On Adapting a Decision Tree Coverage Criterion for Testing Image Classification Models

Lúcio C. A. Pinto
Universidade de São Paulo
São Carlos, SP, Brazil
lucio.pinto@usp.br

Vinicius H. S. Durelli
Universidade Federal de São Carlos
São Carlos, SP, Brazil
vinicius.durelli@ufscar.br

Marcio E. Delamaro
Universidade de São Paulo
São Carlos, SP, Brazil
delamaro@icmc.usp.br

Simone R. S. Souza
Universidade de São Paulo
São Carlos, SP, Brazil
srocio@icmc.usp.br

Abstract

While Machine Learning models are increasingly deployed for complex tasks, verifying their reliability remains a significant challenge, as traditional software testing techniques cannot be directly applied to data-driven systems. Additionally, conventional evaluation strategies (e.g., random sampling and 10-fold cross-validation) do not explicitly exploit structural information about model behavior, potentially limiting their ability to expose challenging or failure-inducing inputs. To circumvent this limitation, specialized testing criteria have emerged. In this context, this paper investigates the applicability of a Decision Tree Coverage (DTC) criterion for guiding test case selection in image classification tasks. Although DTC has previously demonstrated effectiveness in selecting test cases for symbolic classification models, its use in the context of image classification, where models operate over high-dimensional, unstructured data, remains unexplored. To bridge this gap, we propose an adaptation of DTC that leverages decision trees constructed from image embeddings to derive structurally informed test cases for evaluating image classification models. An experiment was conducted using a YOLO-based classification model trained on the Cassava and DeepWeeds datasets. The effectiveness of the proposed approach is assessed in terms of the extent to which it leads to reduced predictive performance relative to test sets obtained via 10-fold cross-validation. Results from our experiment show that the DTC criterion significantly outperforms random selection by identifying more challenging test inputs, thereby indicating its ability to expose more failure-prone cases and reveal model deficiencies more effectively. Moreover, DTC achieves these results while requiring fewer test cases and maintaining comparable execution time. These findings indicate that leveraging internal model structures for test case selection can yield more rigorous evaluations of image classification models.

Keywords

Software Testing, Machine Learning, Image Classification, Decision Tree

1 Introduction

Software development is a complex task, influenced by several factors throughout the development process. Software Engineering (SE) acts as a set of guidelines and best practices to guide this process and ensure that the final product is as aligned as possible with expectations. As with any activity involving human execution, failures may occur at different stages, requiring specific strategies for their mitigation [2]. With the growing popularization of Machine Learning (ML) and its data-driven paradigm, new challenges have emerged in the field of SE, especially in the area of Software Testing (ST).

Traditionally, computational problems are solved through the implementation of algorithms with well-defined rules to address specific issues [9]. This approach proves to be highly effective in deterministic contexts with a clear systematic structure. However, as the complexity of problems increases, so does the need for specialists to render these problems into computational solutions. To reduce this dependency and minimize human intervention, it became necessary to develop more autonomous computational tools. ML is based on the ability to learn tasks from historical data [7]. In other words, when exposed to a specific set of information, an ML-based system is expected to be capable of interpreting new data based on previously acquired knowledge.

Recent studies have shown that testing criteria based on Decision Trees (DT) are effective in selecting test cases for classification algorithms in ML systems [17, 18]. However, such approaches have not yet been adapted to the context of image classification, representing a gap in the literature.

Considering the emerging challenges in the context of verifying ML-based systems, this paper proposes the use of a Decision Tree Coverage (DTC) criterion for selecting test cases in image classification tasks. The main objective is to improve the selection of test cases in order to enable the identification of faults that may arise during the development of image classification models.

The primary contribution of this study is to provide the first systematic investigation of how the DTC criterion can guide the selection of more effective test cases in the context of image classification, a fundamental task in the field of computer vision. In contrast to conventional evaluation procedures (e.g., random partitioning strategies based on 10-fold cross-validation), which do not explicitly exploit model structure, the DTC criterion leverages structural information to derive test cases that are more likely to expose deficiencies in model behavior. The effectiveness of the proposed approach is assessed by examining the extent to which test cases derived from the DTC criterion lead to reductions in prediction scores when compared to randomly selected test cases obtained

through 10-fold cross-validation. By providing empirical evidence that structurally guided test case selection can yield more rigorous evaluations, this study contributes to advancing methodologies for testing and assessing the robustness of image classification models.

The remainder of this paper is organized as follows. Section 2 presents related work. Section 3 describes the experimental design adopted to evaluate the DTC criterion in the context of image classification models. Section 4 presents the experimental results. Section 5 provides a statistical analysis of these results. Section 6 discusses threats to validity. Finally, Section 7 summarizes the main findings and outlines directions for future research.

2 Related Work

Several software testing activities can be formulated as learning problems and analyzed using ML techniques. This has motivated researchers to apply ML to address software testing challenges, increasing the number of studies in the area. In the work conducted by Riccio et al. [16], a Systematic Mapping (SM) was carried out with the goal of identifying and classifying existing solutions focused on functional testing of ML-based systems. The analysis was based on the careful selection of 70 primary studies and the formulation of 33 research questions, distributed across three main dimensions: (i) the context of the addressed problems; (ii) the functionalities tested; and (iii) the empirical evaluation methods employed.

As a conceptual contribution, Riccio et al. [16] propose a third category of testing approach, in addition to the well-known white-box and black-box, namely the data-box. In ML systems, black-box techniques evaluate the system without considering its internal structure, while white-box approaches assume full access to the model architecture. The data-box, in turn, is positioned between these two, using input and output data as well as the trained model, but without requiring knowledge of its internal structure, making it a particularly interesting alternative for systems such as Deep Neural Networks (DNNs). Regarding test adequacy criteria, the study identified 12 distinct proposals primarily focused on evaluating test coverage in ML models, with an emphasis on the generation of new test cases. Most of these criteria focus on architectures based on Artificial Neural Networks (ANNs), with notable works including [4, 8, 14, 19].

The study conducted by Zhang et al. [23] presented a comprehensive survey focused on the area of ST in ML-based systems. The investigation includes a detailed analysis of 144 published studies, from which relevant information was extracted regarding aspects such as system robustness, components used in testing-including datasets, learning algorithms, and frameworks, as well as test case generation and the different application domains explored. An important distinguishing feature of this research is the emphasis placed on ethical and fairness issues in the analyzed models. The authors acknowledge that shortcomings in the representativeness and organization of training data may result in biased models, capable of producing discriminatory or unfair decisions. This highlights the need to consider not only the technical accuracy of models but also their social impact.

With regard to test adequacy criteria, especially those related to coverage, Zhang et al. [23] argue that traditional code coverage metrics, widely used in conventional systems, are ineffective for ML-based applications. This is because, in such systems, the decision logic is not explicitly implemented in the code, but rather embedded in patterns learned from data. In this context, the authors refer to studies that propose new approaches to address this challenge, highlighting the contributions of [8, 14, 19], which explore coverage criteria based on neuron activation in DNNs.

