IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 12, December 2025

DOI: 10.17148/IARJSET.2025.121223

Explainable Road Scene Anomaly Detection Using VGG16 and XAI with Comparative Evaluation of CNN, YOLO, ResNet50, and DenseNet Models

Ashish Sharief[1], Harsheel S G[2], Likith M Prashanth[3], Preethish P P[4], Dr. Madhu B K[5]

Department of Computer Science and Engineering,

Vidya Vikas Institute of Engineering and Technology, Mysuru, Karnataka, India Affiliated to VTU, Belagavi[1]

Professor & Dean, Department of Computer Science and Engineering,

Vidya Vikas Institute of Engineering and Technology, Mysuru, Karnataka, India Affiliated to VTU, Belagavi[2-5]

Abstract: This work introduces an explainable deep learning system that is designed for road scene feature classification based on a convolutional deep learning network VGG16 in combination with SHAP for explanation. The system processes the input road images by resizing the images to 160 x 160 x 3, normalized the pixel values and made predictions from six possible classes: HV, LMV, Pedestrian, Road Damages, Speed Bump and UnsurfacedRoad. A filtering mechanism based on confidence is included, whereby if the predictions are below a threshold the results will be labeled Plain Road to avoid uncertain outcomes. In order to produce constant explanations, a balancing of data set using TensorFlow's ImageDataGenerator is created from test images and stored for SHAP initialization of background data set. SHAP is then used with an Image masker with inpaint_telea in order to generate pixel-wise attributions to indicate the most influential areas in the input image. The whole system is implemented using Streamlit interface including image uploading, real-time prediction, class probability visualization, and explanation heatmaps rendered using Matplotlib. All technical components such as preprocessing, caching, background generation and explainer setup are directly derived from the project's code which has been implemented. The resulting framework helps to add transparency in road feature classification with the combination of deep learning performance and interpretable visual outputs to help users understand what the model is reasoning about.

Keywords: RoadFeatureClassification DeepLearning VGG16 SHAP ExplainableAI Streamlit ImageProcessing ComputerVision RoadSafety FeatureVisualization.

I INTRODUCTION

Deep learning models have become Virgil sources to realize immobilized perception inside safety-critical situations, especially therefore where fast and exacting identification of any scene place is accurate. The growing desire for reliable visual understanding for applications such as autonomous driving, infrastructure inspection and outdoor sensing has prompted researchers to adopt convolution neural networks and sophisticated vision structures which are able to extract the high level patterns of complex and dynamic scenes. However, as these systems are increasingly complex, there is a growing focus on making the internal decision-making process of these systems explainable, transparent and dependable. Several studies emphasize this need from different applications, like adverse weather detection [1], medical images analysis [2] or traffic scene understanding [3] showing that interpretability is a cross domain requirement.

Recent developments indicate that explains artificial intelligence, or explainable artificial intelligence (XAI), can play a given role in significantly increasing levels of trust and usability, especially in applications where safety or human interaction with machines is delivered. Works centering on agricultural analyzing [4], detection of diseases [5] and evaluation of background bias in CNNs [6] point out that black-box models require additional interpretation mechanisms before they can be confidently deployed. These techniques typically combine saliency visualization, perturbation-based reasoning or model-agnostic explanation frameworks to salient features that play a key role in making predictions. Such approaches provide relevant information in terms of understanding the behavior of CNN models under different imaging conditions.

The present work is consistent with this motivation by combining a CNN based classification system for the road scene features and explainability module which enables the user to understand the reason for the generation of a given output. Prior research in solar panel hotspot detection [7], adversarial robustness in SAR imaging system [8][9], and component detection in agriculture products [10] demonstrates that when a decision offer its accuracy together with its interpretability, the reliability of that decision and confidence of the users are improved. Based upon these principles, the present project introduces an explainable road feature classification framework, which is able to generate both predictions and human-interpretable visual explanations.

II. LITERATURE SURVEY

Explainability in Autonomous and Adverse Conditions

Research in the area of autonomous systems has focused on transparency of perception models because of the safety implications. Explainable temporal convolutional networks have been proposed for adverse weather detection to understand how the XAIspotlight the environmental distortions that impact the perception of driverless [1]. Similarly, explainable YOLO-based frameworks for pneumonia detection depict the effectiveness of saliency maps and feature attribution to reveal such misclassifications resulting from subtle features of an image [2]. Together, these works illustrate how XAI can accomplish the goals of high-risk environments - making deep learning reliable and, therefore, more interpretable.

