from route_sel, selecting the result directly from FLVQ, FBSB, or Wang-Mendel.

Importantly, this implementation preserves the formal simplicity of the top level. Although the system logically operates as a conditional cascade, all three VHDL modules can be physically implemented simultaneously, with the final selection performed by switching logic. This simplifies simulation and makes it easy to observe the behavior of all modules in the timing diagram.

To verify model operation, the tb_cascade_decision testbench was implemented to reproduce the complete system operating cycle. It generates the clock signal, performs the initial reset, and sequentially supplies the training and test samples.

The testbench not only applies signals but also collects statistics. It counts the correct decisions generated by FLVQ, FBSB, Wang-Mendel, and the final cascade architecture, as well as the number of auxiliary-module activations and the number of times each auxiliary branch is selected.

The simulation results for the cascade architecture of the intelligent defect classification system confirm the correct operation of the proposed approach and its effectiveness for real-time processing of multidimensional features.

Analysis of the timing diagrams obtained during simulation showed that the classification decision is generated in accordance with the intended logic of the cascade architecture. For each input sample, the primary decision is generated by the base hypersector classifier and its reliability is then evaluated by the switching-logic module. When the confidence margin of the primary decision is insufficient, an auxiliary classification module is automatically activated to refine the result.

As shown in the timing diagram (Figure 7), the final_class signal coincides with true_label in most cases, confirming correct classification. The route_sel_out and use_aux_out signals exhibit the characteristic behavior associated with switching among the modules. In particular, when conf_margin_out decreases, the system switches to the auxiliary algorithms, thereby avoiding classification errors.

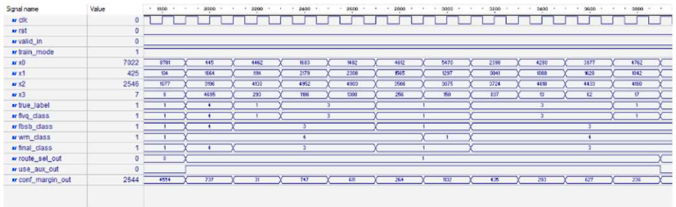


Figure 7. Timing diagram of the cascade architecture

Testing on an independent subset shows the accuracy of 98.33%. One sample was misclassified. Owing to the limited test-set size, this result should be interpreted as a functional validation of the proposed architecture rather than as a definitive estimate of its generalization performance.

It is important to note that these results demonstrate the functional feasibility of the switching mechanism, although its individual contribution requires further ablation analysis. The base Hypersector-FLVQ classifier rapidly generates the primary decision, but its confidence may be insufficient in difficult or boundary cases. In such situations, activating the FBSB or Wang-Mendel modules enables additional analysis and produces a more stable decision. The cascade architecture, therefore, combines high processing speed with improved accuracy.

Regarding hardware cost, the implemented architecture is characterized by moderate resource utilization. The principal memory requirements are associated with storing classifier parameter matrices and service constants. The architecture stores only compact prototype and parameter matrices. However, exact memory consumption and FPGA resource utilization must be established through synthesis for a specified target device. This makes the proposed architecture suitable for deployment in resource-constrained embedded systems.

IV. COMPUTATIONAL COMPLEXITY ANALYSIS AND EFFICIENCY EVALUATION

Unlike conventional approaches, in which complexity is evaluated only for an individual classification algorithm, this study considers the complexity of the entire functional sequence because the proposed system is combined and includes a software-based feature-extraction stage and a hardware-oriented intelligent-classification stage. This approach is consistent with the design principles of specialized computing systems, whose efficiency is determined not only by the number of arithmetic operations but also by memory organization, parallelism, latency, streaming capability, and suitability for implementation in an embedded or reconfigurable environment.

The overall data-processing procedure in the proposed system consists of several consecutive steps. First, the input video stream or set of images acquired during inspection of a renewable energy facility is converted into a sequence of frames. This stage is necessary because the raw video data cannot be used directly by fuzzy classifiers without preprocessing. A video stream contains many repetitive or

weakly informative frames; therefore, frames are sampled at a specified interval to reduce redundancy. In terms of computational complexity, instead of processing all frames, the system processes only N frames, where $N = \left\lfloor \frac{T}{s} \right\rfloor$, where T is the total number of frames in the video stream and s is the frame-sampling interval. Thus, already at the first step, complexity is reduced by approximately a factor of s because subsequent feature extraction is performed not for every frame but only for the selected subsequence. This directly affects system performance because convolutional image analysis is the most computationally intensive stage.

