RESEARCH ARTICLE OPEN ACCESS

Model Testing and Validation of EfficientVisionTransformer (EVT), a Deep Learning Model to Assure Quality and Accuracy

Emmanuel Chinyere Echeonwu*, Moses Okechukwu Onyesolu**, Bolou Dickson Bolou***

*(Department of Computer Science, Nnamdi Azikiwe University, Awka, Nigeria
Email: ec.echeonwu@stu.unizik.edu.ng)
** (Department of Computer Science, Nnamdi Azikiwe University, Awka, Nigeria
Email: mo.onyesolu@unizik.edu.ng)
***(Department of Computer Science, Nigeria Maritime University, Okerenkoko, Delta State, Nigeria
Email: bolou.boluo@nmu.edu.ng)

----------**********----------

Abstract:

This paper presents a rigorous testing and validation methodology for the EfficientVisionTransformer(EVT), a state-of-the-art deep learning model designed to predict vegetation indices from remotely sensed images in Southern Nigeria. Given the critical role of vegetation monitoring in ecological preservation and agricultural productivity, ensuring the accuracy and reliability of such models is paramount. The testing methodology uses a multi-pronged approach, including unit testing, integration testing, boundary value testing, and overall accuracy evaluation. Unit testing tested the correct operation of individual EVT components such as patch embedding, encoder, and decoder modules. Integration testing examined the flawless interoperability of various components, identifying any flaws in their interactions. Boundary value testing pushed the limits by evaluating the model's performance under extreme input situations, examining robustness and capacity to deal with edge cases. Importantly, accuracy testing using real-world data from Southern Nigeria demonstrated the EVT's prediction capabilities. The model has an amazing root mean square error of 0.04582. Furthermore, the normalized root mean square errors were 0.04582 when normalized to the 0-1 range and 0.00864 when normalized to the target variable's 1.8-7.1 range. These measurements show that the model is very accurate, with predictions departing from real vegetation index values by <1% of the whole data range on average. The extensive testing methodology established the EfficientVisionTransformeras a reliable and high-performing solution for predicting vegetation index in Southern Nigeria. The findings of this work provide major contributions to the advancement of remote sensing and deep learning research, as well as enabling more accurate vegetation monitoring, which is vital for ecological conservation, sustainable agriculture, and regional economic growth.

Keywords **—Software Testing, Validation, Remote Sensing, Deep Learning, Root Mean Square Error, Normalized Root Mean Square Error, Unit Testing**

----------**********----------

I. INTRODUCTION

Developing robust and reliable deep learning models requires rigorous quality assurance measures. This paper delves into the model testing and validation strategies employed to ensure the accuracy and efficacy of the deep learning model for vegetation index (VI) prediction in Southern Nigeria. Vegetation indices (VIs) are numerical indicators derived from remote sensing data that assess the abundance and condition of vegetation on the ground surface. VIs are commonly used to

monitor vegetation dynamics, evaluate crop health and yield, and investigate the effects of climate change on vegetation [14].

The normalized difference vegetation index (NDVI) is a simple and widely used indicator based on near-infrared (NIR) and red (RED) light reflectance measurements. It is computed as NDVI = (NIR−RED)/(NIR+RED) [2]. However, NDVI and other classic VIs have drawbacks, including sensitivity to meteorological conditions, soil background, and saturation effects. As a result, more advanced algorithms are required to extract useful information from remote sensing data while also improving the accuracy and reliability of VI prediction. One interesting approach is to employ deep learning models such as convolutional neural networks (CNNs) and vision transformers (ViTs), which can learn complicated and high-level features from data and achieve cutting-edge performance in a variety of computer vision applications [12].

Vision transformers (ViTs) are deep learning models that process picture data with the transformer architecture, which was initially created for natural language processing. ViTs partition the input image into patches and use self-attention processes to detect global and long-range relationships between the patches. ViTs have outperformed CNNs in a variety of computer vision applications, including image classification, object recognition, and semantic segmentation [8]. However, ViTs have certain downsides, including the need for a huge quantity of data and computing resources, as well as memory inefficiency due to tensor reshaping and element-wise functions in the multi-head self-attention (MHSA) module. As a result, various modifications and upgrades to ViTs have been proposed to overcome these concerns and increase their efficiency. One of them is the EfficientVisionTransformer (EVT), a family of high-speed ViTs that use a sandwich layout, i.e., a single memory-bound MHSA between efficient feed-forward network (FFN) layers, and a cascaded group attention module that feeds attention heads with different splits of the full feature, which not only reduces computation cost but also improves attention diversity [13].