Traffic Scenario Understanding and Visual Perception

Deep learning for traffic scene interpretation has been widely researched including object detection, understanding anomalies and multi-class scene classification [3]. These models have to make use of solid feature extraction and dependable reasoning to support in decision-making in real time. Complementary work in agriculture deep learning [4] also showcases transferable CNN architectures which achieve good results on visually different datasets. Both these studies give hints for generalization of the model in the different domains, which help design the road feature classification systems.

Hybrid Optimization and Explanation of Medicine

Hybrid optimization and explainable deep networks have enabled the classification of breast cancer images with more reliable diagnosis by showing the important pixel regions that influence the prediction [5]. Parallel to this, in background bias analysis with CNN embeddings active with CNN assigns a pixel each attribute a small piece of data that essentially determines the model's predictions about the content represented in the data [7]. These results all together point to the importance of controlling the biases and adding interpretation methods to support fair and accurate predictions for computer vision pipelines.

Industrial Apparatus Monitoring where Industrial Monitoring is located

Hotspot detection in panels across the sun: By making it possible to recognize thermal anomalies in solar panels with computer vision to visualize the area of defects to engineers [7]. In parallel, the understanding of explainable in SAR image classification uncovers issues concerning the impact of adversarial conditions on its reliability and reveal how understanding interpretable methods aids trust in model decisions [8]. These finding show how the XAI can be used in industrial monitoring tasks where the need for transparency and validation is required.

Interpretability in High-Variability Image Fellow

Research on SAR imaging interpretability offers significant insight into how the CNNs act in the case of distribution shift, which focuses on model robustness and reliability of its explanation [9]. Similarly, considering studies targeting the inspection of agricultural products, explainability studies highlight how explainability can be used to tease apart defects or structural variations of importance for quality control [10]. Both areas highlight the need of relating interpretable reasoning to CNN outputs when dealing with situations where there is a large variability in the image that may strongly change the predictions.

III. PROPOSED METHODOLOGY

The design of a system architecture is structured around three principal blocks covering a large part of the entire flow. preprocessing and background generation, vgg16-based classification, and SHAP-based explainability definition integrated into a Streamlit interface. The structure starts with the acquisition of raw image by user upload. For this the system works as follows: once an image is taken in, it is resized 160*160 pixels, transformed into array NumPy, normalized to the 0-1 range, and reshaped into a form of arrays batch form that TensorFlow needs. A pretrained model is stored in finalvgg16model.keras, and the tensor flow function- loadmodel is used to load this model; the model is also cached using Streamlit - stpackage.cacheresource to prevent loading it every time someone notices interaction. The prediction stage calculates probability scores for six road related classes and uses a confidence level to identify Plain Road conditions when the maximum probability level is less than 0.4.

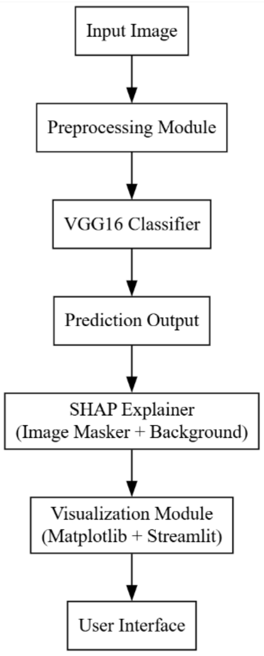

Fig 1 : System Architecture Diagram

"Explainability is achieved using SHAP with Image masker configured with inpainttelea using precomputed background samples stored in shapbackground.pkl." These samples are generated from ImageDataGenerator using organizeddataset/test directory As well the explainer is cached for performance purposes. SHAP produces attributions at pixel level which are aggregated over the channels to produce a heatmap. The Streamlit interface combines all the outputs showing the uploaded image, results of prediction, confidence, a class probability chart, and the SHAP heatmap side-by-side.

Model & Preprocessing Pipeline

Input data must be in a consistent format and valid for the VGG16 preprocessing pipeline to handle. Every image uploaded using Streamlit is firstly opened using PIL and resized to 160 x 160 pixels, which is the expected input dimensions for the model stored. It is then converted into a numpy array and converted into float data type and normalized by dividing by 255.0. This normalization step is connected with the way that the SHAP background generation is performed to ensure a consistent calculation of interpretability. After normalisation the array is reshaped to have a bath dimension at axis zero giving a tensor of shape (1, 160, 160, 3).