2.1 Coverage Criteria Applied to Artificial Neural Networks

In traditional software development, code coverage criteria are widely used as a metric to evaluate the quality and comprehensiveness of the tests applied to a system [2]. These techniques, known as white-box, assume that the evaluator has access to the source code and the internal structure of the application. However, this logic does not directly apply to ML-based systems, whose functionality is not explicitly coded, but learned from data during the training process [7]. Given this paradigm shift, new coverage criteria have emerged, specifically designed to address the particular characteristics of these data-driven systems.

The first relevant initiative in this direction was presented by Pei et al. [14], with the framework *DeepXplore*, which introduced the concept of neuron coverage as an analogy to code coverage. This metric evaluates which regions of a deep neural network are effectively activated during test execution. To address the well-known oracle problem, that is, the absence of a clear reference to judge the correctness of the output, the authors employed differential testing, comparing the behavior of different models for the same input. Additionally, they demonstrated that the generation of test inputs that maximize neuron coverage can be formulated as a joint optimization problem, solvable using gradient-based methods.

Inspired by this work, Ma et al. [8] proposed *DeepGauge*, a framework that extends the notion of coverage in DNNs. *DeepGauge* introduces five new criteria, organized into two levels of analysis: (i) *neuron level* and (ii) *layer level*. Three criteria evaluate individual neurons, while the other two consider the collective behavior of layers. By using multiple activation thresholds and expanding the scope of evaluation beyond binary activations, *DeepGauge* provides a more detailed analysis of the coverage achieved by the tests.

Building on these proposals, several works have continued advancing the definition of coverage metrics for ANNs, such as those by [5, 6, 12]. More recently, Yuan et al. [22] proposed the Neural Layer Coverage (NLC) criterion, which considers the layer, rather than the individual neuron, as the minimum unit of analysis. NLC incorporates variables such as continuity, correlation, distribution, and density of activations, providing a more robust evaluation of coverage. In addition, the authors established eight requirements that, according to them, coverage criteria should satisfy, all of which are met by NLC. In conducted experiments, this new criterion outperformed previous approaches both in performance and in coverage.

Complementing this line of research, Xie et al. [21] proposed two criteria focused on the analysis of paths and decision structures within DNNs. The proposal differs by prioritizing interpretability, a feature that is uncommon but highly desirable in opaque models such as DNNs. Inspired by classical structural testing techniques,

the authors modeled decision graphs and defined critical paths to evaluate coverage. The interpretation technique employed was Layer-wise Relevance Propagation (LRP) [1], which makes it possible to identify the most relevant neurons in each layer during image classification.

2.2 Decision Tree-Based Testing Criteria

Recent research has explored the use of DT models as a basis for defining testing criteria and selecting test cases, showing promising results [17, 18]. In the study conducted by Santos et al. [17], two testing criteria were proposed for classification algorithms based on DT. Both criteria start from the construction of a tree using the dataset related to the problem under analysis. Once the tree is generated, the criteria are applied to guide the selection of test cases.

The first criterion, Decision Tree Coverage (DTC), is defined as follows: "Given a decision tree model M, a test set T is considered DTC-adequate if it traverses all paths from the root to each leaf at least once" [17]. This criterion is inspired by the principles of control flow graph coverage, widely used in white-box testing techniques applied to conventional systems.

The second criterion, known as Boundary Value Analysis (BVA), consists of: "Designing test cases that cover the boundary values of decision nodes. This criterion requires the selection of values that explore the lower and upper limits of each decision" [17]. It is an adaptation of the homonymous criterion commonly applied in functional testing, aiming to exercise sensitive regions of the decision logic.

To satisfy DTC, the test set must contain at least one test case that traverses each path of the tree, from the root to the leaves, which implies, at minimum, one test per leaf. To satisfy BVA, it is necessary to consider the decision boundaries at the internal nodes, which often requires the creation of examples that go beyond the original data. These boundary values are generally inferred based on percentage variations relative to the split points observed in the tree. As an illustration, consider the Iris dataset [3] and the decision tree generated from it in Figure 1.

Table 1: Example of a DTC test case that satisfies the test requirement of covering leaf node 16

sepal length	sepal width	petal length	petal width	class
6.3	3.3	6.0	2.5	2

Table 1 presents an example of a DTC test case that satisfies the requirement of covering node 16, classified as class 2. Table 2 presents an example of a BVA test case for covering the same node, generated by applying a 10% increase to the corresponding thresholds.

Table 2: Example of an BVA test case that satisfies the test requirement of covering leaf node 16

sepal length	sepal width	petal length	petal width	class
5.0	3.0	5.335	1.925	2

Continuing this line of research, Silveira et al. [18] proposed the Decision Tree Mutation Testing (DTMT) criterion, defined as follows: "Given a decision tree, a test set is considered adequate for DTMT if it produces divergent classifications across all generated mutant trees." This criterion is based on the principles of fault-based testing, widely used to reveal faults in traditional systems.

In DTMT, each mutant of the original tree represents a test requirement. Mutations are implemented by replacing relational operators (for example, changing "$x \leq 5$" to "$x \geq 5$") or by modifying constants in decision nodes (such as changing $X[0] = 1$ to $X[0] = 2$). Considering the tree from the Iris dataset (Figure 1), the application of four different operators to 8 decision nodes results in 32 mutant trees. An example of one of these modified trees can be seen in Figure 2.

Note that the modification applied to the decision tree shown in Figure 2 consisted of replacing the relational operator $\leq$ with $\geq$ at node 2, that is, the condition $X[3] \leq 1.75$ was changed to $X[3] \geq 1.75$. In Figure 3, the constant 0.8 was replaced by 1.65 at node 0, characterizing a mutation of the type constant replacement with a valid alternative value [18].

The three criteria, DTC, BVA, and DTMT, demonstrated effectiveness in selecting test cases, outperforming traditional sampling-based approaches such as 10-fold cross-validation, especially in terms of test coverage and diversity. However, it is important to emphasize that these criteria were designed for symbolic classification models and have not yet been validated in more complex tasks, such as image classification.

This section presented an analysis of related work in the area of ST for ML-based systems, with an emphasis on testing criteria applied to ANNs and on approaches based on DT. However, no studies were found that directly address the adaptation or application of these DT-based coverage criteria to image classification tasks. Therefore, this paper applies and evaluates the DTC criterion for ML-based systems in the context of image classification.

3 Research Methodology

Currently, data production is growing exponentially, accompanied by equally rapid advances in computational capacity. This scenario has generated increasing interest, both in industry and academia, in exploring available data. Consequently, ML models have been widely adopted to solve problems across various domains. However, such models are subject to errors and biases introduced throughout multiple stages of development, which are not always easily detectable. Validation, verification, and software testing activities aim to reveal these defects before the system is deployed. Nevertheless, ML-based systems impose a new data-driven paradigm for defining system behavior, which makes the direct application of conventional testing techniques more challenging.

