Empirical Insights of Test Selection Metrics under Multiple Testing Objectives and Distribution Shifts

JINGYU ZHANG, Hong Kong Metropolitan University, Hong Kong
FAN WANG, City University of Hong Kong, Hong Kong
JACKY KEUNG, City University of Hong Kong, Hong Kong
YIHAN LIAO, City University of Hong Kong, Hong Kong
YAN XIAO*, Shenzhen Campus of Sun Yat-sen University, China
LEI MA, University of Tokyo, Japan and University of Alberta, Canada

Deep learning (DL)-based systems can exhibit unexpected behavior when exposed to out-of-distribution (OOD) scenarios, posing serious risks in safety-critical domains such as malware detection and autonomous driving. This underscores the importance of thoroughly testing such systems before deployment. To this end, researchers have proposed a wide range of test selection metrics designed to effectively select inputs. However, prior evaluations of metrics reveal three key limitations: (1) narrow testing objectives, for example, many studies assess metrics only for fault detection, leaving their effectiveness for performance estimation unclear; (2) limited coverage of OOD scenarios, with natural and label shifts are rarely considered; (3) Biased dataset selection, where most work focuses on image data while other modalities remain underexplored. Consequently, a unified benchmark that examines how these metrics perform under multiple testing objectives, diverse OOD scenarios, and different data modalities is still lacking. This leaves practitioners uncertain about which test selection metrics are most suitable for their specific objectives and contexts. To address this gap, we conduct an extensive empirical study of 15 existing metrics, evaluating them under three testing objectives (fault detection, performance estimation, and retraining guidance), five types of OOD scenarios (corrupted, adversarial, temporal, natural, and label shifts), three data modalities (image, text, and Android packages), and 13 DL models. In total, our study encompasses 1,640 experimental scenarios, offering a comprehensive evaluation and statistical analysis.

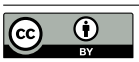
CCS Concepts: • **Software and its engineering → Software testing and debugging**.

Additional Key Words and Phrases: Deep Learning Testing, Test Selection Metrics, Empirical Study

ACM Reference Format:
Jingyu Zhang, Fan Wang, Jacky Keung, Yihan Liao, Yan Xiao, and Lei Ma. 2026. Empirical Insights of Test Selection Metrics under Multiple Testing Objectives and Distribution Shifts. *Proc. ACM Softw. Eng.* 3, FSE, Article FSE058 (July 2026), 24 pages. https://doi.org/10.1145/3797086

*Corresponding Author.

Authors' Contact Information: Jingyu Zhang, Hong Kong Metropolitan University, Hong Kong, Hong Kong, fzhang@hkmu.edu.hk; Fan Wang, City University of Hong Kong, Hong Kong, Hong Kong, fan.wang@my.cityu.edu.hk; Jacky Keung, City University of Hong Kong, Hong Kong, Hong Kong, jacky.keung@cityu.edu.hk; Yihan Liao, City University of Hong Kong, Hong Kong, Hong Kong, yihanliao4-c@my.cityu.edu.hk; Yan Xiao, Shenzhen Campus of Sun Yat-sen University, Shenzhen, China, xiaoy367@mail.sysu.edu.cn; Lei Ma, University of Tokyo, Tokyo, Japan and University of Alberta, Edmonton, Canada, ma.lei@acm.org.



ACM 2994-970X/2026/7-ARTFSE058
https://doi.org/10.1145/3797086

1 Introduction

Nowadays, Deep Neural Networks (DNNs) have been integrated into many software applications due to their great success in performance and efficiency, including safety-critical applications like autonomous driving systems. However, recent research has shown that DNNs with high accuracy on the in-distribution (ID) samples may demonstrate misbehavior on out-of-distribution (OOD) scenarios, such as corrupted [33, 37] and adversarial samples [23, 65]. The fact that deployed DNNs often encounter OOD inputs poses severe threats to user safety and pinpoints the importance of comprehensive testing of DNNs before deployment. A common DNN testing process is to select a small representative set of inputs (often regarded as *test suite*) that achieves the testing objective (e.g., detecting DNN faults) [34, 51, 70]. Three main testing objectives are studied in the field of DNN testing [34]: Objective 1. fault detection, which selects test samples to expose as many faults as possible. We denote a *fault* in DNNs as the root reason that causes mispredictions [1, 6]; Objective 2. performance estimation, which selects and labels samples with the goal of providing an accurate estimation of the model performance (e.g., accuracy) in the unlabeled operational set; Objective 3. retraining guidance, which selects and labels the samples that can improve the model performance as much as possible. Given that the labeling process requires extensive expert labor, the main challenge in DNN testing is to achieve the desired objective with a very limited labeling budget. To address this, a great number of test selection metrics has been proposed, where we roughly categorize them into six types based on their characteristics: uncertainty-based [20, 77], diversity-based [1, 50, 65], surprise-based [37], sampling-based [25, 47], clustering-based [12, 69], and hybrid [33, 82] metrics. They aim to efficiently select and prioritize a small number of test inputs. The selected inputs are then manually labeled and are expected to achieve the targeted objective. For example, DeepGini [20] selects uncertain inputs based on the softmax score, with the intuition that uncertain inputs can be fault-revealing and therefore enhance the model by retraining. As for the performance estimation objective, metrics like Cross Entropy-based Sampling [47] and DeepReduce [82] are proposed to use different techniques to maximize the distribution similarity between the selected test suite and the operational set. However, we observe three key limitations in the existing work:

(1) Narrow testing objectives studied: It remains unclear whether metrics are effective beyond the objectives they were originally designed for. Among the 15 metrics [1, 12, 20, 25, 33, 37, 47, 50, 65, 69, 77, 82] studied in this paper, 9 focus on a single objective, 5 consider two objectives, while none have been evaluated across all three. Yet practitioners may need metrics that serve multiple objectives simultaneously. For example, while DeepGini [20] is known to be effective for fault detection, can it also be used for performance estimation? Similarly, could metrics not designed for fault detection perform even better?

(2) Limited coverage of OOD scenarios: Evaluating metrics under diverse testing environments is essential to assess their generalizability. However, our preliminary survey of 26 test selection papers shows that 23 [2, 6, 11, 12, 20, 25, 26, 29, 35, 37, 38, 45–48, 51, 60, 65, 70, 78, 79, 81] of them focus primarily on original test data, 8 [12, 33, 37, 45, 46, 65, 69, 79] on corrupted shifts, 11 [12, 20, 28, 29, 33, 37, 51, 65, 69, 70, 81] on adversarial shifts, only 2 [33, 78] on natural shifts and none for label shifts.

(3) Biased dataset selection: To ensure generalizability, the effectiveness of test selection metrics should be evaluated across diverse data modalities. However, among 26 surveyed papers, image data dominate the evaluation (23/26 papers [2, 6, 11, 12, 20, 25, 26, 28, 29, 33, 35, 37, 38, 47, 51, 60, 65, 69, 70, 78, 79, 81, 82]), while text data (2/26 [33, 79]) and Android applications (1/26 [65]) remain critically underexplored.

To address these shortcomings, we conduct **a unified and extensive empirical study** that (1) compares 15 existing metrics across three testing objectives, five types of OOD scenarios (corruption, adversarial, temporal, natural, and label shifts), and three data modalities (image, text, and Android applications). To ensure fairness, we also identify the evaluation criteria that reliably reflect each testing objective. **Findings:** Metrics that encourage test input diversity are effective in objective 2, outperforming those specifically designed for this objective. Metrics designed for Objective 3 are also outperformed by other metrics. Moreover, there is no clear change in metrics' performance under different OOD types. (2) Our study also analyzes the metrics' performance under different selection budgets. **Findings:** metrics' performance on objectives 1 and 2 is largely stable across budgets, while there is considerable fluctuation for objective 3. (3) We examine the time efficiency of all 15 metrics for practical usage. **Findings:** A clear performance-speed trade-off exists for best-performing metrics in objective 1. No such trade-off is observed in objectives 2 and 3.

Our study makes the following two main contributions.

(1) To the best of our knowledge, we undertake the first extensive study aimed at systematically assessing the performance of 15 test selection metrics on three testing objectives under five OOD types with three data modalities and 13 DNNs. Our study encompasses a total of 1,640 experimental scenarios, providing a comprehensive evaluation and statistical analysis.

(2) We present findings and provide implications for researchers and practitioners in DNN testing. For example, we recommend utilizing diversity-based metrics for an accurate performance estimation.

2 Background and Related Work

2.1 DNN Testing

To guarantee the robustness and reliability of DNNs, it is essential to conduct comprehensive testing to ensure their performance when facing data with diverse distributions (e.g., OOD scenarios) [35]. While DNNs can demonstrate good performance on data collected from the ID scenarios, they have demonstrated severe misbehavior when encountering OOD inputs (e.g., adversarial inputs [23]) that are highly likely to occur during deployment. A common way to conduct DNN testing is to prepare representative and diverse test suites, label test inputs, and evaluate the model on these curated sets [34]. The diversity of test suites can stem from data modalities, OOD scenarios, and task types. In this paper, we have explored three modalities: image [41, 76], text [52], and Android packages [3], together with five types OOD shifts [59] including corrupted, adversarial, natural, temporal, and label shifts, under two task types, classification and regression.

2.2 Test Selection

To reduce the manual labeling cost of large test suites [34], various test selection metrics have been proposed to choose a small yet effective test suite under a budget. In this paper, we study 15 metrics and categorize them into six types: (1) **Uncertainty-based metrics** select inputs where the model is most uncertain, Entropy (Ent) [77] and DeepGini (Gini) [20] employ different equations to compute uncertainty scores from the softmax probability. (2) **Diversity-based metrics**: Neuron Coverage (NC) [65] and K-multisection Neuron Coverage (KMNC) [50] measure the diversity of a test suite through the coverage of neurons. Geometric Diversity (GD) and Standard Deviation (STD) [1] directly measure the input feature diversity using pre-trained feature extractors. (3) **Surprise-based metrics**: Likelihood-based surprise adequacy (LSA) and distance-based surprise adequacy (DSA) [37] use kernel density estimation and Euclidean distance, respectively, to quantify the surprise of an input compared to the training data. (4) **Sampling-based metrics** utilize different sampling strategies to match the distribution of the testing set: Cross Entropy-based Sampling

(CES) [47] minimizes the cross entropy between the selected set and the whole testing set. DeepEST (EST) [25] utilizes adaptive sampling. We also studied random sampling (Rand) as a baseline metric. (5) **Clustering-based metrics** partition tests into groups and select representative ones from each group (Multiple-Boundary Clustering and Prioritization (MCP) [69] and Practical Accuracy Estimation (PACE) [12]). (6) **Hybrid metrics** employ multiple strategies/phases: DeepReduce (DR) [82] selects the suite to guarantee the same NC and further optimizes by maximizing distribution similarity. Distribution-aware test selection (DAT) [33] selects uncertain ID inputs (using Gini) and randomly selects OOD inputs.