Each selected frame is preprocessed by resizing it to 224×224 pixels, normalizing channel intensities, and converting the image into a SqueezeNet-compatible format. This stage has linear complexity with respect to the number of image pixels $C_{prep} = O(H \cdot W \cdot C)$, where H and W are the image height and width, respectively, and C is the number of channels. For a standard RGB image after scaling $H = 224$, $W = 224$, and $C = 3$.

Features are extracted using the modified SqueezeNet. The convolutional part of SqueezeNet generates a fixed-dimensional feature vector $d = 4$ for each frame, describing the principal surface-condition classes: crack, erosion, corrosion, and normal condition. The choice of SqueezeNet is nevertheless justified because its Fire modules reduce the parameter count by first compressing the channels in the squeeze layer and subsequently expanding them in the expand layers. Consequently, the network retains its ability to extract informative local surface patterns while reducing memory requirements and the number of multiply-accumulate operations.

Experimental results show that the average processing time per frame is 40-60 ms on a CPU and 8-12 ms on a GPU. For a video stream containing 22202 frames, the average rate was 8.73 frames/s. This confirms that feature extraction determines the principal timing complexity of the entire system. However, after the vector of four features has been generated, subsequent classification is considerably faster because the developed neuro-fuzzy models operate in a compact space of dimension $d = 4$ rather than in pixel space.

The generated feature vector is supplied to the base Hypersector Fuzzy LVQ classification module. In the VHDL implementation, this module performs four parallel calculations of the proximity between the input vector and the class prototypes, after which the score vector is generated and the class with the maximum score is determined. As noted in the cascade-architecture implementation description, Hypersector FLVQ is the simplest and fastest module in the system because it contains no iterative procedures, complex fuzzy-inference logic, or multistage optimization.

For K classes and d features, the complexity of Hypersector Fuzzy LVQ is defined as $C_{FLVQ} = O(K \cdot d)$. In this case, $K = 4$ and $d = 4$; therefore, the number of basic operations is fixed and does not depend on the number of frames once the feature vector has been generated $C_{FLVQ} \approx O(1)$. This means that the module has constant-order algorithmic complexity. However, engineering analysis must consider not only the theoretical $O(1)$ complexity but also the operation structure. For each of the four classes, the dot product between the input vector and the class prototype requires approximately four multiplications and three additions. For four classes, this amounts to approximately 16 multiplications and 12 additions, followed by 3-4 comparisons to select the maximum value. In a parallel hardware implementation, these operations can be performed simultaneously, resulting in very low latency. Hypersector Fuzzy LVQ is therefore suitable for the first cascade level.

If the base classifier has insufficient confidence, the system can engage the FBSB module. In the VHDL implementation, FBSB performs a fuzzy matrix mapping from the feature space to the class space. The complexity of this module is also of order $C_{FBSB} = O(K \cdot d) \approx O(1)$. However, the practical complexity of FBSB is somewhat higher than that of Hypersector Fuzzy LVQ because FBSB uses a more complete matrix transformation followed by a limiting function applied to the intermediate values. FBSB implements four parallel computation paths in which the input vector is multiplied by the corresponding columns of the weight matrix and the results are then constrained to the admissible interval. Therefore, although the complexity order remains constant, actual hardware resource utilization is slightly higher because of the additional limiting operations and the larger number of connections between features and classes.

At the sixth step, the modified Wang-Mendel module can be used to refine the decision. Its role is not limited to improving accuracy; it also provides decision interpretability because the method is based on fuzzy rules. The Wang-Mendel module generates intermediate scores as linear combinations of the input features, followed by scaling, limiting, and determination of the maximum-scoring class. For the implemented version, the complexity can be estimated as $C_{WM} = O(R \cdot d)$, where R is the number of fuzzy rules. If the number of rules is fixed and comparable to the number of classes, the complexity also approaches a constant. In the general case, however, the Wang-Mendel method has potentially higher complexity because the number of rules may increase with the numbers of features and fuzzy terms. If M fuzzy terms are used for each feature, the maximum number of rule combinations can be estimated as $R_{max} = M^d$. The complexity then becomes $C_{WM} = O(M^d \cdot d)$. This is the principal limitation of classical rule-based fuzzy systems as feature-space dimensionality increases. In the developed system, this limitation is mitigated because SqueezeNet generates only four of the most informative features, i.e., $d = 4$. Thus, the modified SqueezeNet not only reduces the data volume but also prevents exponential growth in the number of fuzzy rules in subsequent modules.