3.1 Experiment

The *You Only Look Once* (YOLO[1]) model represents an innovative approach to the task of object detection in images by reformulating this traditionally complex task as a single regression problem. This reformulation allows the model, given an input image, to simultaneously estimate the spatial coordinates of the bounding boxes and

[1]https://docs.ultralytics.com/pt/

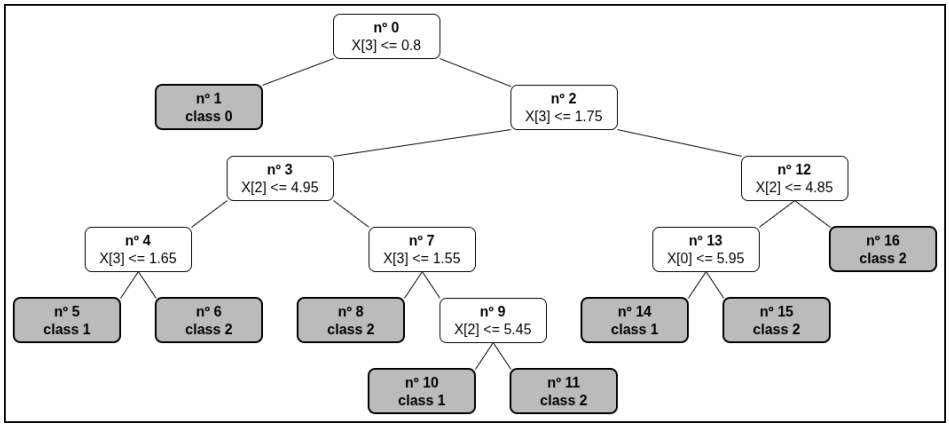

Figure 1: Decision Tree generated from the Iris dataset. The features represented in the figure correspond to: x[0] = sepal length, x[1] = sepal width, x[2] = petal length, x[3] = petal width

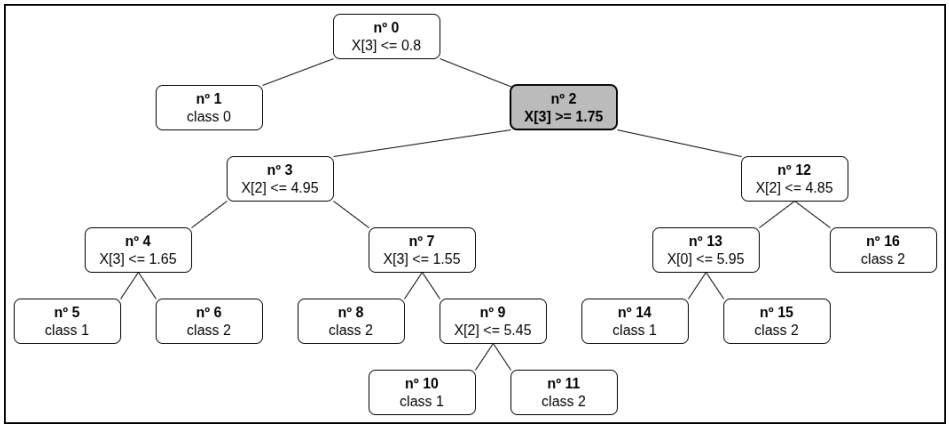

Figure 2: Decision Tree generated from the Iris dataset with a mutation example – Relational operator replacement

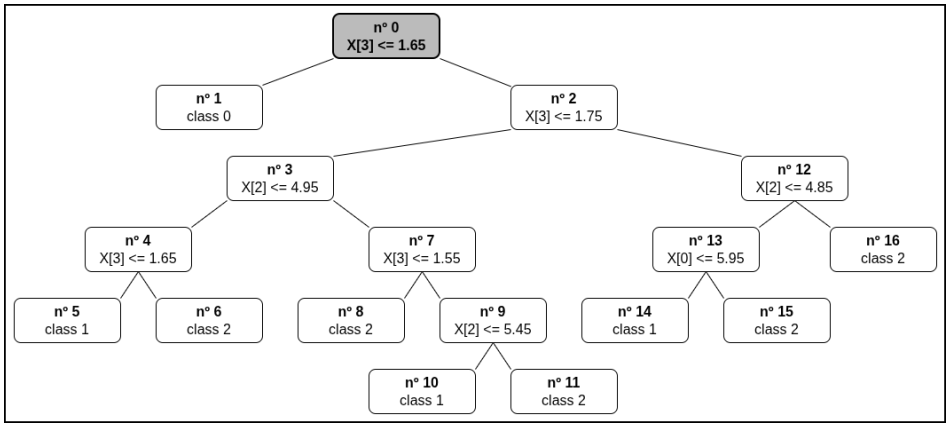

Figure 3: Decision Tree generated from the Iris dataset with a mutation example – Constant replacement

the corresponding probabilities associated with the classes of the detected objects (Figure 4). This unification of the detection process significantly contributes to reducing the computational complexity involved, providing an efficient solution suitable for real-time applications. As a result, YOLO achieves substantially higher detection rates than previous methods without significantly compromising

recognition accuracy [15]. In addition to object detection, the YOLO model also supports other computer vision applications, such as image classification.

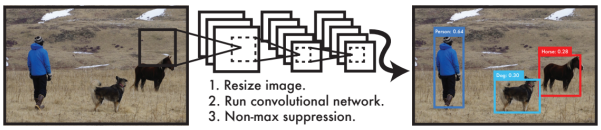

Figure 4: The YOLO detection system [15]

As mentioned, we surmise that t the DTC criterion is more suitable for selecting test cases in ML-based systems compared to random selection. This hypothesis stems from the fact that DTC considers the internal structure of the model, in which decision nodes significantly influence the model's behavior, enabling the selection of potentially more effective test cases by leveraging these structural and informational characteristics. An experiment was designed with the objective of answering the following Research Question (RQ):

- RQ_1: What is the effectiveness of test cases derived using the DTC criterion compared to random test cases obtained through 10-fold cross-validation?

3.2 Scope of the Experiment

The scope of the experiment is established based on the definition of its objectives, which were formulated according to the Goal/Question/Metric (GQM) model [20], as described below:

Analyze the DTC criterion
for the purpose of evaluation
with respect to its effectiveness
from the point of view of the researcher
in the context of testing image classification models.

3.3 Hypothesis Formulation

It was established the expectation that the DTC approach would present superior performance compared to random testing. Based on this premise, RQ_1 was reformulated into the following hypotheses:

- **Null hypothesis** (H_0): there is no difference in effectiveness between DTC-derived test cases and random test cases.
- **Alternative hypothesis** (H_1): there is a significant difference in effectiveness between DTC-derived test cases and random test cases.

The main objective of this study is to analyze whether the DTC criterion is capable of selecting more effective test cases. To this end, it proposes the derivation of test cases that, when applied to a decision tree-based model, satisfy the requirements established by this criterion. Subsequently, these cases, considered adequate according to DTC, are used to evaluate the performance of the YOLO model.