2.3 Test Optimization

These metrics are demonstrated effective in different testing objectives. There are three important objectives in test optimization [34]: (1) fault detection [1, 20, 21, 37], (2) performance estimation [12, 25, 47, 82], and (3) retraining guidance [33, 69].

For (1), the metrics are expected to select inputs that can reveal as many faults as possible. For example, uncertainty [20, 77], diversity [1, 50, 65], and surprise-based metrics [37] are demonstrated to be effective in fault detection. For (2), the metrics (e.g., sampling-based [25, 47], clustering-based [12]) are expected to select a small set of test inputs that can precisely estimate the accuracy of the whole unlabeled testing set. For (3), the metrics (e.g., MCP [69], DAT [33]) are expected to select inputs that enhance DNN as much as possible through retraining.

2.4 Empirical Study on DNN Testing

Several works conduct empirical studies to investigate the effectiveness of existing metrics [33, 35, 51, 70, 79]. For example, Hu *et al.* [33] studies how metrics perform in retraining guidance when facing three types of OOD shifts. Their later work [35] challenges metrics targeting fault detection and performance estimation using new ID test data (e.g., correctly classified but uncertain data) on image classification datasets. Ma *et al.* [51] explores metrics targeted at fault detection on image data with adversarial shifts only. Moreover, Sun *et al.* [73] assess the effectiveness of metrics in fault detection and retraining guidance only on image classification with limited OOD scenarios. Similarly, Demir *et al.* [16] evaluate only the fault-revealing capability of uncertainty-based metrics under original and synthetically generated OOD data for image classification. In addition to test selection, Berend *et al.* [8] analyze how the test generators affect the data distribution, and do not study the multi-objective effectiveness of test selection. We note that none of these works have provided a unified evaluation benchmark of existing metrics under all three testing objectives, while also covering diverse data modalities and OOD types, which highlights the significance of our study.

3 Study Design

This section first outlines the notations and provides a general overview of our study. We then formally define three testing objectives and their associated evaluation criteria. Finally, we introduce the types of OOD data used to construct diverse testing environments and present our research questions.

3.1 Notations

- $\mathcal{X}$ and $\mathcal{Y}$ denotes the input space and the output space, respectively;
- $M : \mathcal{X} \to \mathcal{Y}$ denotes the DNN model under test;
- $(x, y) \in \mathcal{X} \times \mathcal{Y}$ denotes a single input sample x and its ground truth label y;
- $(X_{test}, Y_{test}) \subseteq \mathcal{X} \times \mathcal{Y}$ denotes the testing dataset with samples X_{test} and the corresponding set of ground truth labels Y_{test}. Note that it may contain both ID and OOD data samples;

- $f(M, X, Y)$ denotes a function measuring the performance (accuracy in our context) of M when predicting Y from X;
- $(X_s, Y_s) \subset (X_{test}, Y_{test})$ denotes the data samples X_s and corresponding labels Y_s in the test suite selected by the test selection metric;
- $(X_t, Y_t) = (X_{test} \backslash X_s, Y_{test} \backslash Y_s)$ denotes that X_t is the set of all inputs in X_{test} that are not in X_s, with Y_t as the corresponding label set;
- $P(x|y)$ denotes the class-conditional distribution of seeing input x given that the label is y;
- $P(y)$ denotes the distribution of labels y;
- $P(y|x)$ denotes the probability of a label y given an input x;

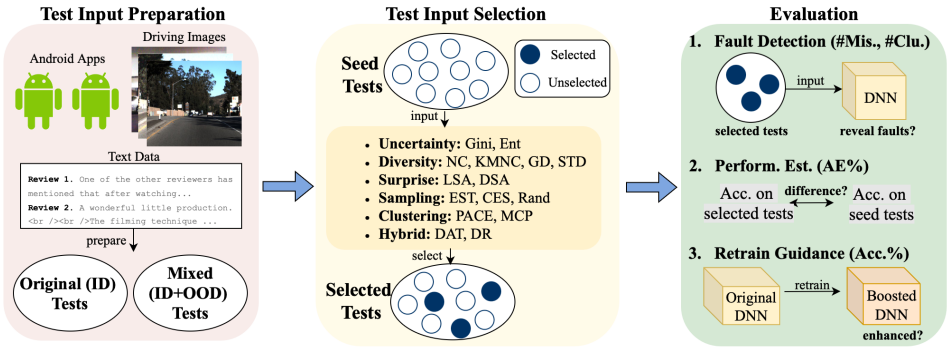


Fig. 1. Overview of our study: 1. Test Input Preparation, which prepares a testing set that contains either original (ID) or mixed (ID+OOD) test samples. 2. Test Input Selection, which uses the 15 studied metrics to select inputs. 3. Evaluation: This part evaluates the selected inputs with three testing objectives: fault detection (#Mis. and #Clu.), performance estimation (assessed by AE%), and retraining guidance (Acc.%).

3.2 Overview

Figure 1 outlines the overview, containing three main parts: test input preparation, test input selection, and evaluation. In **test input preparation**, to assess the generalized performance of test selection metrics, we design the testing sets (containing seed inputs for selection) across diverse data modalities and distribution shifts. Our benchmark evaluates *three data modalities*: Android applications (AndroZoo [3]), images (MNIST [41], Udacity [76]), and text data (IMDb [52]). This mitigates the significant dataset selection bias observed in prior studies, where image data dominate (23 out of 26 surveyed papers), while text data (2 papers) and Android applications (1 paper) remain critically underexplored (see Section 1 for details of the survey). Furthermore, we extend our evaluation beyond original testing sets to include *five OOD types*: corruption, adversarial, temporal, natural, and label shifts. This addresses another gap in the literature, as prior work has focused predominantly on original (23/26), corrupted (8/26), and adversarial (11/26) shifts, with natural shifts rarely studied (2/26) and label shifts (0/26) entirely unexplored. The inclusion of these overlooked shifts enhances the comprehensiveness and practical relevance of our evaluation benchmark. For **test input selection**, we employ 15 widely-studied metrics to select inputs from the curated testing sets. These metrics, proposed between *2014 and 2023*, are categorized into *six types* based on their characteristics (see Section 2.2 for details). Each selected test suite is then **evaluated** on *three testing objectives*: fault detection (9/15 metrics have studied this objective), measured by #Mis. and #Clu., performance estimation (4/15), measured by AE%, and retraining guidance (6/15), measured by

Acc.%. This comprehensive threefold analysis provides a novel comparison, as prior work has not evaluated metrics across all these objectives.

3.3 Testing Objectives

We provide details of testing objectives we would like to achieve with the selected test suite, and their corresponding evaluation criteria, to accurately assess the metrics' performance.

3.3.1 Fault Detection. We denote a *fault* in DNNs as the cause of mispredictions, such that multiple mispredictions may be attributed to the same fault. Formally, given an unlabeled testing dataset X_{test}, a DNN model M, and a labeling budget bg, fault detection objective aims at selecting a subset X_s from X_{test} such that $|X_s| = bg$ and $X_s = \text{argmin}_{X_i \subset X_{test}, |X_i|=bg} f(M, X_i, Y_i)$. Y_i denotes the labels corresponding to X_i. We use two criteria to evaluate this objective: the number of mispredictions [20, 25, 26, 79] and the number of clusters [1, 7].

Number of Mispredictions (#Mis.): We measure the number of mispredictions made by the DNN under test on the selected test suite. We treat classification and regression differently. For classification tasks, we count the number of samples whose predicted class is different from their labeled class. For regression tasks, we set a threshold and count the number of samples whose offset (i.e., difference between the predicted value and the ground truth) is greater than the threshold δ. Specifically, for the Udacity dataset (steering angle prediction task), we evaluate $\delta \in [0°, 2.5°, ..., 22.5°, 25°]$ and report the average number of mispredictions over these thresholds. Note that #Mis. may not always reflect unique DNN faults when different mispredictions are attributed to the same underlying faults [1, 7]

Number of Clusters (#Clu.): To mitigate the limitation of #Mis., we also adopt a clustering-based fault estimation approach [1, 7] to evaluate the selected test suite, where clustering is performed on the mispredicted samples. Each cluster of mispredicted inputs in the feature space is treated as a distinct DNN fault, i.e., mispredicted inputs within the same cluster are assumed to be mispredicted for the same underlying reason. This approach (the clustering pipeline) contains three steps: feature extraction (FE), dimensionality reduction (DR), and clustering algorithm (CA). The effectiveness of this criterion depends on clustering quality, where we examine the cluster-to-fault correspondence in Section 5.1 and identify the optimal pipeline.

3.3.2 Performance Estimation. Given an unlabeled testing set X_{test} (also known as operational set in the context of operational testing) which possibly given as inputs to the target DNN M and a labeling budget bg, the performance estimation objective aims at selecting a subset X_s from X_{test} and obtain corresponding labels Y_s such that $|X_s| = bg$ and $X_s = \text{argmin}_{X_i \subseteq X_{test}, |X_i|=bg} |f(M, X_i, Y_i) - f(M, X_{test}, Y_{test})|$. In other words, the small labeled test suite is expected to accurately estimate the performance of DNN (e.g., accuracy) on the large unlabeled testing set.

Absolute Error (AE%) is a commonly used criterion to measure whether the suite selected by the metrics achieves the performance estimation objective [12, 47], which is evaluated by $AE = |\hat{acc}_i - acc|$. $\hat{acc}_i$ and acc refer to the estimated and actual accuracy (in percentage), respectively. In the implementation, acc is computed from the whole testing set, whereas $\hat{acc}_i$ is calculated from the small selected set. For regression tasks, we use 1-RMSE as the accuracy. RMSE=$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(pred_i - gt_i)^2}$, where $pred_i$ and denote the predicted value and the ground truth label for input i, respectively. N denotes the total number of inputs.