At the seventh step, the switching-logic module is activated in the cascade architecture. Its primary purpose is to avoid engaging more complex modules unnecessarily. The maximum score of the base FLVQ classifier is first determined, followed by the second-largest score, after which the confidence margin is calculated $margin = s_{max1} - s_{max2}$ (s_{max1} is the highest class score produced by the base Hypersector FLVQ classifier, s_{max2} is the second-highest class score, that is, the strongest "competitor" of the main class).

If the confidence margin is insufficient, the feature variability is additionally evaluated using the hardware-efficient indicator $V(\mathrm{x})$ (equation 1).

The indicator is used as a routing heuristic rather than as a statistically calibrated estimate of physical noise. Its implementation requires three subtractions, three absolute-value operations, and two additions. Therefore, its

computational complexity is $O(d)$, where $d = 4$, and its hardware cost remains low.

If $V(\mathrm{x})$ exceeds the specified threshold, FBSB is selected as the branch intended for processing strongly nonuniform feature vectors. If the variability level is below the threshold but the FLVQ decision remains uncertain, Wang–Mendel is selected as the refinement classifier.

The complexity of the switching module is low $C_{switch} = O(K + d) = O(8)$. However, the significance of this module lies not in its own arithmetic complexity but in its ability to reduce the average complexity of the entire architecture. For the cascade system, the complexity for one sample can be expressed as:

$$C_{cascade} = \begin{cases} C_{FLVQ} + C_{switc}\ , & m \geq \tau_m, \\ C_{FLVQ} + C_{switch} + C_{FBSB}, & m < \tau_m, V(x) \geq \tau_V, \\ C_{FLVQ} + C_{switch} + C_{WM}, & m < \tau_m, V(x) < \tau_V. \end{cases} \quad (2)$$

where m is the confidence margin of the base Hypersector FLVQ classifier, τ_m is the confidence-margin threshold, and τ_V is the feature-variability threshold.

The average cascade complexity is determined through the probability of activating the auxiliary branch $C_{avg} = C_{FLVQ} + C_{switch} + P_{aux} \cdot C_{aux}$, where P_{aux} is the probability of using the auxiliary classifier and C_{aux} is the complexity of FBSB or Wang-Mendel. If most samples are well separated in the feature space, P_{aux} is small, and the average cascade complexity approaches that of FLVQ. This is a fundamental advantage of the cascade architecture over ensemble approaches in which all modules operate continuously.

The dependence of processing time on the architecture type should also be examined. Because the CNN stage is implemented in software and the VHDL classifiers operate at the RTL level with a latency of approximately one clock cycle, the feature-extraction time and the decision-making time should be considered separately. At a clock frequency of 50 MHz, one cycle is approximately 20 ns, and the VHDL modules generate a decision within one clock cycle. Therefore, after the feature vector has been obtained, the classification time is many orders of magnitude shorter than the CNN processing time for one frame.

As shown in Figure 8, the timing characteristics of the different stages of the proposed intelligent system differ substantially. The feature-extraction stage based on the modified SqueezeNet convolutional neural network makes the largest contribution to the total processing time per sample, whereas the VHDL-implemented neuro-fuzzy classification modules operate with a latency of approximately one clock cycle. A logarithmic time scale is used in the graph to provide a valid comparison because the CNN processing time and the RTL-module operating time differ by several orders of magnitude.

Analysis of the graph shows that feature extraction requires approximately six orders of magnitude more time than the VHDL classifiers. Thus, once the feature vector has been obtained, the classification decision is generated almost instantaneously. This result is fundamentally important for real-time monitoring systems because it separates the computationally intensive semantic image-analysis stage from the ultrafast logical decision-making stage.