Testing deep learning models using typical software testing approaches confronts two major challenges. The first is the Oracle issue. This relates to the challenge of determining exactly what output a model should generate for a given input. Unlike traditional software, which has a predetermined conclusion, deep learning models frequently deal with complicated input and nuanced outputs, making it difficult to discern the actual output. The second obstacle is test adequacy. This relates to the problem of selecting the appropriate test inputs to accurately assess the model's performance. Simply tossing random data at the model is insufficient. We require well picked inputs that actually evaluate the model's capacity to do its intended goal [1].

Broadly speaking, the following techniques have been proposed for evaluating deep learning models [1] :

a.) Differential Testing : Differential Testing (DT) evaluates a system's functionality by comparing the behavior of multiple implementations or models for the same task. The difference between their results is used to assess the correctness of the tested program. The main challenges of DT are identifying the faulty system and utilizing a test set.

b.) Metamorphic Testing: Metamorphic Testing (MT) is a useful approach for programs that lack a clear oracle. Metamorphic relations (MR) between a program's inputs and outputs help identify program flaws in MT.

c.) Mutation Testing: Mutation Testing (MuT) is a conventional testing method that evaluates a test set and generates fault-revealing test cases. MuT evaluates a test suite based on its ability to discriminate between the original program and its mutants, which are modified versions of the program being tested.

d.) Combinatorial Testing: Combinatorial Testing (CT) evaluates software systems and test suites by analyzing value relationships and coverage of input data. CT assumes that input parameters interact and that certain values may result in problems.

e.) Adversarial Perturbation Testing: Adversarial Perturbation Testing (APT), a technique for attacking machine learning systems, can also be used to evaluate deep learning models. An adversarial example is a purposeful attack on a deep learning model, causing the system to provide incorrect results.

These methods are time-proven and useful, however, the quality and resilience of deep learning (DL) models are determined by a variety of criteria, including the type and quality of the data, the model's complexity and interpretability, the system's intended purpose and domain, and the potential risks and consequences of failure. Different applications may have distinct requirements and expectations for these aspects, making DL system testing a subjective procedure [10]. DL system testing is likewise a difficult procedure since there are several sources of uncertainty and unpredictability that might affect model behavior and outcomes. These include the unpredictability and variability of the training and testing data, the non-linearity and opacity of the model structure and parameters, the existence of adversarial and noisy inputs, and the system's dynamic and developing character [8].

In this paper, we propose using EVT as an innovative and effective method for predicting VI in Southern Nigeria, where vegetation plays a vital role in both the ecology and the economy. We look into model testing and validation methodologies for EVT to assure model quality and dependability. We anticipate that our paper will help develop remote sensing and deep learning research while also providing important insights for vegetation monitoring and management in Southern Nigeria and other places.

II. BACKGROUND

Vegetation index (VI) estimate is an important problem in remote sensing and environmental monitoring, and deep learning algorithms have shown tremendous promise in this area. However, deep learning algorithms for VI estimate present considerable hurdles due to data complexity, environmental unpredictability, and domain uniqueness, necessitating thorough testing and validation. Researchers can overcome these problems by applying a variety of innovative methodologies, ensuring the quality and usefulness of models for real-world applications.

Data complexity refers to the high dimensionality, heterogeneity, and noise of remote sensing data, all of which can have an impact on deep learning model performance and generalization. For example, satellite images might have varying resolutions, cloud cover, atmospheric distortion, and lighting conditions. Data augmentation, multi-scale and multi-modal learning, and attention processes are some strategies for dealing with data complexity [16].

Environmental variability refers to the dynamic and diverse nature of the earth's surface, which can alter plant features and the link between VI and spectral reflectance. Seasonal fluctuations, land cover transitions, and human activities can all have an impact on vegetation structure, phenology, and health. Some strategies used to deal with environmental unpredictability include temporal modeling, domain adaptation, and transfer learning [5].

Domain specificity relates to the sensitivity of VI estimate methods to certain plant kinds, geographies, and applications. For example, various plant kinds may have distinct spectral signatures, different places may have different environmental conditions, and different applications may have

different VI needs and goals. Some strategies used to address domain specificity include fine-tuning, meta-learning, and explainability [9].