3.3.3 Retraining Guidance. Given a pre-trained DNN M with parameter weights trained on the training set (X_{train}, Y_{train}). Retraining guidance objective aims to select a subset X_s from the unlabeled set X_{test} and obtain the corresponding label Y_s, where $|X_s| = bg$ and $X_s = \text{argmax}_{X_i \subseteq X_{test}, |X_i|=bg}$

$f(M', X_t, Y_t)$. The improved DNN M' is usually retrained on either $(X_{train} \cup X_s, Y_{train} \cup Y_s)$ (denoted as type II retraining) or (X_s, Y_s) (type I retraining). As illustrated in the evaluation part (green) in Figure 1, the goal is to maximize the accuracy improvement of M' (boosted DNN in orange) over M (original DNN in yellow).

Accuracy Improvement (Acc.%) is used to measure the capability of retraining guidance [33, 37, 79]. To avoid data leakage [36], the set (X_t, Y_t) used for evaluating the retrained model M' does not overlap with the set (X_s, Y_s) that is used in retraining. Formally, $acc' = acc(M', X_t, Y_t) - acc(M, X_t, Y_t)$. $acc(M, X_t, Y_t)$ denotes the accuracy (in percentage) of M on the set (X_t, Y_t).

3.4 OOD Data Construction

In the real world, there are diverse types of OOD scenarios that DNNs may encounter in deployment, and it is important to understand the effectiveness of these selection metrics on different types of OOD scenarios. In this paper, we formally categorize covariate and label distribution shifts [59], covering corrupted, adversarial, natural, temporal, and label shifts in total.

3.4.1 Covariate shift. It refers to the changes in the input feature distribution, while the label distribution given the input is unchanged. This can be demonstrated by the change in lighting, blur, or noise for images [62, 65]. For malware data, this is commonly observed in data collected at different times, known as temporal covariate shift [24, 57].

Definition. *Covariate shift is defined as the case where* $P_{id}(y|x) = P_{ood}(y|x)$ *and* $P_{id}(x) \neq P_{ood}(x)$.

For MNIST, IMDb, and Udacity datasets, we use three types of covariate shifts: corrupted, adversarial, and natural shifts [68]. For AndroZoo (data collected in 2017 [3]), we use temporal (data in 2018 and 2019 [3]), adversarial, and natural shifts. To simulate a generalized adversarial scenario, for MNIST, Udacity, and AndroZoo, we use a combination of adversarial images generated by FGSM [23], BIM [40], and PGD [54], on three equal parts, respectively. For IMDb, we use Probability Weighted Word Saliency (PWWS) [66] and Banana Word Swap [75] to generate a natural and semantically valid adversarial text. For corrupted OOD shifts, we publicly available corrupted datasets from [62] and [79] for MNIST and IMDb, respectively. To the best of our knowledge, there is no corrupted version of the Udacity dataset in the literature. Therefore, we created the corrupted dataset, where images are corrupted using all methods from the Python package `imagecorruptions` [58], which contains 15 common visual degradations (e.g., blur, noise, weather effects) for real-world image data. For natural covariate shift, we use EMNIST [15], Dave [72], Drebin [5], and Customer Review [32] for MNIST, Udacity, AndroZoo, and IMDb, respectively.

3.4.2 Label shift. The label shift refers to changes in the distribution of the label variable y, but the feature distributions conditional on the labels are fixed [13].

Definition. *It is defined as the case where* $P_{id}(x|y) = P_{ood}(x|y)$ *and* $P_{id}(y) \neq P_{ood}(y)$.

The goal is to change the label distribution in the testing set $P_{ood}(y)$ away from the training distribution $P_{id}(y)$. For MNIST, we simulate label shifts by sampling a skewed distribution, i.e., we choose roughly 34% for digit 0, 15% each for digits 1-3, 5% each for digits 4-6, 2% each for digits 7-9. For IMDb, we simulate sentiment skew by sampling 80% positive and 20% negative reviews, reflecting real-world user feedback distributions. For the AndroZoo dataset, malware examples are much fewer than goodware. We simulate the label shift by oversampling malware samples and undersampling goodware samples (we choose 80% malware and 20% goodware) to create a realistic high-threat environment. For the Udacity dataset, steering angles are predominantly centered around 0 (straight driving). To simulate label shift, we emphasize sharp turns by reducing the frequency of near-zero steering angles and increasing the presence of extreme left and right turns. Specifically, we set 40% of samples with $y < -0.2$, 40% with $y > 0.2$ and only 20% represent near-zero angles ($|y| \leq 0.2$).

3.5 Research Questions

RQ1.1. Selection of optimal clustering pipelines. It is crucial to validate whether each cluster can reliably represent a DNN fault. This research question investigates (1) the clustering quality and (2) the cluster-to-fault correspondence. For (1), we study several candidate techniques used in the three-step clustering pipeline (described in Section 3.3.1) and leverage Silhouette [67] and DBCV [61] scores to assess the clustering quality of each candidate pipeline and identify the best one for each dataset. For (2), we conduct feature pattern inspection and cluster-specific retraining for the best pipeline to validate whether clusters can meaningfully represent different faults.

RQ1.2. Selection of optimal retraining processes. To evaluate retraining guidance fairly, a uniform and effective retraining process ensures comparability across metrics and reveals the true margin of improvement achievable by selected test suites. In this research question, we compare type I (retrain on (X_s, Y_s)) and type II (retrain on $(X_{train} \cup X_s, Y_{train} \cup Y_s)$) retraining processes.

RQ2. Performance of test selection metrics under multifold testing objectives. Evaluating selection metrics under multiple objectives is essential for understanding their practical effectiveness, yet this aspect is missing in existing studies. We provide a unified evaluation benchmark with statistical analysis (Non-Parametric Scott-Knott Effect Size Difference test [74]) that compares 15 metrics across 30 settings under four evaluation criteria (#Mis., #Clu., AE%, Acc.%).

RQ3. Performance of test selection metrics under different selection budgets. This analyzes the stability and scalability of the studied metrics under varying labeling budgets, which is a critical factor when practitioners face strict constraints on labeling resources. To answer this, we provide heatmap plots demonstrating the ranking of each metric across different budgets for each evaluation criterion.

RQ4. Time efficiency of test selection metrics. Efficient test selection is also an important aspect to consider when deploying the metrics in practice. In this question, we evaluate the time efficiency of all studied metrics.

4 Experimental Setup

In this section, we provide a summary of the test selection metrics investigated in this study, as well as the selection budgets. Then, we describe datasets and DNNs used in the experiments.

4.1 Test Selection Metrics

In this paper, we study 15 test selection metrics, as summarized in Table 1. The year of proposed metrics ranges from 2014 to 2023, including classic metrics (e.g., NC in 2017 [65]) and state-of-the-art metrics (e.g., GD and STD in 2023 [1]). Five metrics (Gini [20], Ent [77], DSA [37], MCP [69], DAT [33]) rely on the softmax confidence score that is not available in regression models, therefore, we only report their performance on classification tasks. As shown in the table, these 15 metrics have been studied for only one or two testing objectives, e.g., CES [47], PACE [12], and DR [82] investigate their effectiveness for performance estimation only. However, how they perform under other important objectives is still underexplored. We also find that metrics studied for performance estimation lack analyzing their performance on OOD scenarios, i.e., most of them evaluate on the original ID testing set only. This paper provides an evaluation benchmark that analyzes 15 widely used metrics on three important testing objectives on five different OOD shifts, providing insightful findings and recommendations for researchers and practitioners in the SE community. For implementation details of these metrics (e.g., layer selection for LSA, OOD detectors for DAT, auxiliary variables for EST), please refer to our online repository due to limited space. Similar to existing works [12, 33], we select four different sizes of test suites to assess the effectiveness of the metrics, i.e., our studied **selection budgets are 50, 100, 150, and 200**.

Table 1. Summary of 15 test selection metrics studied in this paper. 'Type' indicates the most suitable type the metric belongs to based on its characteristics. 'C' and 'R' denote whether this metric is applicable to classification and regression tasks, respectively. Under the 'Obj.', we have listed which testing objective this metric has been studied in the literature: 1: fault detection, 2: performance estimation, 3: retraining guidance. 'Year' denotes the time the metric was proposed.

Metrics	Type	Description	C	R	Obj.	Year
Gini [20]	Uncertainty	Select the most uncertain inputs	✓	✗	1, 3	2020
Ent [77]	Uncertainty	Select the most uncertain inputs	✓	✗	1	2014
NC [65]	Diversity	Select inputs with maximum NC	✓	✓	1, 3	2017
KMNC [50]	Diversity	Select inputs with maximum KMNC	✓	✓	1	2018
GD [1]	Diversity	Select test suite with maximum diversity	✓	✓	1	2023
STD [1]	Diversity	Select test suite with maximum diversity	✓	✓	1	2023
LSA [37]	Surprise	Select inputs with maximum surprise	✓	✓	1, 3	2019
DSA [37]	Surprise	Select inputs with maximum surprise	✓	✗	1, 3	2019
CES [47]	Sampling	Select inputs that guarantee distribution similarity	✓	✓	2	2019
PACE [12]	Clustering	Select representative inputs from each cluster	✓	✓	2	2020
DR [82]	Hybrid	Select inputs with multi-objective optimization	✓	✓	2	2020
EST [25]	Sampling	Select inputs with adaptive sampling	✓	✓	1, 2	2021
MCP [69]	Clustering	Select inputs at decision boundary areas.	✓	✗	3	2020
DAT [33]	Hybrid	Selects uncertain ID inputs, randomly selects OOD inputs	✓	✗	3	2022
Rand	Sampling	Randomly select inputs without replacement	✓	✓	NA	NA

4.2 Datasets and DNNs

As shown in Table 2, we design 30 experiment settings, which vary in the testing set, the DNN models, and the OOD types. The testing sets cover three different data types (image, text, and Android packages), with both classification (MNIST [41], AndroZoo [3], and IMDb [52]) and regression (Udacity [76]) tasks. The DNN models are constructed with different structures, thus different accuracies on the testing sets (the column 'Perf.'). Five different distribution shifts are used in the experiments: corrupted, adversarial, label, temporal, and natural shifts (the column 'OOD Type'). Testing sets with distribution shifts are constructed with 50% ID samples and 50% OOD samples (see Section 3.4 for details). As a result, we have a total of $10 \times 4 \times 30 + 5 \times 4 \times 22$ combinations, summing up to 1,640 unique scenarios.

5 Results

This section presents the results of experiments we used to answer the research questions.