More specifically, the YOLO model (version *yolo26n-cls.pt*[2]) is implemented on the original dataset, corresponding to the training set. After training, this model is evaluated through the application of test cases considered adequate according to the DTC criterion. Subsequently, it is evaluated using the 10-fold cross-validation procedure.

In traditional software testing, the effectiveness of a criterion is associated with its ability to select test cases that reveal faults. A criterion C_1 is considered more effective than a criterion C_2 when it produces test inputs that identify more faults than those obtained from C_2. This concept also applies to our study. For a given model M, if the performance is lower when evaluated with test inputs derived from C_1 compared to those generated from C_2, then C_1 is considered more effective than C_2. Consider a metric, denoted by f, used to measure the quality of a model M, and a test set T. The score obtained by executing T on M, according to the metric f, is expressed as $f(M(T))$. Therefore, $f(M(T_1)) < f(M(T_2))$ indicates that T_1 is more effective than T_2 according to the metric f, since T_1 revealed more cases in which the model M fails than T_2 [18].

In this work, the metrics of accuracy, F_1-score, recall, and precision were used as the basis for evaluating the obtained results. The most relevant dependent variable for answering RQ_1 corresponds to **effectiveness**, which is operationalized through the ML performance metrics considered in this experiment.

3.4 Instrumentation

The Google Colaboratory (Google Colab[3]) environment was adopted for conducting the experiment, a cloud-based platform that enables the execution of Python[4] code directly in the browser. The L4 GPU runtime type was used, in which the following resources are available: 53.0 GB of System RAM; 22.5 GB of GPU RAM; and 112.6 GB of Disk. For data manipulation and analysis, the *pandas*[5] library was used, and ML algorithms were implemented using the *scikit-learn*[6] library, which is widely used for the development of ML models [10].

The results of the experiment were compared based on four performance metrics: accuracy, F_1-score, recall, and precision. For the F_1-score, recall, and precision metrics, two different aggregation methods were used: micro and macro averages. The variables used in the calculation of the performance metrics are presented below:

- **TP** (True Positives): number of True Positives;
- **TN** (True Negatives): number of True Negatives;
- **FP** (False Positives): number of False Positives;
- **FN** (False Negatives): number of False Negatives.

Next, the equations of the adopted metrics are described, along with their respective aggregation parameters:

- Accuracy

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

- F_1-score

$$F_1 = \frac{2TP}{2TP + FP + FN} \tag{2}$$

- Recall

$$R = \frac{TP}{TP + FN} \tag{3}$$

[2]https://docs.ultralytics.com/pt/tasks/classify/

[3]https://colab.research.google.com/
[4]https://www.python.org/
[5]https://pandas.pydata.org/
[6]https://scikit-learn.org/stable/

- Precision

$$P = \frac{TP}{TP + FP} \tag{4}$$

- Micro: Calculates the metrics globally by counting the total number of true positives, false negatives, and false positives.
- Macro: Calculates the metrics for each label and obtains the unweighted average. This does not take into account the imbalance between labels.

3.5 Datasets

To investigate the proposed criterion, a sample composed of two datasets available in TensorFlow Datasets[7] was selected: *Cassava*[8] [11] and *DeepWeeds*[9] [13]. Table 3 presents an overview of the datasets.

The *Cassava* dataset consists of images of cassava plant leaves, including healthy leaves and four disease conditions: *Cassava Mosaic Disease* (CMD), *Cassava Bacterial Blight* (CBB), *Cassava Green Mite* (CGM), and *Cassava Brown Streak Disease* (CBSD). The dataset comprises a total of 9,430 labeled images, organized into three subsets: training (5,656 images), testing (1,885 images), and validation (1,889 images). The distribution of samples across classes is imbalanced, as two diseases, CMD and CBSD, account for 72% of the total images [11]. In this experiment, the training subset, consisting of 5,656 images, was used.

The *DeepWeeds* dataset consists of 17,509 images capturing eight distinct species of weeds native to Australia, collected in their natural environment along with the surrounding vegetation. The species considered are characteristic of rangelands distributed across different regions of the state of Queensland [13].

3.6 Execution

The experimental workflow consists of seven steps, the steps that compose the experimental workflow are presented below:

(1) Environment Configuration - Google Colab using the L4 GPU runtime type;
(2) Dataset Download - *Cassava* and *DeepWeeds*;
(3) Sample Selection per Class - 50, 100, 150, 200 and 250;
(4) Embedding Extraction - Using the *embed()* function;
(5) Definition of the *YOLOClassifier* class - seed = 42, epochs = 50, batch = 32 and split = 0.1;
(6) Execution and Evaluation of Test Cases (DTC) - 10 runs and average of the results;
(7) Execution and Evaluation of Test Cases (10-fold cross-validation) - 10 runs and average of the results.

Before the test case selection stage, the data corresponding to all analyzed datasets were imported into the Google Colab environment using Python (version 3.12.12[10]). For each dataset, five independent experiments were conducted. In each experiment, a random selection of images belonging to all classes was performed using the *random*[11] library with the seed parameter set to 42, in order to create subsets containing 50, 100, 150, 200, and 250 samples per class. Subsequently, each subset was divided into two parts, corresponding to the data used for training and their respective associated labels.

The decision tree used in the application was built from the embeddings[12] extracted from the images belonging to each analyzed subset. These embeddings consist of dense, continuous, and low-dimensional vector representations of originally discrete data, functioning as a form of translation between perceptible image features and the numerical representation used by computational models. The extraction of these representations was performed using the *embed* function provided by the YOLO model, producing a fixed-length representation with 256 features. The resulting embeddings are organized into a tabular dataset together with the corresponding image identifiers and class labels. The image identifiers are retained only as metadata, allowing selected instances to be mapped back to the original image files.

After generating the decision tree, a recursive procedure was developed with the purpose of identifying all unique paths between the root node and each leaf node, following the definition of DTC criterion. Considering that each leaf is reached by a single path from the root, it becomes possible to represent these paths individually. Therefore, starting from the root and adopting a post-order traversal strategy, in which the nodes of the right subtree are visited first, followed by the nodes of the left subtree, the implemented recursive approach allows for systematically tracking each distinct path to the leaves. As a result, the procedure selects a set of test cases that ensures coverage of all leaves present in the analyzed decision tree model.

Each subset of images was used for training and evaluating the YOLO model. The configuration adopted for the model considered the following parameters: seed set to 42; epochs defined as 50; batch size of 32; and a split ratio of 0.1. Both the training and evaluation stages were conducted using 10-fold cross-validation. Finally, the YOLO models were evaluated using test cases selected according to the DTC criterion, and a comparison was then performed between the results of these tests and those obtained from random selection.

4 Results

To answer the RQ_1, the feasibility of using a DTC-based approach to guide the selection of test cases for ML models was investigated. The motivation is to obtain a test set that negatively impacts the performance of the evaluated model. The underlying hypothesis is that it is possible to select stronger test cases by leveraging the internal structure of decision tree models.