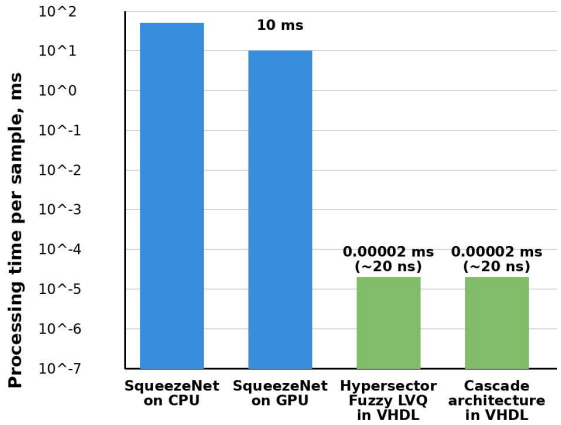


Figure 8. Comparison of processing time per sample at the feature-extraction and classification stages

The obtained results confirm the feasibility of a hybrid architecture in which the convolutional neural network is used as a software-based or hardware-accelerated feature-extraction module, while the neuro-fuzzy classifiers are implemented as compact digital logic. This organization simultaneously provides a highly informative feature space, low decision latency, moderate hardware cost, and suitability for embedded-system implementation. This is particularly important for UAV-based wind turbine blade monitoring, where computational resources are limited and decision delay must be minimized. In such systems, the response speed after feature extraction determines the capability for real-time operation, inspection-route control, and prompt detection of critical surface defects.

V. DISCUSSION

The obtained results confirm the effectiveness of the proposed intelligent system for classifying defects in renewable energy facilities. The system combines feature extraction using a modified SqueezeNet network, the Wang-Mendel, Fuzzy BSB, and Hypersector Fuzzy LVQ neuro-fuzzy classifiers, and cascade decision-making mechanisms. The proposed approach is characterized not only by high classification accuracy but also by a combination of noise robustness, result interpretability, and suitability for implementation in real-time embedded systems.

The analysis of current research showed that most existing studies focus primarily on deep convolutional neural networks, including ResNet [23], EfficientNet [24], YOLO [25-27], and Mask R-CNN [28]. These models demonstrate high accuracy but are characterized by large parameter counts, high computational cost, and limited explainability of their decisions. In many cases, they require powerful GPUs or cloud infrastructure, complicating their use on mobile UAVs and edge platforms.

The closest prototype of the proposed approach is the study in [11], which combined Mask R-CNN with a fuzzy-voting mechanism to classify wind turbine blade defects. In that study, fuzzy logic aggregates the outputs of several deep models, thereby improving classification reliability. However, resource-intensive CNN models remain the core of the system, while the fuzzy component performs an auxiliary function at the final voting stage. By contrast, in the proposed architecture, the neuro-fuzzy methods are direct classifiers and participate in decision formation at all cascade levels.

The principal novelty of the proposed cascade is its transition from output aggregation to conditional sample-dependent decision routing. In conventional ensembles and fuzzy-voting schemes, several models usually process the same input sample, and the final result is obtained by averaging, voting, weighting, or fuzzy aggregation of their outputs. In contrast, the proposed architecture does not treat the classifiers as equivalent parallel voters. The Hypersector Fuzzy LVQ module acts as a fast primary classifier, whereas the Fuzzy BSB and Wang–Mendel modules are activated only for samples for which the primary decision is insufficiently reliable.

This makes the proposed cascade fundamentally different from fuzzy-voting-based prototypes. The switching logic analyses two complementary criteria: the confidence margin of the primary FLVQ decision and the feature-variability indicator of the compact input vector. The confidence margin determines whether the base classifier can be trusted, while the feature-variability indicator determines which auxiliary classifier is more appropriate for the uncertain sample. Therefore, Fuzzy BSB is selected for samples with stronger feature nonuniformity, where robustness to perturbations and overlapping feature regions is required, whereas the Wang–Mendel module is selected for uncertain but more stable samples, where rule-based refinement and interpretability are preferable.

Thus, the novelty of the cascade lies not in the isolated use of Hypersector Fuzzy LVQ, Fuzzy BSB, or Wang–Mendel classifiers, but in their functional integration within a unified hardware-oriented routing architecture. The proposed system combines three complementary properties: fast geometric separation, robustness to feature disturbances, and interpretable rule-based decision refinement. Unlike existing ensemble approaches, this interaction is controlled by explicit hardware-efficient criteria and can be implemented as deterministic switching logic in VHDL. As a result, the cascade provides not only improved classification reliability but also a structure suitable for real-time FPGA-oriented embedded implementation.