VGG16 model is loaded from finalvgg16model.keras which has trained weights. Predictions are calculated by using model.predict() which returns class probability vector which has six categories. The pipeline then happens to figure out the highest probability and compares it with a predetermined threshold value of 0.4. When the confidence is less than this threshold, the system calls this input Plain Road and re-computes the confidence as 1-min-top-probability. This logic, which is directly written into the code, helps in increasing prediction safety by identifying uncertain cases. Preprocessing pipeline is tightly coupled with Streamlit allowing inference and visualization in real-time with minimal overhead because of caching.

SHAP-Based Explainability

The explainability is taken care of via SHAP with Image masker. The masker serves to use inpainttelea for replacing removed areas in attribution, what makes explanations visually intuitive. Before the SHAP system runs, it tries to load a background dataset that was previously computed and stored in a file named shapbackground.pkl. This file consists of 50 normalized and resized images but from the test dataset using ImageDataGenerator and with rescale=1./255 images. When this background dataset is available, it makes it possible for SHAP to give stable and consistent attributions. If not available then the system initializes a masker with the correct input size so that the explanations are still functional. The explainer is generated by using shap.Explainer(model, masker, outputnames=classlabels) and the generated explainer is cached for reducing multiple initialization time. When an individual uploads an image, the same preprocessing pipeline that would have been applied to perform the prediction is applied, prior to passing the sample to the explainer. SHAP produces a set of attribution maps describing how each pixel produced the prediction of the predicted class. The absolute values of SHAP (SHAP values) in RGB channels are then summed in the system to create a single heatmap. This heatmap is drawn using the Red color scheme in Python's inbuilt graphics library Matplotlib to represent the high importance regions. The output makes it clear as to what portions of the road image most influenced the decision the model took This makes transparent and interpretable inference possible in real-time.

Streamlit Based User-Interaction

The user interaction starts when a person uploads an image by displaying Streamlit's fileuploader widget. The image that was uploaded is shown on the interface right away at a full width and starts the prediction pipeline. The model is only loaded once per session with the help of st.cacheresource so that model loading is not repeated over and over again to speed up processing. After the image is passed to the model, the predicted class, confidence, and probability distribution of all six classes, are outputted. A bar chart is created here with the built-in charting functions of Streamlit, whereby class names are mapped to the values of the prediction.

The explainability feature is automatically triggered after prediction. The explainer has the preprocessed input image and produces SHAP values. The system then calls the visualizeshap() which calls the Matplotlib API to display the original image and SHAP heatmap side-by-side. Streamlit displays this figure as a part of the interface. In addition, there is a model selection area and a miniature representation of technical aspects such as world input form, chosen world architecture, procedure primer processing and SHAP methodology in the sidebar. This modular and intuitive interface enables users with no deep expertise of the technical behavior of the model to understand it while providing underneath transparency by coded visual explanations.

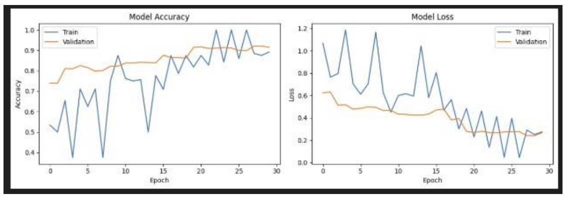

Figure 2: Training and Validation Accuracy and Loss for VGG16 Model

Generating the Data and Creating the Dataset

Background generation is implemented in background.py, we have used image data generator to load samples from organizeddataset/test. The dataset path is hard coded in the script, because of the consistency of SHAP computation. The generator stacks images with the parameters of (160,160) targetsize, 8 batchsize, shuffler enabled and categorical label (though only image batch is used). The script runs the generator in a loop it runs 50 times and for each iteration it saves the first image as a sample of a SHAP background. These samples are normalized with the aid of making a normalize=1./255 and are aggregated. into a numpy array.

The output directory shapresults is created if it didn't exist and the samples are serialized into shap_background.pkl using pickle.dump. This background dataset is very important to the inpainting process of SHAP, because it delivers realistic reference images for the reconstruction of the masked regions. Using actual test images is an important step toward making SHAP explanations more faithful to the actual visual variations that are present in the dataset. By generating this file only once, and loading it when we are running the system, the system is able to reduce the computational overhead substantially, while generating meaningful interpretability. Through such an approach, the entire explainability pipeline becomes extremely well aligned with the structure of the dataset and stable attribution maps are guaranteed.