In this context, effectiveness was evaluated based on the impact of the test cases on the predictive performance of the analyzed model. The greater the negative impact on the model's performance, the more effective the test set was considered. In this study, random testing corresponds to 10-fold cross-validation, a technique that divides the dataset into 10 parts and repeats the training and testing of the model 10 times, producing an overall performance estimate.

According to the results presented in Tables 4, 6, 8, and 10, the evaluation of the YOLO models using test cases selected by the DTC criterion revealed a significant reduction in predictive performance across nearly all analyzed metrics. Considering the values of accuracy, micro F_1-score, and macro F_1-score, the results indicate

[7] https://www.tensorflow.org/datasets
[8] https://www.tensorflow.org/datasets/catalog/cassava
[9] https://www.tensorflow.org/datasets/catalog/deep_weeds
[10] https://www.python.org/downloads/release/python-31212/
[11] https://docs.python.org/3.12/library/random.html
[12] https://www.ultralytics.com/pt/glossary/embeddings

Table 3: Overview of the datasets used in the experiment

Datasets	Samples	Classes	Samples per Class
Cassava	5,656	5	[466; 1,443; 773; 2,658; 316]
DeepWeeds	17,509	9	[1,125; 1,064; 1,031; 1,022; 1,062; 1,009; 1,074; 1,016; 9,106]

that test cases selected based on the DTC criterion produce more challenging inputs.

4.1 Cassava

The experiments conducted with the *Cassava* dataset considered five distinct classes: *Cassava Bacterial Blight (CBB)*; *Cassava Brown Streak Disease (CBSD)*; *Cassava Green Mite (CGM)*; *Cassava Mosaic Disease (CMD)*; and *Cassava Healthy*. For each experiment, balanced subsets containing 50, 100, 150, 200, and 250 samples per class were evaluated. The results presented in Tables 4, 5, 6, and 7 allow for a direct comparison of the models' performance when evaluated using test cases derived from the DTC approach and from random selection.

It is observed that the results associated with random selection present higher values for all subsets. For the test cases selected according to the DTC criterion, the improvement in accuracy from 0.56 to 0.74 and in macro F_1-score from 0.40 to 0.62 stands out. While random selection shows macro metrics close to the micro metrics, suggesting more uniform performance, the results obtained with DTC reveal variations across classes.

Increasing the number of samples per class leads to a growth in structural complexity under the DTC criterion, as evidenced by the increase in the average number of nodes, leaves, and paths. Despite this, the execution time of the experiment using DTC was similar to that of random selection, and the proposed criterion achieved more efficient results while selecting fewer test cases.

4.2 DeepWeeds

The *DeepWeeds* dataset consists of nine classes: *Chinee apple*; *Lantana*; *Negative*; *Parkinsonia*; *Parthenium*; *Prickly acacia*; *Rubber vine*; *Siam weed*; and *Snake weed*. Similar to the previous experiment, balanced subsets with 50, 100, 150, 200, and 250 samples per class were considered. The comparison between DTC and random selection highlights consistent differences in the behavior of the metrics, as presented in Tables 8, 9, 10, and 11.

In the selection of test cases following the DTC criterion, an improvement in performance metrics is observed as the number of samples increases, with accuracy ranging from 0.67 to 0.88 and macro F_1-score from 0.58 to 0.82. In random selection, a progressive convergence between micro and macro metrics can be observed, indicating more homogeneous behavior across classes. On the other hand, the results obtained with DTC maintain differences between these metrics throughout the experiments, suggesting a greater ability of the approach to highlight class-specific classification difficulties. This aspect is particularly important in datasets with a larger number of classes, such as *DeepWeeds*, where detailed per-class performance analysis becomes essential.

Structural complexity also shows significant growth in the results obtained using the DTC criterion, with a substantial increase in the average number of nodes, leaves, and paths as the number of samples per class increases. As in the *Cassava* dataset, DTC achieved more efficient results using fewer test cases, while maintaining a runtime similar to that of random selection.

Overall, random selection presents higher average metrics and more stable behavior as the number of samples per class increases. However, the results obtained through test case selection using the DTC approach reveal greater sensitivity in identifying performance variations across classes and across different data configurations. Consequently, the results indicate that test cases derived using DTC constitute an effective strategy to support performance analysis in image classification tasks.

4.3 Hypothesis Testing

To verify whether the proposed criterion contributes to the selection of more effective test cases in evaluating the performance of the YOLO model, a two-tailed Wilcoxon signed-rank test was applied to analyze differences in the central tendency of the metrics considered in the experiment, specifically accuracy, micro F_1-score, macro F_1-score, micro recall, macro recall, micro precision and macro precision. Before performing the statistical tests, the normality of the data was assessed using the Shapiro-Wilk test. The results presented in Tables 5, 7, 9, and 11 indicated *p-values* lower than 0.05 in the Shapiro–Wilk test, leading to the rejection of the null hypothesis of normality for the analyzed metrics.

The comparison between the approaches was conducted using a two-tailed Wilcoxon signed-rank test for paired samples, since the results obtained by DTC and by the random selection strategy were generated under the same experimental conditions and using the same subset sizes (50, 100, 150, 200, and 250 samples per class). Therefore, each pair of observations represents results corresponding to the same experimental scenario, differing only in the test case selection strategy.

Based on the obtained results, it can be observed that the proposed criterion is capable of selecting test cases that are statistically different from those generated by the random selection strategy. In general, the results indicate that the use of DTC significantly impacts the analyzed performance metrics. The complete results are presented in Tables 12 and 13 through the *W-values* and their corresponding *p-values*.

For the *Cassava* dataset, it is observed that all analyzed metrics present *p-values* below the significance level of 0.05. This result indicates the rejection of the null hypothesis of equality between the means, suggesting that there are statistically significant differences between the results obtained with DTC and those generated by the random selection strategy. In addition, the *W-values* indicates that the results obtained with DTC are lower than those obtained with random selection for all considered metrics. This behavior can be observed both in metrics based on micro averages and in metrics based on macro averages, with the latter showing even larger differences.