The proposed architecture achieved an accuracy of 98.33% on the examined test subset. However, this value should not be directly compared with accuracy values reported for CNN- and YOLO-based approaches because the studies use different datasets, class definitions, and experimental protocols. The main advantage demonstrated in this study is the compact hardware-oriented organization of the decision stage rather than absolute superiority over published deep-learning models.

This adaptive routing mechanism also has practical significance. Unlike CNN-based ensembles, in which all models operate simultaneously for every sample, the proposed system performs conditional refinement only for problematic samples. At the RTL level, this principle creates the basis for future reductions in dynamic power consumption, provided that inactive branches are disabled through clock-enable or operand-isolation mechanisms during FPGA synthesis.

Another advantage is the possibility of hardware implementation on FPGA platforms [29]. Active-HDL simulation showed that a decision is generated within one clock cycle, which corresponds to approximately 20 ns at 50 MHz. This latency refers exclusively to the decision stage and should not be directly compared with end-to-end CNN inference latency. The architecture stores compact prototype and parameter matrices; however, their exact memory footprint and FPGA resource utilization must be determined through synthesis for a specified target device.

The use of VHDL in this study is also important from the perspective of further deployment on FPGA platforms. In the proposed system, VHDL is not used only as a simulation language; it serves as an intermediate hardware description level that transforms the trained neuro-fuzzy cascade from an algorithmic model into a synthesizable digital architecture. This is essential because FPGA implementation requires explicit representation of arithmetic operations, data widths, synchronization signals, routing logic, memory organization, and parallel computational paths. Therefore, the VHDL model makes it possible to verify whether the proposed cascade can be represented as deterministic register-transfer-level logic before moving to physical synthesis.

The necessity of VHDL is particularly justified by the structure of the proposed cascade. The system contains several classifier modules, a confidence-margin calculation block, a feature-variability estimation block, and a switching mechanism that selects the final processing route. Such logic must operate synchronously and predictably in real time, especially in UAV-based inspection systems where decision delay after feature extraction should be minimized. VHDL allows these operations to be described as parallel hardware processes rather than sequential software procedures. As a result, the decision stage can be organized with low latency, fixed timing behavior, and reduced dependence on a general-purpose processor.

Another important aspect is that the VHDL description prepares the architecture for subsequent FPGA synthesis and implementation. At the current stage, Active-HDL simulation confirms the functional correctness of sample-dependent routing and synchronous decision generation. However, functional simulation alone does not provide complete information about hardware resource consumption, maximum clock frequency, timing closure, power consumption, or the use of LUTs, flip-flops, DSP blocks, and embedded memory. These parameters can only be obtained after synthesis for a selected FPGA device [30, 31]. Thus, the developed VHDL model should be considered a necessary prerequisite for the next research stage, which will include physical FPGA synthesis, post-synthesis timing analysis, resource-utilization estimation, and hardware-in-the-loop verification.

Consequently, the VHDL implementation strengthens the practical significance of the proposed method. It demonstrates that the neuro-fuzzy cascade is not limited to software-based classification but can be further transformed into a compact reconfigurable hardware module. This is important for embedded monitoring systems operating under limited computational resources, where FPGA-based implementation can provide parallel processing, deterministic latency, energy-efficient inference, and reliable integration with sensor interfaces or hardware-accelerated CNN feature extractors.

Compared with multisensor systems based on RGB and thermal fusion [32-34], the proposed approach does not require complex synchronization of heterogeneous data sources and can operate with a siformative feature space generated by a single modified CNN model. This simplifies the system architecture and reduces hardware requirements without substantially degrading classification quality.

Interpretability is another notable advantage. Most current deep-learning models remain difficult to explain, whereas the modified Wang-Mendel method can generate IF-THEN fuzzy

rules that enable analysis of the reasons underlying a decision. This is an important advantage for monitoring critical energy infrastructure, where decision reliability and explainability are no less important than classification accuracy.

Thus, the results show that the proposed cascade architecture not only achieves high accuracy but also provides a better balance among processing speed, robustness, interpretability, and hardware efficiency than existing analogues and the closest prototype based on fuzzy voting and deep CNN models. This confirms the feasibility of using neuro-fuzzy cascade systems for real-time monitoring and diagnosis of renewable energy facilities.

VI. CONCLUSIONS

This study developed a hardware-oriented neuro-fuzzy cascade architecture for the real-time classification of defects in renewable energy facilities. The proposed system integrates feature extraction using a modified SqueezeNet network with three complementary classifiers: Hypersector Fuzzy LVQ, Fuzzy BSB, and a modified Wang–Mendel network. The developed switching mechanism analyses the confidence margin of the primary FLVQ decision and a hardware-efficient feature-variability indicator to select an appropriate processing route.