Testing and validation are critical processes in ensuring the accuracy and generalizability of deep learning models for VI estimation. However, due to the aforementioned issues, testing and validation can be complex and time-consuming. To assess the models' functionality, accuracy, and robustness in different scenarios and domains, various testing and validation strategies are used, including unit testing, integration testing, system testing, cross-validation, performance metrics, and qualitative analysis [15].

Testing deep learning models is an important step in ensuring their dependability and resilience in real-world scenarios. However, assessing these models presents distinct issues due to their complexity, nondeterminism, and data reliance. As a result, multiple testing procedures have been created and modified to solve these issues and evaluate various elements of deep-learning models. In this section, we will summarize and compare three commonly used testing strategies: unit testing, integration testing, and system testing.

Unit testing is the process of examining individual model components, such as layers or modules, in isolation to ensure that they perform as intended. The PyTorch unit test framework automates this process by providing tools for creating and executing test cases, verifying assertions, and reporting findings [1]. Unit testing can assist uncover problems and errors in model implementation while also validating the model design and parameters. However, unit testing alone cannot guarantee the model's overall performance and behavior since it does not take into account the interactions and dependencies between model components, as well as the input and output data [10].

Integration Testing evaluates how many components interact and collaborate to reach the model's ultimate purpose. Depending on the complexity and structure of the model, integration testing can be performed at several levels of granularity, including layer-level, module-level, and model-level [3]. Integration testing can help uncover errors and inconsistencies in model integration while also evaluating the model's usefulness and efficiency. However, creating and implementing integration testing can be difficult since it necessitates specifying appropriate test inputs, outputs, and criteria for each level of integration [5].

System testing is an approach that uses real-world data and situations to evaluate the overall performance of the entire model in the intended task. System testing can be done in a variety of ways, depending on the evaluation objectives and criteria [11]. System testing can help evaluate the model's accuracy, robustness, and generalizability, as well as detect any flaws and weaknesses in its behavior. However, system testing can be costly and time-consuming since it necessitates big and diversified datasets, realistic and representative test cases, and rigorous and thorough analysis [10]

III. METHODOLOGY

The technique used for model testing and validation of the EfficientVisionTransformer takes a thorough approach to assure model quality and accuracy. The testing procedures uses a variety of methodologies, including boundary value testing, unit testing, integration testing, and accuracy evaluation. This section outlines the comprehensive testing methodology employed.

a.) Unit testing is a software engineering technique adopted for deep learning models. It entails isolating and testing each component or module of the model individually. This strategy aids in the early detection and resolution of difficulties, ensuring that each component performs properly prior to integration. Unit testing also enhances code reuse and maintainability, which are critical for complicated deep learning models.

b.) Boundary value testing is an important step in determining the model's resilience and behavior at the limits of its input domain. This method includes feeding the model edge cases, outliers, and inputs that push the limits of its predicted operation. Analyzing the model's performance under these settings allows us to find possible weaknesses, edge case behaviors, and opportunities for improvement.

c.) Integration testing is the practice of merging different components or modules to assess their overall functioning. Integration testing in deep learning models guarantees that all of the components, such as feature extraction, attention mechanisms, and classification layers, function together flawlessly. This testing technique aids in the identification and resolution of integration issues, ensuring that the model's overall performance matches the required criteria.

d.) Accuracy testing is a critical component of assessing the EfficientVisionTransformer's performance. This technique entails calculating the model's predicted accuracy using a wide set of test data that is typical of real-world circumstances. Mean square error, mean absolute error, the root mean square error and Normized root mean square error [4], were accuracy measures that are generated and examined to evaluate the model's efficacy and indicate areas for improvement. Furthermore, the model's accuracy is compared to other cutting-edge models or human performance, offering a comparative overview of its capabilities.

This comprehensive testing technique, which includes boundary value testing, regression testing, interpretability testing, unit testing, integration testing, and accuracy testing, provides a strong foundation for ensuring the quality and correctness of the EfficientVisionTransformer model. By painstakingly implementing these various testing methodologies, we can verify the EVT's dependability, generalizability, and trustworthiness in real-world scenarios.