Table 4: Results of the *Cassava* dataset – DTC

Subsets	Accuracy	F_1-score (micro)	Recall (micro)	Precision (micro)	F_1-score (macro)	Recall (macro)	Precision (macro)	Nodes	Leaves	Paths	Tests	Time (min)
50 samples per class	0.56	0.56	0.56	0.56	0.40	0.37	0.49	102.4	51.7	51.7	3.9	8.69
100 samples per class	0.63	0.63	0.63	0.63	0.58	0.57	0.62	187.2	94.1	94.1	5.7	15.42
150 samples per class	0.64	0.64	0.64	0.64	0.51	0.56	0.54	278.2	139.6	139.6	9.3	21.2
200 samples per class	0.71	0.71	0.71	0.71	0.58	0.61	0.63	363.6	182.3	182.3	5.8	29.0
250 samples per class	0.74	0.74	0.74	0.74	0.62	0.64	0.67	437.2	219.1	219.1	5.5	35.48

Table 5: Descriptive statistics and normality tests of the *Cassava* dataset – DTC

Statistic	Accuracy	F_1-score (micro)	Recall (micro)	Precision (micro)	F_1-score (macro)	Recall (macro)	Precision (macro)
Max	0.74	0.74	0.74	0.74	0.62	0.64	0.67
Min	0.56	0.56	0.56	0.56	0.40	0.37	0.49
Mean	0.66	0.66	0.66	0.66	0.54	0.55	0.59
Std Dev	0.06	0.06	0.06	0.06	0.08	0.09	0.07
Shapiro-Wilk (W)	W=0.9520 p=0.0414	W=0.9520 p=0.0414	W=0.9520 p=0.0414	W=0.9520 p=0.0414	W=0.9354 p=0.0089	W=0.9324 p=0.0068	W=0.9526 p=0.0438

Table 6: Results of the *Cassava* dataset – 10-fold Cross-Validation

Subsets	Accuracy	F_1-score (micro)	Recall (micro)	Precision (micro)	F_1-score (macro)	Recall (macro)	Precision (macro)	Tests	Time (min)
50 samples per class	0.65	0.65	0.65	0.65	0.64	0.65	0.68	25	8.25
100 samples per class	0.72	0.72	0.72	0.72	0.71	0.72	0.74	50	15.27
150 samples per class	0.77	0.77	0.77	0.77	0.76	0.77	0.78	75	21.1
200 samples per class	0.76	0.76	0.76	0.76	0.76	0.76	0.77	100	28.84
250 samples per class	0.75	0.75	0.75	0.75	0.75	0.75	0.76	125	35.33

Table 7: Descriptive statistics and normality tests of the *Cassava* dataset – 10-fold Cross-Validation

Statistic	Accuracy	F_1-score (micro)	Recall (micro)	Precision (micro)	F_1-score (macro)	Recall (macro)	Precision (macro)
Max	0.77	0.77	0.77	0.77	0.76	0.77	0.78
Min	0.65	0.65	0.65	0.65	0.64	0.65	0.68
Mean	0.73	0.73	0.73	0.73	0.72	0.73	0.75
Std Dev	0.04	0.04	0.04	0.04	0.05	0.04	0.04
Shapiro-Wilk (W)	W=0.8992 p=0.0005	W=0.8992 p=0.0005	W=0.8992 p=0.0005	W=0.8992 p=0.0005	W=0.9034 p=0.0006	W=0.8992 p=0.0005	W=0.9455 p=0.0223

For the *DeepWeeds* dataset, a behavior distinct from that observed for *Cassava* can be noted. For the accuracy, micro F_1-score, micro recall, and micro precision, no statistically significant differences were observed between the approaches (W = 423.0, *p-value* = 0.0593). In these metrics, the results obtained with DTC tend to be lower than those achieved with random selection. In contrast, for the macro-averaged metrics (macro F_1-score, macro recall, and macro precision), statistically significant differences were identified (W = 244.0, *p-value* = 0.0001; W = 270.0, *p-value* = 0.0004; W = 253.0, *p-value* = 0.0002, respectively).

The results indicate that the DTC criterion selects more effective test cases. For the *Cassava* dataset, all evaluated metrics presented *p-values* below the significance level of 0.05, indicating statistically significant differences between the results obtained with DTC and those generated by random selection. In all cases, the results obtained with DTC are lower than those achieved with the random strategy, indicating that DTC selects more effective test cases for this dataset. For the *DeepWeeds* dataset, no statistically significant differences were observed for the accuracy, micro F_1-score, micro recall, and micro precision, although a tendency towards lower values with DTC can be noted. In contrast, the macro-averaged metrics presented statistically significant differences, with DTC consistently resulting in lower metric values than those obtained through random selection.

5 Discussion

To answer RQ_1, it was investigated whether the DTC criterion can guide the selection of test cases in image classification. Since the purpose of the criterion is to select test cases that differ from the training data, it was assumed that test sets aligned with the DTC criterion tend to lead to a reduction in the performance of the evaluated model. It is considered that ML models have a higher probability of error when exposed to unseen inputs or inputs that are significantly different from those observed during training.

In this scenario, the effectiveness of the test inputs was evaluated in relation to the dataset used for training the model. Effectiveness was defined by the negative impact that the test data exert on the predictive performance of the analyzed model. Therefore, the greater the effectiveness of the test set, the greater the expected degradation in the model's performance. In this study, the so-called

Table 8: Results of the *DeepWeeds* dataset – DTC

Subsets	Accuracy	F_1-score (micro)	Recall (micro)	Precision (micro)	F_1-score (macro)	Recall (macro)	Precision (macro)	Nodes	Leaves	Paths	Tests	Time (min)
50 samples per class	0.67	0.67	0.67	0.67	0.58	0.60	0.60	202.8	101.9	101.9	5.1	13.88
100 samples per class	0.89	0.89	0.89	0.89	0.78	0.8	0.78	382.0	191.5	191.5	9.5	21.42
150 samples per class	0.83	0.83	0.83	0.83	0.74	0.75	0.77	530.8	265.9	265.9	11.4	34.35
200 samples per class	0.84	0.84	0.84	0.84	0.73	0.74	0.74	692.2	346.6	346.6	6.2	42.4
250 samples per class	0.88	0.88	0.88	0.88	0.82	0.83	0.83	857.6	429.3	429.3	9.6	54.26

Table 9: Descriptive statistics and normality tests of the *DeepWeeds* dataset – DTC

Statistic	Accuracy	F_1-score (micro)	Recall (micro)	Precision (micro)	F_1-score (macro)	Recall (macro)	Precision (macro)
Max	0.89	0.89	0.89	0.89	0.82	0.83	0.83
Min	0.67	0.67	0.67	0.67	0.58	0.60	0.60
Mean	0.82	0.82	0.82	0.82	0.73	0.74	0.74
Std Dev	0.08	0.08	0.08	0.08	0.08	0.08	0.08
Shapiro-Wilk (W)	W=0.8858 p=0.0002	W=0.8858 p=0.0002	W=0.8858 p=0.0002	W=0.8858 p=0.0002	W=0.9170 p=0.0018	W=0.8960 p=0.0004	W=0.9124 p=0.0013

Table 10: Results of the *DeepWeeds* dataset – 10-fold Cross-Validation

Subsets	Accuracy	F_1-score (micro)	Recall (micro)	Precision (micro)	F_1-score (macro)	Recall (macro)	Precision (macro)	Tests	Time (min)
50 samples per class	0.81	0.81	0.81	0.81	0.80	0.81	0.83	45	12.75
100 samples per class	0.86	0.86	0.86	0.86	0.86	0.86	0.87	90	21.19
150 samples per class	0.87	0.87	0.87	0.87	0.87	0.87	0.87	135	34.12
200 samples per class	0.89	0.89	0.89	0.89	0.89	0.89	0.89	180	41.92
250 samples per class	0.91	0.91	0.91	0.91	0.91	0.91	0.91	225	53.84