The principal scientific contribution of this study is the adaptive organization of previously developed neuro-fuzzy classifiers within a unified sample-dependent decision architecture. The proposed approach extends the individual Hypersector Fuzzy LVQ, Fuzzy BSB, and Wang–Mendel models by introducing a formal routing mechanism that selects a classifier according to the confidence of the primary decision and the characteristics of the input feature vector. Thus, the novelty lies in the conditional interaction of complementary classifiers and its hardware-oriented representation rather than in the individual classification modules themselves.

The experimental results demonstrated that the conditional activation of auxiliary classifiers improves classification reliability while avoiding the use of unnecessary auxiliary decisions. The cascade architecture correctly classified test samples, corresponding to an accuracy of 98.33%. Owing to the limited test-set size, this result represents functional validation of the proposed architecture. Hypersector Fuzzy LVQ provides rapid primary classification, Fuzzy BSB increases robustness to noise and overlapping classes, and the Wang–Mendel network refines uncertain decisions while supporting their interpretation through fuzzy rules.

Hardware-oriented inference representations of the trained neuro-fuzzy classifiers and the cascade switching logic were implemented in VHDL and functionally verified in the Active-HDL simulation environment. Model training, fuzzy-rule generation, and parameter estimation were performed offline and were not included in the RTL implementation. After the feature vector has been generated, the classification modules provide a decision within approximately one clock cycle. The complexity analysis showed that conditional routing reduces the number of auxiliary decisions logically required for each sample. The current parallel RTL implementation does not yet demonstrate corresponding reductions in physical FPGA resources or power consumption.

Thus, the proposed architecture provides an effective balance among classification accuracy, computational efficiency, noise robustness, interpretability, and low decision latency. It can serve as the basis for embedded FPGA-oriented monitoring systems for wind turbine blades and other renewable energy facilities. The feature-variability criterion used in the current implementation is an empirical fixed-order routing heuristic and should not be interpreted as a direct physical measurement of image noise. The limitations of the study also include the relatively small test subset and empirical threshold selection.

Further research should include FPGA synthesis and implementation, evaluation of hardware resource utilisation and power consumption, and validation using larger datasets obtained under real operating conditions.

References

[1] "Solar panel clean and faulty images," Kaggle Dataset, [Online]. Available at: https://www.kaggle.com/datasets/pythonafroz/solar-panel-clean-and-faulty-images.

[2] "Wind Turbine YOLOv8 Training," Kaggle Notebook, [Online]. Available at: https://www.kaggle.com/code/hunhcgiatn/wind-turbine-yolov8-training?select=NordTank586x371.

[3] B. Altice, E. Nazario, M. Davis, M. Shekaramiz, T.K. Moon, M.A.S. Masoum, "Anomaly detection on small wind turbine blades using deep learning algorithms," *Energies*, vol. 17, issue 5, article 982, 2024. https://doi.org/10.3390/en17050982.

[4] M. Davis, M. Shekaramiz, M.A.S. Masoum, A. Seibi, "Identification and localization of wind turbine blade defects using YOLO and Mask R-CNN," *Applied Sciences*, vol. 14, issue 14, article 6319, 2024. https://doi.org/10.3390/app14146319.

[5] M. Memari, M. Shekaramiz, M.A.S. Masoum, A.C. Seibi, "Data fusion and ensemble learning for advanced anomaly detection using multi-spectral RGB and thermal imaging of small wind turbine blades," *Energies*, vol. 17, issue 3, article 673, 2024. https://doi.org/10.3390/en17030673.

[6] A. Rahman, "Solar panel surface defect and dust detection: Deep learning approach," *Journal of Imaging*, vol. 11, issue 9, article 287, 2025. https://doi.org/10.3390/jimaging11090287.

[7] A. Karakan, "Detection of defective solar panel cells in electroluminescence images using deep learning," *Sustainability*, vol. 17, issue 3, article 1141, 2025. https://doi.org/10.3390/su17031141.

[8] D. Dwivedi, K.V.S.M. Babu, P.K. Yemula, P. Chakraborty, M. Pal, "Identification of surface defects on solar PV panels and wind turbine blades using attention-based deep learning model," *Engineering Applications of Artificial Intelligence*, vol. 131, article 107836, 2024. https://doi.org/10.1016/j.engappai.2023.107836.