IV. RESULTS AND DISCUSSION

TABLE I
UNIT AND INTEGRATION

Test Case	Patch Embedding Output Shape	Encoder Output Shape	Decoder Output Shape
Functionality Tested	Output dimensions after patch embedding	Dimensions of Encoder output	Dimensions of decoder output
Sample Data Shape	(4, 3, 600, 600)	(4,3,600,600)	(4, 3, 600,600)
Actual Result	(4, 600, 37, 37)	(4,1369,600)	(4, 1)
Expected Result	(4, 600, 37, 37)	(4,1369,600)	(4, 1)
Pass/Fail	Pass	Pass	Pass

Table I shows the results of test scenarios meant to validate the EfficientVisionTransformer's (EVT) output dimensions at various processing stages. It compares the predicted output forms to those achieved throughout the testing procedure.

Test Case: This column indicates the functionality under test, which in this case is the dimensional output of each stage of the EVT model.

Functionality Tested: This column contains a more complete description of what each test case verifies. It explains that the tests are testing the dimensions of the model's output at three important points: after the patch embedding layer, after the encoder, and after the decoder.

Sample Data Shape: This column displays the shape of the input data used in the test scenarios. In this case, the sample data takes the shape of (4, 3, 600, 600), indicating a set of four images, each having three channels (RGB) and a height and width of 600 pixels.

Actual Result versus Expected Result: These columns compare the dimensions generated by the EVT during testing (Actual Result) to the expected dimensions based on the model's design.

Pass/Fail: This last column shows the results of each test scenario. In this case, all three tests succeeded, indicating that the EVT's output dimensions are consistent with expectations.

The patch embedding layer converts the input image into a series of patches. The intended and actual output shapes of (4, 600, 37, 37) show that the layer successfully turned the batch of four images into a sequence of 600 patches, each represented by a vector of 37 features. The encoder processes the previous stage's patch sequence. The intended and actual output shape of (4, 1369, 600) indicate that the encoder converted the series of 600 patches for every image in the batch into a new sequence of 1369 elements, each with a dimension of 600. The decoder generates the final prediction based on the encoder's output.

TABLE II
BOUNDARY VALUE

Test Case	**Case 1**	**Case 2**	**Case 3**
Functionality	Test Minimum value (1.8)	Test Maximum Value (7.1)	Test Minimum and Maximum (1.8 & 7.1)
Result	-0.138	-0.138	-0.139

This test is designed to evaluate the model's behavior at the limits of its predicted input range. By putting boundary images into the model, we may uncover possible concerns such as unexpected outputs or mistakes that may occur when the model faces extreme input values in real-world scenarios. The findings show that the model makes consistent predictions in circumstances when all pixels have the same minimum or maximum value. However, the varied forecast in Case 3 indicates that the model's behavior may be more complex when dealing with mixed input values.

TABLE III
MODEL ACCURACY

Functionality	RMSE	NRMSE 1 (0-1)	NRMSE 2 (I.8 – 7.1)
Result	0.04582	0.04582	0.00864

Table III shows the results of the Normalized Root Mean Square Errors (NRMSE) for different test values. The EfficientVisionTransformer achieved a Root mean Square Error (RMSE) of 0.04582. When normalized with data range (0-1), the RMSE equals the NRMSE. NRMSE 2 shows the result when normalized with range (1.8-7.1). The Normalized Root Mean Square Error (NRMSE) is calculated by:

$$NRMSE = \frac{RMSE}{\text{Range Max} - \text{Range Min}} \quad \ldots. \quad (1)$$

Where RMSE = Root Mean Square Error
Range Max = Maximum value of the observed data
Range Min = Minmum value of the observaed data

NRMSE 1 is normalized for the target range (0-1). The number of 0.04582 indicates that the model's average error is approximately 4.5% of the target variable's range. In other words, on average, the model's predictions depart from real values by 4.5% of the variable's potential range. NRMSE 2 shows a better perforance showing a 0.8% deviation on the target variable.

Normalized Root Mean Square Error (NRMSE) is an important measure in remote sensing. It is used to evaluate the accuracy of models that forecast the geographical distributions of various parameters acquired from remote sensing data, such as vegetation indices, soil moisture, and land surface temperatures.