5.1 RQ1.1. Selection of optimal clustering pipelines.

5.1.1 The clustering quality. We conduct preliminary clustering experiments with different candidate techniques for each step on the mispredicted samples from the original testing set of each dataset, averaging over all studied models. Recall that the clustering pipeline contains three steps: feature extraction (FE), dimensionality reduction (DR), and clustering (CA to denote clustering algorithms). We use Silhouette [67] and DBCV [61] scores for evaluation, where both range from -1 to 1. A higher value indicates a better clustering result. For the AndroZoo dataset, we assess DeepDrebin and BasicDNN [43, 44] for FE, where we use the first hidden layer for both [80], three techniques for DR: Principal Component Analysis (PCA) [64], Uniform Manifold Approximation and Projection (UMAP) [56], and Gaussian Random Projection (GRP) [9], and four CA: K-Means [53], Hierarchical Agglomerative Clustering (HAC) [39], Density-based spatial clustering of applications with noise

Table 2. Summary of all 30 experiment settings used in our evaluation. '#Params' denotes the number of parameters in the model. 'Perf.' indicates the model accuracy on the corresponding testing set. '#Tests' is the number of samples in the testing set.

ID	Testing Set	Model	#Params	Perf.	#Tests	OOD Type
1	MNIST [41]	LeNet-1 [42]	7,206	94.86%	10,000	Original
2		LeNet-4 [42]	77,998	96.79%	10,000	Original
3		LeNet-5 [42]	89,698	98.68%	10,000	Original
4	Udacity [76]	Dave2V1 [10]	2,116,983	96.35%[1]	5,614	Original
5		Dave2V2 [22]	2,116,983	95.67%	5,614	Original
6		Dave2V3 [71]	3,276,225	97.06%	5,614	Original
7		Epoch [27]	18,969,665	98.41%	5,614	Original
8	AndroZoo [3]	DeepDrebin [44]	2,404,802	99.23%	21,336	Original
9		BasicDNN [43]	1,626,081	99.22%	21,336	Original
10	IMDb [52]	Linear [31]	640,033	87.48%	25,000	Original
11		LSTM [17]	692,785	85.47%	25,000	Original
12		GRU [17]	680,753	84.68%	25,000	Original
13		Transformer [63]	653,566	87.57%	25,000	Original
14	MNIST-C [62]	LeNet-5 [42]	89,698	92.07%	10,000	Corrupted covariate shift
15	MNIST-Adv	LeNet-5 [42]	89,698	50.06%	10,000	Adversarial covariate shift
16	MNIST-label	LeNet-5 [42]	89,698	98.97%	10,000	Label shift
17	MNIST-EMNIST	LeNet-5 [42]	89,698	92.76%	10,000	Natural covariate shift
18	Udacity-C	Epoch [27]	18,969,665	97.19%	5,614	Corrupted covariate shift
19	Udacity-Adv	Epoch [27]	18,969,665	56.05%	5,614	Adversarial covariate shift
20	Udacity-label	Epoch [27]	18,969,665	95.85%	5,614	Label shift
21	Udacity-Dave	Epoch [27]	18,969,665	98.44%	5,614	Natural covariate shift
22	AndroZoo-2018	DeepDrebin [44]	2,404,802	96.33%	16,000	Temporal covariate shift
23	AndroZoo-2019	DeepDrebin [44]	2,404,802	96.36%	16,000	Temporal covariate shift
24	AndroZoo-Adv	DeepDrebin [44]	2,404,802	57.28%	21,336	Adversarial covariate shift
25	AndroZoo-label	DeepDrebin [44]	2,404,802	97.84%	21,336	Label shift
26	AndroZoo-Drebin	DeepDrebin [44]	2,404,802	84.48%	21,336	Natural covariate shift
27	IMDb-C [79]	Transformer [63]	653,566	74.99%	25,000	Corrupted covariate shift
28	IMDb-Adv	Transformer [63]	653,566	42.89%	25,000	Adversarial covariate shift
29	IMDb-label	Transformer [63]	653,566	88.02%	25,000	Label shift
30	IMDb-Customer	Transformer[63]	653,566	78.90%	7,550	Natural covariate shift

(DBSCAN) [19], and Hierarchical DBSCAN (HDBSCAN) [55]. We perform a similar procedure for the IMDb dataset, which tests pre-trained BERT [18], RoBERTa [49], and ELECTRA [14] for FE (we use the penultimate layer for all [18]), the same candidates for DR and CA as in AndroZoo. For a fair pipeline comparison, we use our optimally tuned hyperparameter values (see online repository for details). To streamline our analysis, we conduct ablation studies by varying techniques in one step while holding the other two constant. For instance, when evaluating FE candidates, we fix UMAP for DR and DBSCAN for CA for all datasets, following the approach recommended by [7]. Then, we fix the identified optimal FE method and DBSCAN for CA when ablating DR candidates. Finally, we fix the optimal FE and DR methods when ablating CA candidates. Based on the Silhouette and DBCV scores in Table 3, the optimal pipelines are **(DeepDrebin, UMAP, DBSCAN)** for the AndroZoo dataset and **(RoBERTa, UMAP, DBSCAN)** for IMDb. Note that for image datasets (MNIST and Udacity), we directly use the best pipeline **(ResNet-50 [30], UMAP, DBSCAN)** validated by [7], which achieves average scores of 0.69 and 0.47 for MNIST and Udacity, respectively.

5.1.2 The cluster-to-fault correspondence. To validate whether the resulting cluster from the best pipeline indeed represents a DNN fault, we conduct feature pattern inspection and cluster-specific retraining validation [1]. Figure 2 displays the heatmaps of a randomly selected cluster from the best clustering pipeline and a randomly selected cluster obtained from another pipeline. Due to limited space and similar patterns, we only show results of the Udacity and AndroZoo datasets (see online

Table 3. The Silhouette and DBCV results of candidate pipelines for AndroZoo and IMDb datasets, average over all studied models. 'Deep': DeepDrebin; 'Basic': BasicDNN; 'RBT': RoBERTa; 'BT': BERT; 'ET': ELECTRA; 'DB': DBSCAN; 'HDB': HDBSCAN, 'KM': K-Means.

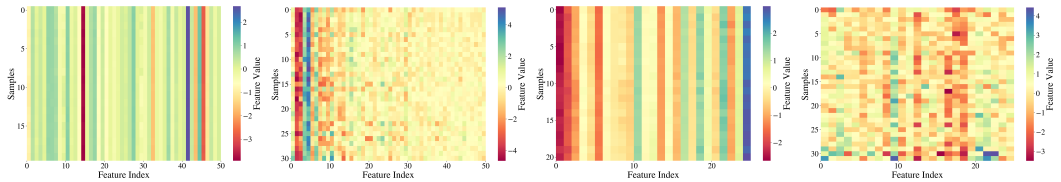
(a) AndroZoo.

	FE		DR			CA			
Score	Deep	Basic	UMAP	PCA	GRP	DB	HDB	HAC	KM
Sil.	0.83	0.58	0.83	0.06	0.04	0.83	0.56	0.33	0.51
DBCV	0.91	0.75	0.91	-0.47	-0.46	0.91	0.32	0.24	0.02
Avg	0.87	0.67	0.87	-0.21	-0.21	0.87	0.44	0.29	0.27

(b) IMDb.

	FE			DR			CA			
	RBT	BT	ET	UMAP	PCA	GRP	DB	HDB	HAC	KM
	0.91	0.16	0.43	0.91	0.25	0.26	0.91	0.87	0.32	0.17
	0.93	0.06	0.43	0.93	0.01	0.01	0.93	0.90	0.43	-0.28
	0.92	0.11	0.43	0.92	0.13	0.14	0.92	0.89	0.38	-0.06

repository for MNIST and IMDb). Each row represents a single mispredicted sample, and each column represents a (reduced) feature. We colored the feature values using the spectral colormap. By comparing the 'best' and 'other' heatmaps, we can see that the high-quality cluster displays a uniform feature pattern across all mispredicted samples within this cluster, which suggests a shared underlying root cause of the fault. On the other hand, the low-quality cluster demonstrates dispersed and inconsistent feature patterns across mispredicted samples within that cluster.



(a) Udacity (best) (b) Udacity (other) (c) AndroZoo (best) (d) AndroZoo (other)

Fig. 2. Feature visualization of samples in a high-quality cluster ('best', Udacity: ResNet-50, UMAP, DBSCAN; Andro: DeepDrebin, UMAP, DBSCAN) and a low-quality cluster ('other', Udacity: ResNet-50, PCA, HAC; Andro: BasicDNN, GRP, K-Means).

Table 4 demonstrates the results of cluster-specific retraining validation. Retraining is conducted using the 85% samples from C_i (we randomly pick three clusters from the best pipeline on one of the studied models). We validated and reported the accuracy improvement on the remaining 15% samples of C_i and on all samples in $C_{j \neq i}$. The results show that the retrained model is significantly more accurate on the cluster C_i for which it was retrained, and less accurate on other clusters ($C_j, j \neq i$) for all datasets. Specifically, the differences in accuracy improvement are 46%, 68%, 49%, and 63% for MNIST, Udacity, AndroZoo, and IMDb, respectively. This indicates that retraining fixes the fault represented by C_i but not the faults represented by C_j. Therefore, we can conclude that each cluster represents a unique fault and different clusters correspond to distinct faults.

Table 4. Cluster-specific retraining validation on all studied datasets.

	MNIST		Udacity		AndroZoo		IMDb	
Cluster ID C_i	C_i	$C_{j \neq i}$	C_i	$C_{j \neq i}$	C_i	$C_{j \neq i}$	C_i	$C_{j \neq i}$
Cluster 0	72%	21%	75%	19%	71%	26%	83%	12%
Cluster 1	67%	22%	99%	18%	86%	28%	56%	23%
Cluster 2	61%	20%	88%	20%	85%	42%	98%	13%
Average	67%	21%	87%	19%	81%	32%	79%	16%

Finding 1: Heatmap visualization demonstrates the coherence of intra-cluster feature patterns from the best pipeline, and cluster-specific retraining further validates cluster-to-fault correspondence.