Table 11: Descriptive statistics and normality tests of the *DeepWeeds* dataset – 10-fold Cross-Validation

Statistic	Accuracy	F_1-score (micro)	Recall (micro)	Precision (micro)	F_1-score (macro)	Recall (macro)	Precision (macro)
Max	0.91	0.91	0.91	0.91	0.91	0.91	0.91
Min	0.81	0.81	0.81	0.81	0.80	0.81	0.83
Mean	0.87	0.87	0.87	0.87	0.87	0.87	0.87
Std Dev	0.03	0.03	0.03	0.03	0.04	0.03	0.03
Shapiro-Wilk (W)	W=0.9353 p=0.0088	W=0.9353 p=0.0088	W=0.9353 p=0.0088	W=0.9353 p=0.0088	W=0.9424 p=0.0168	W=0.9353 p=0.0088	W=0.9473 p=0.0264

Table 12: The results of the two-tailed Wilcoxon signed-rank tests for the *Cassava* dataset

Metric	*W*	*p-value*
Accuracy	340.5	0.0111
F_1-score (micro)	340.5	0.0111
Recall (micro)	340.5	0.0111
Precision (micro)	340.5	0.0111
F_1-score (macro)	176.0	<0.0001
Recall (macro)	197.0	<0.0001
Precision (macro)	200.0	0.0001

Table 13: The results of the two-tailed Wilcoxon signed-rank tests for the *DeepWeeds* dataset

Metric	*W*	*p-value*
Accuracy	423.0	0.0593
F_1-score (micro)	423.0	0.0593
Recall (micro)	423.0	0.0593
Precision (micro)	423.0	0.0593
F_1-score (macro)	244.0	0.0001
Recall (macro)	270.0	0.0004
Precision (macro)	253.0	0.0002

random tests correspond to the evaluation of the model using 10-fold cross-validation. In this procedure, the model is trained using subsets of the available data and subsequently evaluated on test subsets.

The answer to RQ_1 is based on the metrics accuracy, micro F_1-score, macro F_1-score, micro recall, macro recall, micro precision and macro precision, which are used as indicators of the effectiveness of the test cases. The analysis of the accuracy values indicates that the DTC criterion selects more challenging test cases for the model, Tables 4, 6, 8, and 10. As an example, for the YOLO model trained on the *Cassava* dataset with 50 samples per class, the accuracy value obtained during 10-fold cross-validation was 0.65. However, when evaluated with test cases selected according to the DTC criterion, the performance dropped to 0.56. In the case of the *DeepWeeds* dataset with 50 samples per class, the accuracy value

obtained under 10-fold cross-validation was 0.81, and when subjected to test cases derived from DTC, the performance decreased to 0.67.

The statistical analysis also reveals significant differences between the evaluated scenarios. Considering accuracy, there is a statistically significant difference between DTC (mean = 0.66) and random selection (mean = 0.73) on the *Cassava* dataset, Table 5 and Table 7. For the *DeepWeeds* dataset, the accuracy of DTC had a mean of 0.82, while random selection achieved 0.87, Table 9 and Table 11.

Table 12 presents the results obtained for the *Cassava* dataset, the micro F_1-score and macro F_1-score metrics resulted *W-values* of 340.5 and 176.0, with *p-values* of 0.0111 and <0.0001, respectively. Similarly, for the *DeepWeeds* dataset (Table 13), the *W-values* were 423.0 and 244.0, with *p-values* of 0.0593 and 0.0001, respectively.

These results indicate that the DTC criterion can guide the selection of test cases based on existing data, transformed into the internal representation of decision tree models. This process helps reveal incorrect model behaviors, that is, situations in which ML models fail to produce correct predictions for different test inputs.

6 Threats to Validity

The experiments were conducted using balanced subsets containing different numbers of samples per class, which may influence the behavior of the observed performance metrics. In addition, the training and evaluation of the YOLO model[13] depend on specific parameters and conditions that may impact the obtained results.

The goal of this study was not to optimize hyperparameters but to compare test-selection strategies under controlled conditions. The subsets (50, 100, 150, 200, and 250 samples per class) were chosen to evaluate DTC across different dataset sizes while respecting the computational limits of the Google Colab L4 GPU runtime type. The model was trained for 50 epochs because preliminary experiments showed performance stabilization. All parameters were kept identical across experiments to ensure a fair comparison between DTC and 10-fold cross-validation. Therefore, the observed differences are attributable to the test-selection strategy rather than variations in model configuration.

The effectiveness was measured using metrics widely employed in the evaluation of ML models, such as accuracy, F_1-score, recall, and precision. Although these metrics are suitable for the quantitative analysis of performance in classification tasks, they may not fully capture qualitative aspects related to the model's behavior in specific scenarios.

The results of this study were obtained from experiments conducted on two datasets, i.e.,*Cassava* and *DeepWeeds*. Despite differences in the number of classes and the visual variability of the images, we cannot rule out that these findings may not generalize to other contexts, application domains, or machine learning model architectures. The effectiveness of the DTC criterion may vary when applied to datasets with different characteristics, varying levels of imbalance, or greater visual complexity.

[13]Version *yolo26n-cls.pt*.

7 Conclusion

This study investigates the effectiveness of the DTC criterion for selecting test cases in the evaluation of image classification models, considering its application in comparison with the 10-fold cross-validation procedure. The model analyzed throughout the experiments corresponds to the YOLO architecture, subsequently fine-tuned on the *Cassava* and *DeepWeeds* datasets. The evaluation is primarily centered on the F_1-score, which serves as the main performance metric. To the best of our knowledge, this study represents the first systematic investigation of the DTC criterion in the context of testing image classification models. The DTC criterion leverages the internal structure of a machine learning model (i.e., decision trees) to derive test cases that can be used to evaluate image classification systems. By adapting this criterion to a domain in which it has not previously been applied, this study contributes novel empirical evidence regarding its effectiveness relative to randomly selected test data.

The results observed on the *Cassava* and *DeepWeeds* datasets indicate that test cases derived according to the DTC criterion enable a more detailed characterization of the model's behavior under different experimental configurations. The application of test cases derived using the DTC criterion revealed significant differences compared to randomly selected test cases. Therefore, the obtained results indicate that the DTC criterion constitutes a relevant strategy to complement traditional evaluation approaches in ML, such as cross-validation.

As future work, the investigation of the application of the DTC criterion in object detection tasks using the YOLO model is highlighted. In this context, the aim is to analyze the effectiveness of the approach in evaluating models capable of simultaneously performing localization and classification of multiple instances within the same image, thereby expanding the understanding of the criterion's potential in more complex scenarios and under different experimental configurations.

Artifact Availability

The experimental artifacts, including the notebooks and experimental results, are available at https://doi.org/10.5281/zenodo.22135292, licensed under the **Creative Commons Attribution 4.0 International License (CC BY 4.0)**.

Acknowledgements

The authors thank the support of Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), process numbers 306719/2025-8 and 406941/2025-4.