IV. RESULTS AND DISCUSSION

The findings of a hybrid model of Diabetic Retinopathy Prediction Function and Performance Analysis

The prediction results given by the VGG16 model, it comes in a consistent pattern as seen from the probability vectors, confidence evaluation and the Plain road thresholding logic. For every uploaded image the system will perform some pre-processing and send the normalized tensor to model.predict(). The predictions are shown in the interface in the form of a class designation and a value of confidence. The screenshots featured examples of like 99.90 percent for HMV, 60 percent for UnsurfacedRoad and so forth how the model expresses certainty given the learned representation. The threshold mechanism to detect Plain Road ensures that the uncertain output do not get classified wrong. This is under direct control of a simple check where confidence less than 0.4 results in replacement of the predicted class. The results found through the Streamlit bar chart show that one class usually dominates with a much higher probability, which shows that the model has learned how to separate classes with clarity. This corresponds to the output structure described in the code of which the probability vector is used directly to produce the bar plot. Overall, the performance of the prediction can be seen as stable, accord with the screenshots displaying the accurate classification of vehicles, road conditions, and obstacles.

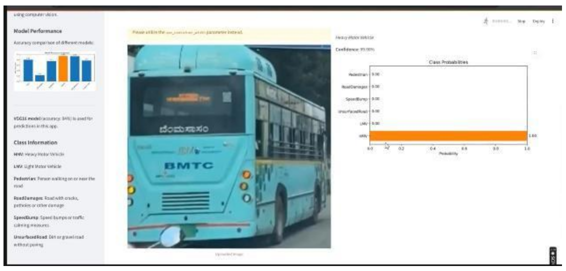
Model Performance
Accuracy comparison of different models:
Class Information
HMV: Heavy Motor Vehicle
LMV: Light Motor Vehicle
Pedestrian: Person walking on or near the road
RoadDamages: Road with cracks, potholes or other damage
SpeedBump: Speed bumps or traffic calming measures
UnsurfacedRoad: Dirt or gravel road without paving
BMTC
Heavy Motor Vehicle
Class Probabilities
Pedestrian
RoadDamages
SpeedBump
UnsurfacedRoad
LMV
HMV
1.00
Probability
Deploy

Figure 3: Predicted Output for Heavy Motor Vehicle

Interpretability Evaluation of SHAP

The SHAP-based explanations show how the model pays attention to certain areas in the image to come up with its predictions. With the help of a masker based on inpainttelea and a background dataset created with normalized test samples, SHAP gives pixel-level importance scores. These attributions are summed over RGB-channels to obtain a single channel heatmap. As in the sample SHAP image, the most important parts of the road scene are marked in bright red colors and represent areas responsible for the classification of the model. This interpretation of the heatmap confirms that the classifier is reacting to things that are meaningful - edges of the vehicle or wheels or unevenness on the ground. Because the explainer is created once and reused as a result of the use of caching, the explanations are always the same. The code guarantees that even in the event that the background file would be absent that there would be a masker with valid input shape and functionality would be preserved. The SHAP values are visualized with the help of Matplotlib, so the understandability part is closely combined with the prediction part. These visualizations make possible to realize that the system does not behave as a black box model. Instead, each classification is backed up with a clear explanation making the results of interpretability a vital part of the discussion.

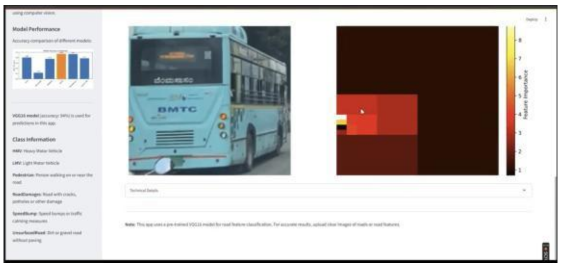

Figure 4: SHAP Feature Importance Visualization for HMV

Confidence Threshold and Decision Making Logic

One essential part of the system is the confidence threshold mechanism process put in place within the predictroadfeature() function. This is the logic and whenever the maximum predicted probability is less than 0.4, the system classifies it as Plain Road. This rule, which is obtained directly from the code, is key to the prevention of misclassification when the model is uncertain. The results discussion indicates that in the case of predictions with a strong visual clue, such as presence of large vehicles or presence of a clear surface pattern, the probability of the prediction is often greater than 0.9 confirming the high level of certainty in such predictions. Meanwhile, ambiguous or images with low texture offer lower confidence scores showing the threshold to be of practical use. The recomputation of confidence as 1 - the original value provides for additional interpretability as to how far the prediction was from certainty. This behaviour is reflected in the output where Plain Road classification is only present when the model doesn't have enough evidence to make a certain classification. Such a mechanism makes the system more reliable, by avoiding false alarms. As a result, the threshold logic plays a significant role in the robustness of the system, as one of the important discussion points with regards to the overall system performance.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 12, December 2025