5.2 RQ1.2. Selection of optimal retraining processes.

We report the results of type I and type II retraining for the Udacity and AndroZoo datasets, respectively. We adopt the recommended process (type II retraining) by Hu *et al.* [33] for MNIST and IMDb datasets. Specifically, We conduct experiments on all budgets (50, 100, 150, 200), and report the average results over all applicable test selection metrics. The hyperparameters of retraining (e.g., learning rate, optimizer) are set in the same way as in the original training process. Results are illustrated in Table 5. For Udacity, in 20 out of 32 cases, type II retraining demonstrates a higher accuracy improvement. Specifically, type II retraining gives an improvement of 3.72% on average, which is 2.06% higher than type I retraining. We found that type II retraining gives outstanding results compared to type I when there is an adversarial shift. For AndroZoo, we observe an opposite trend, where type I outperforms type II retraining in 25 out of 28 cases by a margin of 0.95% on average. We observe that the recommended retraining processes differ across data types. We analyze these differences from the perspectives of input geometry and feature characteristics. Udacity data are represented by spatially structured pixels with an input shape of (3,100,100), whereas AndroZoo data are represented with 10,000-dimensional binary vectors that are extremely sparse (around 99% zeros) without semantic continuity, where a small pattern change may flip classification. In this case, Type I allows the model to zoom in on discriminative features of new data more than Type II. For Udacity, dense image features are consistent in semantics. Type II preserves a full manifold of visual contexts for stable model updates, however, under some certain cases (ID 7, 18, 20 in Table 5a), Type I performs better since it allows the model to focus on the newly identified visual pattern.

Table 5. Comparison of type I and type II retraining processes for Udacity and AndroZoo datasets.

(a) Udacity.

ID	bg=50		bg=100		bg=150		bg=200	
	Type I	II	I	II	I	II	I	II
4	1.6	2.03	1.63	2.07	1.68	2.11	1.72	2.16
5	0	0.04	0	0.04	0	0.05	0	0.05
6	0.77	0.99	0.79	1.02	0.79	0.98	0.81	0.92
7	0.12	-1.43	0.12	-1.48	0.12	-1.51	0.12	-1.52
18	0.57	0.38	0.57	0.34	0.6	0.37	0.63	0.39
19	7.43	26.77	7.64	27.46	8.02	28.31	8.46	29.24
20	1.23	-1.27	1.26	-1.28	1.3	-1.21	1.35	-1.09
21	0.98	1.08	1	1.1	1.03	1.08	1.06	1.07
Avg.	1.59	3.57	1.63	3.66	1.69	3.77	1.77	3.9

(b) AndroZoo.

ID	bg=50		bg=100		bg=150		bg=200	
	Type I	II	I	II	I	II	I	II
8	-0.13	-0.24	-0.13	-0.35	-0.23	-0.89	-0.23	-0.5
9	0.01	-0.2	0.02	-0.2	-0.02	-0.34	-0.04	-0.51
22	-2.38	-3.18	-1.12	-2.42	-1.76	-4.46	-3.87	-5.14
23	-2.22	-2.12	-0.95	-2.41	-1.73	-4.65	-4.01	-4.7
24	4.14	1.17	4.82	3.11	4.48	1.74	3.75	0.26
25	0.04	-0.36	0.19	-0.34	0.06	-0.56	-0.25	-0.49
26	7.78	8.72	8.06	8.6	8.18	7.68	8	6.75
Avg.	1.03	0.54	1.56	0.86	1.28	-0.21	0.48	-0.62

Finding 2: For Udacity, type II outperforms type I retraining by approximately 3.06% on average. We observe an opposite trend for AndroZoo data, where type I outperforms type II by 0.95% on average.

5.3 RQ2. Performance of test selection metrics under multifold testing objectives.

5.3.1 Statistical significance. We conduct a statistical analysis to provide an overall ranking of all 15 metrics examined in this paper. To determine the statistical significance of the observed performance differences among multiple test selection metrics, we employ the Non-Parametric Scott-Knott Effect Size Difference (NPSK) test [74], which is a multiple comparison approach that

Table 6. Performance of 15 test selection metrics across 30 settings under four evaluation criteria: #Mis. (↑), #Clu. (↑), AE% (↓), and Acc.% (↑). Results are averaged over 4 budgets. Darker cells signify superior performance, while various colors denote the statistical significance among the 15 studied test selection metrics for each ID and criterion (Non-Parametric Scott-Knott ESD test with a p-value ≤ 0.05) (ID: 1-15).

	ID	Eval.	Rand	Gini	Ent	NC	KMNC	GD	STD	LSA	DSA	CES	PACE	EST	DR	MCP	DAT
MNIST	1	#Mis.	6.75	68.75	64.25	5.0	3.75	7.0	4.75	6.25	45.25	6.75	5.25	**38.0**	8.0	57.25	3.25
		#Clu.	1.5	12.75	12.25	1.25	1.0	**1.75**	0.5	1.0	7.50	**1.75**	1.0	7.0	1.5	12.00	0.50
		AE%	0.75	51.94	48.03	1.51	2.68	1.58	**1.86**	0.71	32.03	3.15	1.15	25.57	**1.92**	41.19	2.76
		Acc.%	3.94	3.41	3.46	**4.0**	3.97	4.1	4.12	**4.0**	3.71	3.86	4.04	3.6	3.76	3.39	3.84
	2	#Mis.	4.5	62.25	61.5	**12.75**	5.5	5.0	2.25	6.25	78.50	5.0	2.75	36.0	1.5	62.00	4.50
		#Clu.	1.25	8.0	8.0	2.25	1.5	1.0	0.0	1.0	12.75	0.75	0.5	**6.0**	0.0	9.25	0.75
		AE%	1.71	48.58	48.08	8.0	1.83	1.19	1.5	**2.0**	61.25	1.81	1.75	26.08	2.25	49.96	1.02
		Acc.%	**2.6**	2.04	1.99	2.52	2.63	2.52	2.8	2.52	1.99	2.19	2.68	2.2	2.11	1.66	2.05
	3	#Mis.	1.0	51.25	50.25	2.0	2.25	2.75	2.0	0.0	59.75	0.25	**3.25**	34.75	0.5	47.75	0.75
		#Clu.	0.0	8.25	9.75	0.0	0.25	0.25	0.0	0.0	11.75	0.0	0.0	**6.0**	0.0	8.75	0.00
		AE%	1.28	40.64	40.05	0.39	1.05	0.97	0.67	1.32	50.60	**1.15**	**1.18**	26.35	1.03	39.18	0.78
		Acc.%	0.96	0.45	0.39	0.76	**0.81**	0.73	0.92	0.7	0.42	0.28	0.83	0.55	0.19	0.12	0.46
Udacity	4	#Mis.	41.75	-	-	71.95	65.59	44.84	42.27	65.2	-	54.25	42.48	50.39	**36.7**	-	-
		#Clu.	8.66	-	-	13.82	12.75	9.48	8.77	11.55	-	**10.32**	9.68	12.41	6.25	-	-
		AE%	1.63	-	-	18.61	13.69	1.57	2.23	8.72	-	6.68	0.45	**3.51**	4.0	-	-
		Acc.%	1.8	-	-	**2.69**	2.23	2.31	3.08	2.8	-	0.93	3.14	1.53	0.3	-	-
	5	#Mis.	38.68	-	-	56.11	38.86	39.68	38.5	71.98	-	38.66	38.45	38.27	**22.3**	-	-
		#Clu.	8.82	-	-	12.52	8.27	8.45	8.68	14.18	-	8.61	8.84	8.34	**4.93**	-	-
		AE%	0.5	-	-	6.05	1.42	2.11	**1.71**	24.89	-	1.11	**1.88**	**1.78**	9.8	-	-
		Acc.%	**0.04**	-	-	0.04	0.03	0.04	0.04	0.1	-	0.02	0.03	0.03	0.01	-	-
	6	#Mis.	**34.86**	-	-	29.36	53.2	**33.34**	**33.55**	38.39	-	45.16	31.82	37.57	37.73	-	-
		#Clu.	7.86	-	-	5.16	10.73	**7.0**	8.3	**7.27**	-	10.48	**6.86**	8.68	7.8	-	-
		AE%	1.18	-	-	5.86	15.43	2.64	1.71	1.37	-	12.4	**3.35**	**3.32**	5.83	-	-
		Acc.%	1.29	-	-	**0.85**	0.56	0.82	0.65	0.79	-	1.28	0.62	1.46	1.08	-	-
	7	#Mis.	27.32	-	-	57.66	36.02	26.18	27.3	17.16	-	34.61	**19.64**	25.2	**21.5**	-	-
		#Clu.	5.91	-	-	10.52	6.82	**5.34**	5.95	1.86	-	7.18	3.0	4.73	3.98	-	-
		AE%	1.22	-	-	14.06	**3.31**	0.88	2.21	7.01	-	4.39	4.5	1.15	**3.2**	-	-
		Acc.%	**-1.37**	-	-	-1.49	-1.91	-1.86	-2.15	-1.86	-	-0.7	-2.38	-1.05	-0.45	-	-
AndroZoo	8	#Mis.	0.25	36.0	36.5	2.75	1.0	1.5	0.0	4.0	88.25	1.0	3.25	**20.25**	0.25	23.25	0.25
		#Clu.	4.25	4.5	8.0	**10.5**	**9.5**	3.25	5.25	**10.25**	16.50	**10.0**	3.0	4.5	14.75	2.50	13.25
		AE%	0.71	29.24	30.24	1.33	0.46	1.33	**0.84**	2.37	76.70	0.33	1.91	18.83	0.71	18.33	0.71
		Acc.%	0.02	**0.04**	0.0	0.08	0.09	0.09	-0.68	**0.04**	-2.12	-0.13	-0.08	0.0	-0.05	-0.06	0.02
	9	#Mis.	0.75	40.75	39.25	2.75	3.75	0.25	1.0	0.75	93.25	1.0	0.25	**29.0**	0.75	31.25	0.00
		#Clu.	0.0	7.0	6.5	0.0	0.25	0.0	0.0	0.0	13.75	0.0	0.0	4.25	0.0	**6.00**	0.00
		AE%	0.35	35.81	34.93	1.85	2.89	0.61	0.87	0.35	80.81	0.6	0.65	26.81	0.47	27.10	**0.78**
		Acc.%	0.05	-0.07	-0.01	0.02	**0.04**	0.07	-0.07	0.07	-0.31	-0.01	0.1	0.01	-0.0	-0.02	0.05
IMDb	10	#Mis.	21.25	68.5	68.5	7.0	8.75	21.0	20.5	7.0	123.00	23.0	24.0	44.5	28.0	68.25	**34.50**
		#Clu.	4.0	2.0	2.0	2.5	1.75	4.5	**2.75**	1.25	2.00	3.0	3.0	2.5	2.25	**2.75**	7.50
		AE%	2.55	39.9	39.9	9.77	8.56	3.46	3.37	9.77	82.57	**4.42**	2.9	17.86	8.9	40.48	11.78
		Acc.%	0.85	0.44	0.53	0.39	0.6	**0.81**	-1.47	0.26	-2.45	1.87	-0.76	0.77	3.88	0.87	0.55
	11	#Mis.	18.0	63.5	63.5	0.75	25.25	20.0	19.0	11.75	124.00	17.0	10.75	52.25	**43.75**	63.75	29.75
		#Clu.	3.0	3.0	2.75	0.0	2.75	1.75	2.0	2.0	**2.25**	**2.25**	1.75	**2.25**	2.75	**2.25**	5.25
		AE%	1.04	36.17	36.17	13.37	8.63	2.5	1.58	**3.79**	85.50	2.54	4.75	29.33	26.42	36.29	9.13
		Acc.%	-1.02	**-1.6**	-1.41	-1.88	**-1.56**	-1.3	-2.25	-1.81	-3.18	-1.88	-1.74	-1.3	-1.14	-1.90	-1.38
	12	#Mis.	24.5	59.75	59.75	1.5	22.25	24.25	25.75	9.0	116.25	21.25	23.25	54.0	49.5	61.75	**37.25**
		#Clu.	2.75	2.75	4.0	0.0	2.0	3.75	**3.0**	2.5	2.00	1.75	2.5	3.25	3.25	**3.00**	7.50
		AE%	2.04	28.08	28.08	18.67	7.79	1.34	2.25	11.59	73.20	**3.0**	1.71	24.37	20.28	29.62	9.37
		Acc.%	4.85	4.4	4.62	4.38	4.47	4.85	3.95	4.19	2.65	4.26	4.7	**4.58**	4.14	4.40	**4.59**
	13	#Mis.	18.0	62.75	62.75	0.25	20.25	23.0	17.5	8.5	118.75	21.5	15.75	48.0	**40.0**	62.00	31.00
		#Clu.	3.75	**3.0**	3.75	0.0	1.5	**3.0**	2.5	2.25	2.50	**3.0**	3.5	2.75	3.25	2.75	7.50
		AE%	4.12	35.08	35.08	15.38	**4.71**	4.33	2.17	8.21	80.83	1.75	3.38	24.17	15.33	34.92	8.54
		Acc.%	-0.95	-1.26	-1.37	-1.66	-1.34	-1.26	-1.94	-1.67	-2.85	-0.84	-1.36	**-1.14**	0.07	**-1.14**	-1.34
MNIST-C	14	#Mis.	14.25	76.75	72.0	6.75	5.75	10.25	9.0	10.25	115.00	4.25	7.5	**47.0**	12.0	69.75	16.00
		#Clu.	4.75	13.0	13.75	2.25	1.5	2.75	2.0	3.0	14.50	0.5	1.75	10.5	2.75	14.00	**7.00**
		AE%	3.15	54.03	50.61	**2.22**	**2.46**	1.58	1.26	1.79	86.15	4.76	1.97	30.28	1.61	49.61	4.69
		Acc.%	3.21	2.59	3.2	2.53	3.09	2.99	**2.83**	2.25	1.13	0.75	3.01	2.67	0.19	1.75	3.08
MNIST-Adv	15	#Mis.	61.25	118.25	117.5	1.75	15.0	62.5	65.0	63.0	125.00	**32.25**	58.75	107.0	59.5	106.50	**33.25**
		#Clu.	**11.25**	16.5	12.5	0.0	2.5	11.75	13.75	**11.25**	6.75	8.75	12.0	**10.5**	11.75	13.25	6.75
		AE%	**3.96**	45.1	44.1	48.36	37.23	**3.89**	2.02	1.07	50.06	22.19	2.84	36.44	2.72	35.69	23.15
		Acc.%	38.34	33.31	35.99	37.08	**37.42**	36.25	38.89	36.61	37.01	21.24	40.86	34.13	9.8	24.93	34.48