[9] W. Yaïci, E. Entchev, "Adaptive Neuro-Fuzzy Inference System modelling for performance prediction of solar thermal energy system," *Renewable Energy*, vol. 86, pp. 302–315, 2016, https://doi.org/10.1016/j.renene.2015.08.028.

[10] V. Mukhin, V.Zavgorodnii, V.Liskin, et al. "A model for classifying information objects using neural networks and fuzzy logic". Sci Rep, vol. 15, 15904, 2025. https://doi.org/10.1038/s41598-025-00897-4.

[11] R. Pratt, A. Clark, M.A.S. Masoum, A. Seibi, "Defect detection and classification on wind turbine blades using deep learning with fuzzy voting," *Machines*, vol. 13, issue 4, article 283, 2025. https://doi.org/10.3390/machines13040283.

[12] L. Dubchak, "Adaptive cascade bagging of neuro-fuzzy models for the classification of defects in renewable energy facilities," *Advances in Cyber-Physical Systems*, vol. 11, issue 1, pp. 26–32, 2026. https://doi.org/10.23939/acps2026.01.026.

[13] E. Lughofer, "Evolving fuzzy and neuro-fuzzy systems: Fundamentals, stability, explainability, useability, and applications," *in: Handbook on Computer Learning and Intelligence, World Scientific*, Singapore, 2022, pp. 133–234. doi: 10.1142/9789811247323_0004.

[14] F. M. Siraj, S. T. K. Ayon, M. A. Samad, J. Uddin, K. Choi, "Few-shot lightweight SqueezeNet architecture for induction motor fault diagnosis using limited thermal image dataset," *IEEE Access*, vol. 12, pp. 50986–50997, 2024, https://doi.org/10.1109/ACCESS.2024.3385430.

[15] L.-X. Wang, J.M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, issue 6, pp. 1414–1427, 1992. https://doi.org/10.1109/21.199466.

[16] A. Hichri et al., "Genetic-algorithm-based neural network for fault detection and diagnosis: Application to grid-connected photovoltaic systems," *Sustainability*, vol. 14, issue 17, article 10518, 2022. https://doi.org/10.3390/su141710518.

[17] A. Eskandari, M. Aghaei, J. Milimonfared, A. Nedaei, "A weighted ensemble learning-based autonomous fault diagnosis method for photovoltaic systems using genetic algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 144, article 108591, 2023. https://doi.org/10.1016/j.ijepes.2022.108591.

[18] M. Mansouri, K. Dhibi, H. Nounou, M. Nounou, "An effective fault diagnosis technique for wind energy conversion systems based on an improved particle swarm optimization," *Sustainability*, vol. 14, issue 18, article 11195, 2022. https://doi.org/10.3390/su141811195.

[19] A. Sabati, R. Bayindir, J. Rahebi, "Photovoltaic panels fault detection with convolutional neural network and Bitterling Fish Optimization (BFO) algorithm," *International Journal of Computational Intelligence Systems*, vol. 18, article 239, 2025. https://doi.org/10.1007/s44196-025-00984-4.

[20] L. Dubchak, Y. Bodyanskiy, O. Savenko, V. Kochan, A. Sachenko, "Intelligent system for real-time detection and classification of solar panel defects," *Radioelectronic and Computer Systems*, issue 3(115), pp. 111–121, 2025. https://doi.org/10.32620/reks.2025.3.08.

[21] L. Dubchak, B. Rusyn, C. Wolff, T. Ciszewski, A. Sachenko, Y. Bodyanskiy, "Hypersector-based method for real-time classification of wind turbine blade defects," *Energies*, vol. 19, issue 2, article 442, 2026. https://doi.org/10.3390/en19020442.

[22] O. Kilincceker, E. Turk, F. Belli, M. Challenger, "Model-based ideal testing of hardware description language programs," *Software and Systems Modeling*, vol. 21, pp. 1209–1240, 2022. https://doi.org/10.1007/s10270-021-00934-6.

[23] B. Koonce, ResNet 50, in: *Convolutional Neural Networks with Swift for TensorFlow*, Apress, Berkeley, 2021. https://doi.org/10.1007/978-1-4842-6168-2_6.