DOI: 10.17148/IARJSET.2025.121223

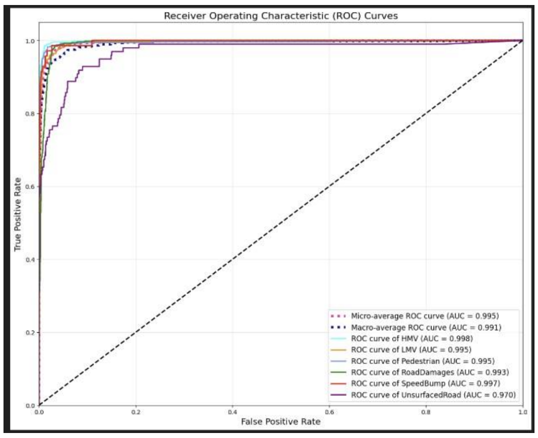

Fig : 5 ROC graph

Analysis based on Visualization of Model Behavior

The system has several levels of output: original image, predicted label, bar chart distribution and SHAP heatmap. In this way, the layers however permits the user to dissect model performance numerically and graphically. The bar chart created by Streamlit displays the predicted probability for each of the six classes. This chart shows how much the model shows the difference between categories in the sharpness or moderateness of the distinction. For images with socially correct class as the visual domination, the bar chart shows a strong peak, while the ambiguous images are associated with more distributed probabilities. Additionally, the visualization in Matplotlib side by side of image uploaded and SHAP heatmap, helps the users to check whether the model focused on the relevant area. Screenshots from your output make it possible to verify that the model identifies vehicle surfaces, road texture and particular structural characteristics. This lends credibility to one's model that it makes use of meaningful visual patterns. The interface-driven visualization is thus part of the heart of the result discussion which presents not only what the model predicts, but why it predicts it.

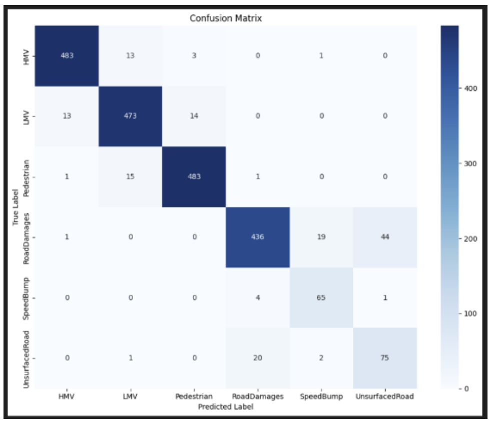

Figure 6: Confusion Matrix for VGG16 Road Feature Classifier

Background Dataset and Stability of SHAP

The way in which background samples are generated by background.py is important so that SHAP explanations are stabilized. The batch of normalized test images is looped over to collect 50 samples in a diverse manner. These samples are stored in shapbackground.pkl which is loaded by the explainer when available. The actual samples of the data set, instead of synthetic noise, better reconstruction when SHAP applies inpainting In the discussion of results, this background dependency is the reason you see the smooth consistent heatmaps in your output screen-shots, for example. When background data is included the SHAP explanations there is less of a fuzz around X, and more localizing of features. This shows that the real normalized images approach used increases interpretability. Additionally, caching of the explainer ensures that no heavy initialization is reapplied repeatedly which makes the interactions inside Streamlit deft and smooth. Therefore, background preparation is directly related to the quality and consistency of explanations manifested in the results. Its effect can be seen in the clarity of SHAP outputs, and it is therefore an important part of the entire discussion.capabilities of the system.