(Continued)

Table 7. Performance of 15 test selection metrics across 30 settings under four evaluation criteria: #Mis. (↑), #Clu. (↑), AE (↓), and Acc.% (↑) results are averaged over 4 budgets (ID: 16-30).

	ID	Eval.	Rand	Gini	Ent	NC	KMNC	GD	STD	LSA	DSA	CES	PACE	EST	DR	MCP	DAT
MNIST-label	16	#Mis.	1.75	46.75	47.0	1.0	1.5	1.5	1.75	1.0	55.25	0.5	3.5	**33.75**	1.0	42.25	1.75
		#Clu.	**0.25**	10.5	11.0	0.0	0.0	0.0	0.0	0.0	11.00	0.0	0.75	8.5	0.0	10.75	0.00
		AE%	1.0	39.51	39.3	0.47	1.08	0.85	**0.69**	0.6	46.55	0.78	3.28	25.43	0.36	35.89	0.96
		Acc.%	0.78	0.27	0.26	0.68	0.72	0.78	0.77	**0.7**	0.32	-0.03	0.8	0.45	0.24	0.19	0.80
MNIST-EMNIST	17	#Mis.	9.5	69.0	69.25	**13.0**	6.75	10.0	10.75	10.75	119.75	5.75	6.75	55.75	5.0	66.50	9.75
		#Clu.	1.75	15.0	12.5	2.75	2.25	2.75	3.0	3.5	18.50	1.25	2.0	**5.75**	1.5	10.75	3.25
		AE%	1.67	48.01	49.43	**3.01**	2.49	**2.8**	2.17	2.34	89.63	3.32	3.29	35.34	3.7	47.72	0.92
		Acc.%	5.97	5.34	5.3	5.57	5.57	5.83	6.18	5.69	5.04	4.34	6.27	5.42	3.56	4.36	**5.88**
Udacity-C	18	#Mis.	**31.48**	-	-	69.02	49.8	**34.77**	**32.39**	43.41	-	47.7	**34.25**	**32.16**	23.05	-	-
		#Clu.	0.5	-	-	7.75	3.25	1.0	**1.33**	4.33	-	4.67	**1.58**	**1.5**	-0.75	-	-
		AE%	2.04	-	-	20.84	7.68	2.33	3.46	4.61	-	7.82	**2.63**	1.27	5.77	-	-
		Acc.%	0.55	-	-	-0.18	0.17	0.37	0.34	0.48	-	**0.44**	0.33	0.61	0.14	-	-
Udacity-Adv	19	#Mis.	56.43	-	-	121.98	100.48	54.73	54.86	53.91	-	91.34	58.82	54.41	53.57	-	-
		#Clu.	**10.93**	-	-	21.98	18.55	**10.7**	**10.64**	10.23	-	15.41	**11.32**	9.68	**10.91**	-	-
		AE%	4.04	-	-	72.56	63.8	**5.12**	2.5	7.5	-	40.44	2.73	1.08	7.28	-	-
		Acc.%	24.68	-	-	**35.82**	32.32	29.44	40.18	39.49	-	13.59	40.66	20.64	4.06	-	-
Udacity-label	20	#Mis.	47.61	-	-	61.5	42.77	44.18	44.3	43.77	-	48.02	40.36	45.05	42.2	-	-
		#Clu.	11.16	-	-	10.64	8.64	10.34	9.84	9.8	-	10.98	8.66	8.89	8.95	-	-
		AE%	1.46	-	-	10.3	0.51	1.3	**2.16**	0.23	-	2.58	2.72	1.41	**2.32**	-	-
		Acc.%	**-1.04**	-	-	-1.28	-1.9	-1.76	-1.13	-1.22	-	-0.8	-1.27	-1.01	-0.66	-	-
Udacity-Dave	21	#Mis.	**28.25**	-	-	61.16	31.05	**28.36**	25.84	**28.32**	-	38.59	24.48	**28.25**	**27.18**	-	-
		#Clu.	6.18	-	-	12.95	6.3	6.18	4.68	**5.75**	-	8.41	4.57	**5.91**	**5.45**	-	-
		AE%	0.73	-	-	11.81	1.07	**1.49**	1.81	0.57	-	4.78	1.12	2.06	0.6	-	-
		Acc.%	1.38	-	-	0.49	**1.19**	1.21	0.94	0.94	-	1.16	0.84	1.37	0.62	-	-
AndroZoo-2018	22	#Mis.	2.75	55.75	55.75	11.5	11.75	3.5	4.0	3.75	120.00	3.0	4.25	**26.75**	5.25	30.00	2.00
		#Clu.	0.0	10.0	9.5	**4.25**	3.5	0.5	0.0	0.5	16.00	0.5	0.5	6.0	0.75	6.75	0.00
		AE%	1.67	41.78	41.78	6.32	7.07	0.94	**1.25**	1.82	91.24	1.13	0.32	20.41	0.95	18.49	2.05
		Acc.%	0.57	0.24	0.14	0.28	**0.39**	0.44	-14.21	0.34	-23.26	0.09	0.03	0.26	0.14	-0.12	0.46
AndroZoo-2019	23	#Mis.	5.25	46.5	46.5	**18.75**	8.75	3.25	3.75	12.25	125.00	3.5	3.75	22.0	2.0	**20.00**	4.00
		#Clu.	0.75	7.0	7.5	**4.75**	2.25	0.25	1.0	3.75	17.50	1.0	0.5	6.0	0.0	**4.50**	1.00
		AE%	1.67	32.12	32.12	14.54	**4.25**	1.52	1.5	6.75	96.46	1.69	0.86	14.62	2.29	10.50	0.86
		Acc.%	**0.76**	0.32	0.3	0.89	1.0	**0.76**	-13.89	**0.73**	-27.90	0.53	1.19	0.69	0.34	0.51	0.35
AndroZoo-Adv	24	#Mis.	11.0	39.75	40.0	65.75	10.5	11.5	11.75	**29.5**	124.00	16.0	19.5	19.25	17.5	20.25	**27.50**
		#Clu.	3.25	**9.0**	10.25	12.75	2.5	2.75	3.75	4.5	21.75	3.25	5.25	4.25	4.5	4.50	5.50
		AE%	1.33	25.5	25.63	46.96	1.08	1.25	2.71	12.83	90.29	**4.75**	9.04	8.92	6.63	8.67	12.13
		Acc.%	5.45	5.81	5.8	5.53	6.51	6.22	-0.72	**6.05**	-9.21	5.74	3.93	5.95	5.07	6.11	6.21
AndroZoo-label	25	#Mis.	3.75	**22.75**	**22.75**	8.75	3.75	1.5	1.25	7.5	117.50	2.25	2.25	28.0	5.25	42.25	2.25
		#Clu.	0.25	3.0	3.25	1.5	0.5	0.25	0.5	**2.5**	14.75	0.0	0.0	**2.75**	1.75	5.25	0.00
		AE%	**1.34**	19.36	19.36	5.11	0.8	**1.45**	1.78	3.94	91.02	1.17	1.03	21.86	2.34	35.69	0.44
		Acc.%	0.51	0.56	0.35	0.87	0.9	0.9	-2.73	0.75	-5.25	0.57	**0.79**	0.25	0.35	0.42	0.93
AndroZoo-Drebin	26	#Mis.	14.25	57.25	57.25	25.5	10.25	21.5	18.25	42.0	121.50	12.5	13.25	**35.0**	**34.25**	**33.25**	0.50
		#Clu.	3.25	6.75	5.5	3.75	2.75	6.5	3.25	9.75	22.50	3.75	3.5	**6.0**	6.75	7.00	0.00
		AE%	2.5	29.79	29.79	8.79	4.3	**5.12**	3.79	19.7	84.58	2.71	2.92	13.5	18.45	14.54	12.46
		Acc.%	8.81	5.71	3.45	**9.74**	**9.75**	9.63	10.0	10.16	7.42	6.03	10.8	8.24	3.48	7.59	9.27
IMDb-C	27	#Mis.	34.75	61.25	61.25	0.75	41.0	34.0	32.75	72.25	120.75	29.75	19.25	59.5	36.75	62.75	**47.75**
		#Clu.	4.5	2.0	3.0	0.0	2.5	**3.5**	**3.75**	5.5	2.00	3.25	4.5	4.75	5.0	3.00	10.00
		AE%	2.87	22.46	22.46	26.45	**6.75**	2.37	3.17	35.67	70.25	3.54	10.5	22.8	2.42	24.00	11.21
		Acc.%	6.14	5.98	6.31	7.03	**6.59**	6.39	5.35	6.73	3.28	6.14	6.79	5.9	7.24	6.24	6.00
IMDb-Adv	28	#Mis.	**39.0**	63.75	63.75	0.0	21.5	**37.0**	**37.75**	8.0	120.50	**37.75**	26.25	57.75	30.5	62.75	46.50
		#Clu.	3.0	4.5	4.25	0.0	2.0	**3.5**	4.75	0.75	3.25	3.25	3.0	3.0	2.75	**3.75**	7.25
		AE%	2.14	20.52	20.52	31.19	14.1	4.33	2.55	26.81	65.65	4.84	10.6	13.4	7.52	19.52	**6.31**
		Acc.%	7.49	8.23	8.3	9.67	8.72	9.24	10.41	8.14	5.98	-1.22	0.12	5.63	-2.04	2.89	**9.07**
IMDb-label	29	#Mis.	29.0	66.0	66.0	0.75	49.25	25.0	31.5	33.5	119.00	25.25	12.5	51.75	60.5	64.50	**45.75**
		#Clu.	2.25	**2.0**	2.25	0.0	**2.0**	2.75	**2.0**	1.75	**2.00**	**2.0**	1.25	2.75	5.0	**2.00**	**2.00**
		AE%	2.42	32.63	32.63	20.87	23.38	2.83	4.47	**6.93**	74.63	1.24	12.03	20.8	29.68	30.01	16.26
		Acc.%	5.29	3.6	3.97	**4.83**	5.4	5.41	5.69	5.35	4.15	5.65	6.24	4.45	3.61	4.19	3.78
IMDb-Customer	30	#Mis.	21.0	64.0	64.0	0.0	19.0	21.5	**23.5**	8.5	82.25	**24.5**	22.5	45.5	48.5	63.50	16.25
		#Clu.	4.5	2.0	3.0	0.0	2.5	**3.5**	**3.75**	5.5	2.00	3.25	4.5	4.75	5.0	3.00	10.00
		AE%	**3.06**	32.84	32.84	19.03	4.2	1.89	5.04	12.24	47.51	1.46	1.14	15.68	17.22	31.72	7.87
		Acc.%	-0.67	-0.84	**-0.4**	1.37	0.39	-0.57	1.25	0.48	-2.64	-4.86	1.96	-1.96	-6.31	-3.60	-0.92
Avg	-	#Mis.	21.63	59.14	58.60	23.92	23.84	21.79	21.43	23.93	105.11	23.08	19.29	41.90	25.15	52.80	17.93
		#Clu.	4.01	6.93	7.06	4.79	4.09	3.95	3.86	4.58	10.16	4.31	3.64	5.80	4.09	6.26	4.32
		AE%	1.87	35.87	35.51	15.46	8.49	2.19	2.18	7.58	73.07	5.08	3.29	17.15	7.07	30.87	6.55
		Acc.%	4.04	3.59	3.69	4.39	4.29	4.18	3.24	4.46	-0.28	2.35	4.44	3.50	1.46	2.85	4.01