V. CONCLUSIONS

This paper investigates the application of simplistic model testing and validation procedures as it applies to deep learning models. It explores the challenges associated with testing and validation of deep learning models for VI estimation,

emphasizing the importance of addressing data complexity, environmental variability, and domain specificity. To ensure the robustness and generalizability of the EfficientVisionTransformer (EVT) model, the study implementes a comprehensive testing methodology incorporating unit testing, integration testing, boundary value testing, regression testing, interpretability analysis, and accuracy evaluation. The results demonstrated that the EVT produced consistent and accurate VI predictions, with a Normalized Root Mean Square Error (NRMSE) of approximately 0.8% when normalized for the target range (1.8 - 7.1). These findings suggest that EVTs hold promise for VI estimation in Southern Nigeria and potentially other regions. Future research areas include investigating the explainability of the EVT model to acquire a better understanding of its decision-making processes. Furthermore, research into transfer learning techniques may allow the EVT model to be adapted to different geographical areas and plant types, increasing its generalizability for larger real-world applications.

ACKNOWLEDGMENT

I am grateful to Google for supporting this project through the EMEA PhD Research Fellowship. This expedition would have been significantly more difficult, if not impossible, without your generosity. I also want to thank my superivor, Prof M. O Onyesolu for his support and encouragement.

REFERENCES

[1] Ahuja, M. K., Gotlieb, A., & Spieker, H. (2022). Testing deep learning models: A first comparative study of multiple testing techniques. arXiv preprint arXiv:2202.12139.

[2] Borowik, T., Pettorelli, N., Sönnichsen, L. & Jedrzejewska, B.(2013). Normalized difference vegetation index (NDVI) as a predictor of forage availability for ungulates in forest and field habitats. *Eur J Wildl Res*59:675– 682. https://doi.org/10.1007/s10344-013-0720-0.

[3] Chorev, S. (2022). How to test machine learning models. Deepchecks.https://deepchecks.com/how-to-test-machine-learning-models/. [Accessed on 13 February 2014].

[4] Dimitriadou, S., and Konstantinos G. N.. (2022). Development of the Statistical Errors Raster Toolbox with Six Automated Models for Raster Analysis in GIS Environments. *Remote Sensing* 14, no. 21: 5446. https://doi.org/10.3390/rs14215446.

[5] Ghamisi, P., Souza, R., Benediktsson, J. A., Zhu, X. X., Rasti, B., & Werth, H. (2017). Multilevel image segmentation based on fractional-order Darwinian particle swarm optimization. IEEE Transactions on Geoscience and Remote Sensing, 55(5), 2664-26741.

[6] Gill, J. K. (2022). Machine learning model testing training and tools. XenonStack. https://www.xenonstack.com/blog/cognitive-automation-machine-learning-model-testing-training-tools/.[Accessed on 13 February 2024].

[7] Khan, R. (2021). Deep learning system and its automatic testing: An approach. Annals of Data Science, 10 (2):1019–1033.

[8] Liu, X., Peng,H., Zheng,N., Yang,Y., Hu, H., Yuan, Y. (2023). EfficientViT: Memory Efficient Vision Transformer with Cascaded Attention. https://arxiv.org/abs/2305.07027.

[9] Liu, Y., Wang, Z., Li, H., & Zhang, L. (2020). Deep learning for pixel-level image fusion: Recent advances and future prospects. Information Fusion, 64 :71-872.

[10] Ma, W., Papadakis, M., Tsakmalis, A., Cordy, M., & Le Traon, Y. (2020). Test Selection for Deep Learning Systems2. ACM Transactions on Software Engineering and Methodology, 30(2), 1-22. https://doi.org/10.1145/3417330.

[11] Mandal, S. (2022). Debugging deep learning models. https://mlearning.ai. [Accessed on 13 February 2024].

[12] Nguyen, K. A., Seeboonruang, U., and Chen, W. (2023). Projected Climate Change Effects on Global Vegetation Growth: A Machine Learning Approach. *Environments* 10(12): 204. https://doi.org/10.3390/environments10120204.

[13] Patro, B. & Agneeswaran, V.S. (2023). Efficiency 360: Efficient Vision Transformers. https://arxiv.org/abs/2302.08374.

[14] SpaceSence, (2021). Vegetation Indices: What, how and When?.https://www.spacesense.ai/blog-posts/vegetation-indices-what-how-and-when. [Accessed 10 February 2024].

[15] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-19583.

[16] Zhang, C., Sargent, I., Pan, X., Li, H., Gardiner, A., Hare, J., & Atkinson, P. M. (2020). Joint deep learning for land cover and land use classification. Remote Sensing of Environment, 221, 173-187.