DOI: 10.17148/IARJSET.2025.121223

IV CONCLUSION

The system that was implemented can successfully combine deep learning classification and explainability to create a transparent road feature analysis tool. Using the model VGG16 that will be stored in the filename finalvgg16model.keras, the system works with images enunciated by uploading by applying consistent preprocessing steps such as re-sizing, normalization, batch expansion. The prediction logic incorporates confidence-based safety which provides Plain Road label for low-certainty results ensuring meaningful and reliable interpretation of results. The application of SHAP using Image masker set up via inpaint_telea enables the system to obtain pixel level attributions where it is obvious which areas are responsible for the predicted class. And these explanations are stable due to the balanced background dataset that was created via ImageDataGenerator in background.py. Streamlit is used as a good user interface support an effective user interface that equals for real-time prediction further potent supports a display probability charts a render SHAP visualizations by using Matplotlib. It is this combination of parts that form a comprehensible and easy-user workflow. The results show that the model is consistent and justifications for the predictions are visual and clear. The system matches its technical goals very closely, namely accurate classification, transparent reasoning and interactive access. Overall, the project represents a framework of how the output of deep learning can be made more understandable and trustworthy, without changing the structure of the underlying model.

V. FUTURE WORK

Future improvements can be created directly on top of the existing code structure to increase scalability, interpretability and improve user interface. The current system books one image at a time; the extension of the interface to work on batches would add more usability in the case of larger datasets. Current background generation script could be extended to have a more diverse sample, or be automated for better SHAP explanation stability with new data. Although the confidence threshold mechanism is helpful to reduce the misclassification, additional logic could be added to ensure that uncertain cases get logged for later review or retraining. The Streamlit interface could also add this option for the user to toggle explanations of the different types, as the framework anyway supports modular visualization (with Matplotlib). Performance optimizations could be made to the system by caching more of its components, or lowering the cost of SHAP computation while maintaining the quality of the attributions. A breakthrough would be the integration of measurement methods for lightweight and easily built-in evaluation of the model performance directly in the interface, so that the users can understand model performance without external tools. These improvements can be applied without modifying the main part of VGG16 model, which helps to maintain continuity and the possibility of updating the system with improvements.

REFERENCES

[1]. Alzanin, S. (2025). Explainable artificial intelligence with temporal convolutional networks for adverse weather condition detection in driverless vehicles. *Scientific Reports*, *15*(1), 19475.

[2]. Ahmed, A., Siam, A. I., Atwa, A. E. M., Atwa, M.
, Abdelrahim, E. M., & Atlam, E. S. (2025). An Explainable YOLO-Based Deep Learning Framework for Pneumonia Detection from Chest X-Ray Images. *Algorithms*, *18*(11), 703.

[3]. Dolatyabi, P., Regan, J., & Khodayar, M. (2025). Deep Learning for Traffic Scene Understanding: A Review. *IEEE Access*.

[4]. Khan, H. (2025). Advancements in Crop Analysis through Deep Learning and Explainable AI. *arXiv preprint arXiv:2508.19307*.

[5]. Mustafa, M. A., Erdem, O. A., & Söğüt, E. (2025). Hybrid Optimization and Explainable Deep Learning for Breast Cancer Detection. *Applied Sciences*, *15*(15), 8448.

[6]. Schwalbe, G., Mikriukov, G., Heinert, E., Gerolymatos, S., Keser, M., Knoll, A., ... & Mütze,
(2025, July). On Background Bias of Post-Hoc Concept Embeddings in Computer Vision DNNs. In *World Conference on Explainable Artificial Intelligence* (pp. 45-71). Cham: Springer Nature Switzerland.

[7]. Fernando, N., Seneviratne, L., Weerasinghe, N., Rathnayake, N., & Hoshino, Y. (2025). Efficient Hotspot Detection in Solar Panels via Computer Vision and Machine Learning. *Information*, *16*(7), 608.

[8]. Chen, T., Zhang, L., Guo, W., Zhang, Z., & Datcu,

[9]. M. (2025). Analyzing the Adversarial Robustness and Interpretability of Deep SAR Classification Models: A Comprehensive Examination of Their Reliability. *Remote Sensing*, *17*(11), 1943.

[10]. Chen, T., Zhang, L., Guo, W., Zhang, Z., & Datcu,

[11]. M. (2025). Analyzing the Adversarial Robustness and Interpretability of Deep SAR Classification Models: A Comprehensive Examination of Their Reliability. *Remote Sensing*, *17*(11), 1943.

[12]. Zhao, Y., & Xie, Q. (2025). Review of Deep Learning Applications for Detecting Special Components in Agricultural Products. *Computers*, *14*(8), 309.