leverages a hierarchical clustering to partition the set of median values of techniques into statistically distinct groups with a non-negligible difference. The NPSK does not require the assumptions of normal distributions, homogeneous distributions, and the minimum sample size. Different groups exhibit statistically significant differences at the predetermined significance level of 0.05 (α = 0.05). The NPSK ensures the magnitude of the difference between metrics within each group is not statistically significant, and the magnitude of the difference between metrics located in different groups is statistically significant. We use the same color for metrics within the same group and different colors for different groups. We utilize three distinct colors to underscore the statistical differences among 15 test selection metrics for each setting in terms of four evaluation criteria. Darker colors represent a superior performance. Specifically, the darkest blue denotes the first group, moderate blue indicates the second group, and the lightest blue represents the third group. If there are at least five groups, the results of the fourth group will be bolded while maintaining a white cell background. We use results collected from 4 different budgets for statistical test.

5.3.2 Key results. Tables 6, 7 show the results of 15 studied test selection metrics on 4 evaluation criteria across four benchmark datasets with five types of OOD scenarios. The average results are computed over all applicable settings. We perform within-task-type comparison for each objective.

Fault Detection. The effectiveness of a test suited to satisfy this testing objective is demonstrated by two criteria: number of mispredictions (#Mis) and number of clusters (#Clu.). For classification tasks (22 settings in total), the test suite selected by DSA detects the highest number of misclassifications and is located in the best statistical group in 21 out of 22 settings, with an average of 105.11 misclassifications detected. Gini and Ent appear in the best statistical group in the left case, detecting an average of 59.14 and 58.60 misclassifications. Meanwhile, DSA detects the highest number of clusters and is located in the best statistical group in 12 out of 22 settings (10.16 clusters on average), with DAT, Gini, Ent, and DR locating in the best statistical group in 7, 3, 3, and 1 settings, respectively. For the regression task (Udacity dataset, 8 settings in total), recall that we report the average number of mispredictions and clusters over 11 thresholds for criteria '#Mis.' and '#Clu.', respectively (Section 3.3.1). Test suites selected by NC are located in the best statistical group in '#Mis.' and '#Clu.' in 6 settings for each, detecting 66.09 mispredictions and 11.92 clusters on average.

> **Finding 3:** With fault detection as the testing objective, in general, we recommend using DSA for classification tasks and NC for regression tasks.

Performance Estimation. This objective is assessed by AE%. For classification tasks, Rand achieves the most accurate estimation with statistical significance in 6 out of 22 settings, followed by DAT located in the best statistical group in 4 settings. On average, Rand demonstrates the lowest error (1.97%), followed by STD and GD, which give AE% of 2.16% and 2.19%, averaging over all 22 classification settings. We note that metrics that cover different aspects of test suites can ensure the distribution similarity between the selected set and the whole testing set, which gives a more accurate estimation. However, in our studied settings, metrics specifically designed for performance estimation (CES, PACE, EST, and DR) cannot even outperform Rand, where EST demonstrates a large error of 22.68%. It is also worth noting that DSA gives the poorest performance, with the largest average error (73.07%). Similarly, uncertainty-based metrics, Gini and Ent also demonstrate poor performance. For regression tasks, LSA, Rand, and EST appear in the best statistical group in 3, 2, and 2 out of 8 settings, leading to an error value of 6.86%, 1.6%, and 1.95%, respectively. NC gives the worst performance, with an average AE% of 20.01%, followed by KMNC and CES, with error values of 13.36% and 10.03%, respectively.

Finding 4: Metrics (Rand, GD, STD) that encourage test diversity outperform metrics (CES, PACE, EST, and DR) that are specifically designed for performance estimation. EST (22.68%) and CES (10.03%) give a very inaccurate estimation in classification and regression tasks, respectively.

Retraining Guidance. Overall, retraining with test suites selected from datasets containing original inputs only (e.g., MNIST, MNIST-label) or mild distribution shifts (e.g., Udacity-C, IMDb-Customer) produces a small (e.g., ID 3, 18, 25) or even negative (e.g., ID 7, 8, 9, 11, 13, 20, 30) performance improvement. In these settings, all metrics achieve similar results because the model is already well-trained with the input distribution, and retraining on the selected set does not provide new information. In contrast, datasets containing adversarial OOD shifts (e.g., Udacity-Adv, MNIST-Adv) show significantly larger retraining gains (e.g., ID 15, 19, 28). For classification tasks, KMNC (4.37%) gives the highest average improvement, followed by GD (4.31%) and PACE (4.15%). They are located in the best statistical group in 2, 1, and 8 settings, respectively. DSA (-0.28%) gives the poorest performance. Metrics (MCP, DAT) that are designed solely for this objective achieves an average improvement of 2.85% and 4.01%, respectively. For regression tasks, PACE (5.25%) gives the largest average improvement, followed by STD (5.24%).

Finding 5: KMNC and PACE demonstrate the highest retraining improvement for classification and regression, respectively, outperforming metrics (MCP, DAT) specifically designed for this objective.