References

[1] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLOS ONE* 10, 7 (07 2015), 1–46. doi:10.1371/journal.pone.0130140

[2] M. Delamaro, M. Jino, and J. Maldonado. 2016. *Introdução Ao Teste De Software - 2ª Edição.* Elsevier, Rio de Janeiro.

[3] R. A. Fisher. 1936. The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics* 7, 2 (1936), 179–188. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1469-1809.1936.tb02137.x doi:10.1111/j.1469-1809.1936.tb02137.x

[4] Jianmin Guo, Yu Jiang, Yue Zhao, Quan Chen, and Jiaguang Sun. 2018. DLFuzz: differential fuzzing testing of deep learning systems. In *Proceedings of the 2018 26th*

ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (Lake Buena Vista, FL, USA) *(ESEC/FSE 2018)*. Association for Computing Machinery, New York, NY, USA, 739–743. doi:10.1145/3236024.3264835

[5] Jinhan Kim, Robert Feldt, and Shin Yoo. 2019. Guiding deep learning system testing using surprise adequacy. In *Proceedings of the 41st International Conference on Software Engineering* (Montreal, Quebec, Canada) *(ICSE '19)*. IEEE Press, Piscataway, NJ, USA, 1039–1049. doi:10.1109/ICSE.2019.00108

[6] Jinhan Kim, Jeongil Ju, Robert Feldt, and Shin Yoo. 2020. Reducing DNN labelling cost using surprise adequacy: an industrial case study for autonomous driving. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Virtual Event, USA) *(ESEC/FSE 2020)*. Association for Computing Machinery, New York, NY, USA, 1466–1476. doi:10.1145/3368089.3417065

[7] Ana Lorena, Katti Faceli, Tiago Almeida, Andre de Carvalho, and João Gama. 2025. *Inteligência Artificial: uma abordagem de Aprendizado de Máquina - 3ª Edição*. LTC, Rio de Janeiro.

[8] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. 2018. DeepGauge: multi-granularity testing criteria for deep learning systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering* (Montpellier, France) *(ASE '18)*. Association for Computing Machinery, New York, NY, USA, 120–131. doi:10.1145/3238147.3238202

[9] José Augusto N. G. Manzano and Jayr Figueiredo de Oliveira. 2019. *Algoritmos: Lógica Para Desenvolvimento de Programação de Computadores - Edição Revisada e Atualizada*. Editora Érica, São Paulo, SP. https://books.google.com.br/books?id=qYywDwAAQBAJ

[10] A.C. Müller and S. Guido. 2016. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media, Incorporated, Sebastopol, CA, USA. https://books.google.com.br/books?id=q5pnAQAACAAJ

[11] Ernest Mwebaze, Timnit Gebru, Andrea Frome, Solomon Nsumba, and Jeremy Tusubira. 2019. iCassava 2019 Fine-Grained Visual Categorization Challenge. arXiv:1908.02900 [cs.CV] https://arxiv.org/abs/1908.02900

[12] Augustus Odena, Catherine Olsson, David Andersen, and Ian Goodfellow. 2019. TensorFuzz: Debugging Neural Networks with Coverage-Guided Fuzzing. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, Long Beach, California, USA, 4901–4911. https://proceedings.mlr.press/v97/odena19a.html

[13] Alex Olsen, Dmitry A. Konovalov, Bronson Philippa, Peter Ridd, Jake C. Wood, Jamie Johns, Wesley Banks, Benjamin Girgenti, Owen Kenny, James Whinney, Brendan Calvert, Mostafa Rahimi Azghadi, and Ronald D. White. 2019. DeepWeeds: A Multiclass Weed Species Image Dataset for Deep Learning. *Scientific Reports* 9, 2058 (14 2 2019), 2058. doi:10.1038/s41598-018-38343-3

[14] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2019. DeepXplore: automated whitebox testing of deep learning systems. *Commun. ACM* 62, 11 (Oct. 2019), 137–145. doi:10.1145/3361566

[15] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Las Vegas, NV, USA, 779–788. doi:10.1109/CVPR.2016.91

[16] Vincenzo Riccio, Gunel Jahangirova, Andrea Stocco, Nargiz Humbatova, Michael Weiss, and Paolo Tonella. 2020. Testing machine learning based systems: a systematic mapping. *Empirical Softw. Engg.* 25, 6 (Nov. 2020), 5193–5254. doi:10.1007/s10664-020-09881-0

[17] Sebastião Santos, Beatriz Silveira, Vinicius Durelli, Rafael Durelli, Simone Souza, and Marcio Delamaro. 2021. On Using Decision Tree Coverage Criteria for Testing Machine Learning Models. In *Proceedings of the 6th Brazilian Symposium on Systematic and Automated Software Testing* (Joinville, Brazil) *(SAST '21)*. Association for Computing Machinery, New York, NY, USA, 1–9. doi:10.1145/3482909.3482911

[18] Beatriz Silveira, Vinicius Durelli, Sebastião Santos, Rafael Durelli, Marcio Delamaro, and Simone Souza. 2023. Test Data Selection Based on Applying Mutation Testing to Decision Tree Models. In *Proceedings of the 8th Brazilian Symposium on Systematic and Automated Software Testing* (Campo Grande, MS, Brazil) *(SAST '23)*. Association for Computing Machinery, New York, NY, USA, 38–46. doi:10.1145/3624032.3624038

[19] Youcheng Sun, Xiaowei Huang, Daniel Kroening, James Sharp, Matthew Hill, and Rob Ashmore. 2019. Testing Deep Neural Networks. arXiv:1803.04792 [cs.LG] https://arxiv.org/abs/1803.04792

[20] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, and A. Wesslén. 2012. *Experimentation in Software Engineering*. Springer Berlin Heidelberg, Berlin, Heidelberg. https://books.google.com.br/books?id=QPVsM1_U8nkC

[21] Xiaofei Xie, Tianlin Li, Jian Wang, Lei Ma, Qing Guo, Felix Juefei-Xu, and Yang Liu. 2022. NPC: Neuron Path Coverage via Characterizing Decision Logic of Deep Neural Networks. *ACM Trans. Softw. Eng. Methodol.* 31, 3, Article 47 (apr 2022), 27 pages. doi:10.1145/3490489

[22] Yuanyuan Yuan, Qi Pang, and Shuai Wang. 2023. Revisiting Neuron Coverage for DNN Testing: A Layer-Wise and Distribution-Aware Criterion. In *Proceedings of the 45th International Conference on Software Engineering* (Melbourne, Victoria, Australia) *(ICSE '23)*. IEEE Press, Piscataway, NJ, USA, 1200–1212. doi:10.1109/ICSE48619.2023.00107

[23] Jie M. Zhang, Mark Harman, Lei Ma, and Yang Liu. 2022. Machine Learning Testing: Survey, Landscapes and Horizons. *IEEE Transactions on Software Engineering* 48, 1 (2022), 1–36. doi:10.1109/TSE.2019.2962027