[24] B. Koonce, EfficientNet, in: *Convolutional Neural Networks with Swift for TensorFlow*, Apress, Berkeley, 2021. https://doi.org/10.1007/978-1-4842-6168-2_10.

[25] P. Jiang, D. Ergu, F. Liu, Y. Cai, B. Ma, "A review of YOLO algorithm developments," Procedia Computer Science, vol. 199, pp. 1066–1073, 2022. https://doi.org/10.1016/j.procs.2022.01.135.

[26] J. Terven, D.-M. Córdova-Esparza, J.-A. Romero-González, "A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Machine Learning and Knowledge Extraction*, vol. 5, issue 4, pp. 1680–1716, 2023. https://doi.org/10.3390/make5040083.

[27] M. Hussain, "YOLO-v1 to YOLO-v8, the rise of YOLO and its complementary nature toward digital manufacturing and industrial defect detection," *Machines*, vol. 11, issue 7, article 677, 2023. https://doi.org/10.3390/machines11070677.

[28] X. Bi, J. Hu, B. Xiao, W. Li, X. Gao, "IEMask R-CNN: Information-enhanced Mask R-CNN," *IEEE Transactions on Big Data*, vol. 9, issue 2, pp. 688–700, 2023. https://doi.org/10.1109/TBDATA.2022.3187413 3.

[29] D. Cock, A. Ramdas, D. Schwyn, M. Giardino, A. Turowski, Z. He, N. Hossle, D. Korolija, M. Licciardello, K. Martsenko, R. Achermann, G. Alonso, T. Roscoe, "Enzian: An open, general, CPU/FPGA platform for systems software research," *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS"'22)*, Lausanne, Switzerland, February 28–March 4, 2022, pp. 434–451. https://doi.org/10.1145/3503222.3507742.

[30] R. Neris, A. Rodríguez, R. Guerra, S. López, R. Sarmiento, "FPGA-based implementation of a CNN architecture for the on-board processing of very high-resolution remote sensing images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 3740–3750, 2022, https://doi.org/10.1109/JSTARS.2022.3169330.

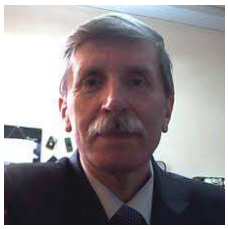
[31] S. S. Lingala, S. Bedekar, P. Tyagi, P. Saha, P. Shahane, "FPGA based implementation of neural network," *Proceedings of the 2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)*, Chennai, India, 2022, pp. 1–5, https://doi.org/10.1109/ACCAI53970.2022.9752656.

[32] M. Brenner, N.H. Reyes, T. Susnjak, A.L.C. Barczak, "RGB-D and thermal sensor fusion: A systematic literature review," *IEEE Access*, vol. 11, pp. 82410–82442, 2023. https://doi.org/10.1109/ACCESS.2023.3301119.

[33] Y. Wang, H.K. Chu, Y. Sun, "PEAFusion: Parameter-efficient adaptation for RGB-thermal fusion-based semantic segmentation," *Information Fusion*, vol. 120, article 103030, 2025. https://doi.org/10.1016/j.inffus.2025.103030.

[34] W. Zhou, S. Dong, C. Xu, Y. Qian, "Edge-aware guidance fusion network for RGB-thermal scene parsing," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, issue 3, pp. 3571–3579, 2022. https://doi.org/10.1609/aaai.v36i3.20269.

***LESIA DUBCHAK** received the Master's degree in computer systems and networks from Ternopil Academy of National Economy, Ternopil (2004), the Candidate of Technical Science degree in computer systems and components from Ternopil National Economy University (2013). From 2003 to 2005 she was an Engineer in Ternopil State Economic University, and from 2005 to 2013 she was a lecturer, from 2013 to 2022 she was a doctoral assistant at the Department of Computer Engineering of Ternopil National Economic University. From 2022 until now, she has been the Head of the Department of Computer Engineering of West Ukrainian National University.*



***BOHDAN RUSYN** received the D.Sc. degree in information technologies, in 1997. He is currently a Full Professor and the Head of the Remote Sensing Information Technologies Department, Karpenko Physico-Mechanical Institute, NAS of Ukraine. His research interests include computer vision, pattern recognition, 3D reconstruction, and digital signal processing. He serves as an Editor for International Journal of Computing and Mathematical Modeling and Computing journal.*