OOD sensitivity analysis. We analyze how the effectiveness of test selection metrics varies across different types of OOD shifts (corrupted, adversarial, label, temporal, and natural shifts). Our results demonstrate that, with a few exceptions, the relative performance rankings of test selection metrics remain largely stable across different OOD scenarios. For example, DSA and NC stably reaches top performance in fault detection for classification and regression tasks across different OOD types, respectively. However, we note that CES (1.06%) gives the best performance estimation in label shifts for classification tasks (3 settings), followed by Rand (1.59%). For the regression task (1 setting), EST performs the best in adversarial (1.08%) and corrupted (1.27%) shifts, followed by STD (2.5%) and Rand (2.04%), respectively. In general, metrics (Rand, GD, STD) that encourage test diversity still achieve outstanding performance in performance estimation. Overall, the effectiveness of test selection metrics is not highly sensitive to the type of OOD shifts, and the choice of metric can be made without needing to re-adjust for different OOD scenarios.

Finding 6: Overall, there is no clear fluctuations in metrics' performance under different OOD scenarios.

5.4 RQ3. Performance of test selection methods under different selection budgets.

Figure 3 shows the ranking of test selection metrics based on their average performance under different budgets for four criteria (#Mis., #Clu., AE%, Acc.%). The averages are calculated across all applicable experimental settings (e.g., 22 settings for classification-only metrics). A rank of '1' indicates the best performance. For instance, DSA's rank of '1' under #Mis. at budget=50 means it selected the test suite that detects the highest number of mispredictions on average. This visualization clearly shows how each metric's performance changes with the budget. Under **fault detection** (#Mis. and #Clu.), most metrics' rankings are stable. For example, DSA consistently ranks first, followed by Gini and Ent. However, the ranking of KMNC decreases as the budget increases, likely because larger suites cover more neurons but not necessarily more faults. For

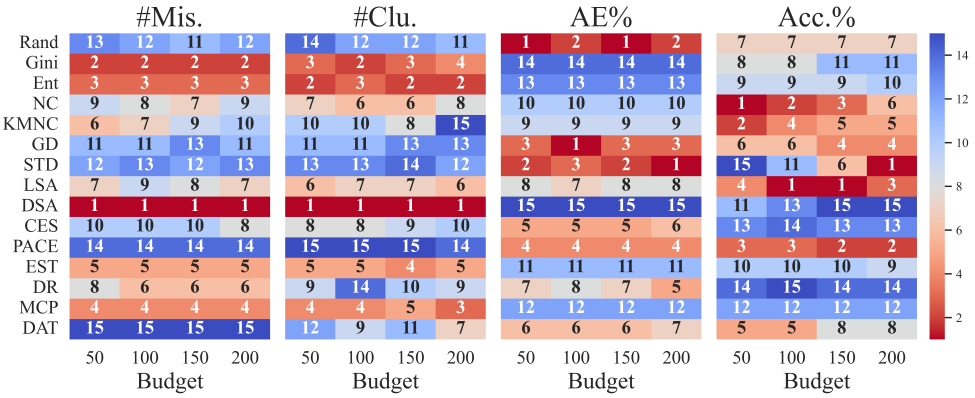


Fig. 3. The rankings of each test selection metric across different budgets for each studied criterion (#Mis., #Clu., AE, Acc.%). A rank of '1' indicates the best performance.

performance estimation (AE%), Rand consistently ranks first or second, while DSA consistently ranks last. Under **accuracy improvement** (Acc.%), rankings are more volatile. STD's ranking improves from last to first as the budget increases, with its average accuracy improvement rising from 0.89% to 4.76%. A similar trend is observed in GD, which also selects the suite based on input diversity computed from features extracted in a black-box manner. Conversely, uncertainty-based (Gini, Ent) and coverage-based (NC, KMNC) metrics show a consistent decline in ranking.

> **Finding 7:** Metric performance for fault detection and performance estimation is largely stable across budgets, while performance for accuracy improvement shows considerable fluctuation.

5.5 RQ4. Time efficiency of test selection metrics.

We evaluate the time efficiency of different test selection metrics by examining the total time costs (in seconds) of each metric when sampling 200 inputs (budget=200) from all settings for MNIST, Udacity, AndroZoo, and IMDb datasets. For example, for MNIST, we record the aggregated time costs of sampling 200 images from all 7 settings (i.e., ID 1, 2, 3, 14, 15, 16, 17). In this way, we can have a generalized overview of metrics' time efficiency under different settings.

Table 8. The time costs (in seconds) of all 15 metrics when sampling 200 inputs from all settings under each dataset. Best values are highlighted.

Dataset (settings)	Rand	Gini	Ent	NC	KMNC	GD	STD	LSA	DSA	CES	PACE	EST	DR	MCP	DAT
MNIST (7)	0.15	24.71	24.37	15.75	2561.05	101.38	98.82	447.92	974.57	288.33	19.38	1130.49	38.89	8.81	394.73
Udacity (8)	0.05	-	-	55.41	1960.33	325.62	320.57	83.07	-	760.74	32.11	137.23	82.72	-	-
AndroZoo (7)	0.04	54.08	55.24	40.93	9574.37	59.50	54.62	219.13	311.30	743.47	43.80	915.01	95.47	22.62	272.73
IMDb (8)	0.00	76.25	75.71	62.69	938.56	88.07	64.65	206.32	504.88	120.52	60.16	1492.51	266.59	35.71	430.56
Average	0.06	51.68	51.77	43.69	3758.58	143.64	134.67	239.11	596.92	478.26	38.87	918.81	120.92	22.38	366.01

Table 8 displays the time cost results. Rand (0.06s) is the fastest metric across all datasets. It is followed by clustering-based metrics (MCP: 22.38s, PACE: 38.87s) and uncertainty-based metrics (Gini: 51.68s, Ent: 51.77s). KMNC incurs the highest time cost on MNIST, Udacity, and Androzoo, whereas EST is the slowest on IMDb. The high time cost of KMNC arises from its iterative input selection process designed to maximize neuron coverage. EST's time cost varies because it relies

on different auxiliary metrics (LSA for Udacity; DSA and confidence scores for others). This also explains the time difference between regression and classification datasets. Surprise-based metrics (LSA, DSA) are costly as they must compute and store activation traces for the entire training set to gauge 'surprise'. For **fault detection**, a clear performance-speed trade-off exists: DSA is most effective (105.11 and 10.16 in #Mis. and #Clu., respectively) in classification tasks, while uncertainty metrics that rank second (Gini, Ent) are ten times faster but only half as effective, making them only suitable for time-critical applications. For **performance estimation**, the best-performing metrics (Rand, GD, STD) are also among the most time-efficient. For **retraining guidance**, while KMNC (4.37%) and PACE (5.25%) achieve the largest improvements for classification and regression, respectively, KMNC's high computational cost is a significant drawback. We recommend PACE for general use, as it offers a comparable 4.15% improvement (a difference of only 0.22%) for classification with far greater efficiency.

> **Finding 8:** A clear performance-efficiency trade-off exists for best-performing metrics in fault detection. However, for performance estimation and retraining guidance, the most effective metrics are also highly efficient, offering no such trade-off.

6 Discussion

6.1 Discussion of Findings

We discuss our findings from three perspectives: practical implications for DNN testing, explanations of findings, and their relation to prior work.

Practical Implications. Our results offer actionable guidance for selecting test metrics, retraining strategies, and fault identification pipelines under different objectives, data types, and budgets. (1) Testers should always prioritize inputs with high model surprise when aiming to detect faults in DL-enabled classifiers, irrespective of whether the test selection budget is tight or loose. In contrast, for DL-based regressors, test suites should be constructed by incorporating inputs that achieve broader neuron coverage. (2) Testers are encouraged to prioritize diverse inputs to achieve an accurate performance estimation, irrespective of whether the selection budget is tight or loose. (3) When the test selection budget is loose, select diverse inputs to retrain the model. This is based on the drastic improvement of STD's ranking as the budget increases, discovered in RQ3. (4) Type I retraining is recommended for sparse data (e.g., AndroZoo), whereas Type II is preferable for dense visual and textual data. (5) For efficient fault identification and targeted model repair, we recommend using (DeepDrebin, UMAP, DBSCAN) for malware data, (RoBERTa, UMAP, DBSCAN) for textual data, and (ResNet-50, UMAP, DBSCAN) for image data to conduct cluster-specific retraining for fault-targeted repair.

Rationale Explanations of Findings. We provide rationales behind the main findings in each of the three testing objectives. Specifically, DSA is effective in fault detection since it captures boundary closeness by comparing activation distances to same-class versus different-class neighbors, where input close to the decision boundary is more likely to be misclassified [37]. Moreover, EST shows a very inaccurate estimation in the classification problem. This may be due to its sampling-based strategy with DSA-based auxiliary variable, which over-selects mispredicted inputs, producing a suite that is not representative of the full test distribution. Diversity-based metrics better approximate the underlying data distribution, explaining their strong performance in estimation. Diversity-based metrics become increasingly effective for retraining guidance as the budget increases, as they progressively construct a more representative subset of the feature space.

Alignment with Prior Findings. While some of our findings are consistent with prior work, they account for only a small portion of our results. For example, our observation that random

selection can outperform specialized metrics for retraining under OOD settings aligns with [31]. Similarly, Devlin et al. [18] report that uncertainty-based metrics outperform coverage-based metrics for fault detection in image classification, which we replicate and extend this finding across additional data modalities and OOD shifts. However, existing work *lacks a unified evaluation of metrics designed for different objectives*. Our study addresses this gap and reveals that some metrics can outperform others even when they are not explicitly designed for the target objective.

6.2 Threats to Validity

The validity typology provides a system for classifying and improving inferences related to three validity types: internal validity, external validity, and construct validity [4].

Internal Validity concerns factors that could influence our results and the causal relationships we draw. To mitigate the threat in three aspects: (1) the implementation quality of the 15 test selection metrics, we mitigate the threats by closely following the original papers and using publicly available code. (2) hyperparameter manipulation, we mitigate this by using optimal values identified in the original papers for existing metrics (e.g., NC, KMNC), and for techniques used in the clustering pipeline, we employed established methods to determine the best hyperparameters. We provide details in our online repository for reproducibility. (3) Randomness with test input selection. We mitigate this by running experiments three times for metrics with randomness (EST, Rand, DAT, KMNC, NC, GD, STD, PACE, CES) and report the average value. We run only once for deterministic metrics (Gini, Ent, LSA, DSA, DR, MCP).

External Validity concerns the generalizability of our findings beyond our experimental settings, which we address through three aspects: dataset and model selection, OOD scenarios, and selection budgets. To mitigate them, (1) we use four datasets with three modalities and 13 models across two tasks. (2) We construct five types of OOD scenarios (i.e., corruption, adversarial, temporal, natural, and label shifts). Specifically, adversarial and corrupted sets are constructed with multiple attack methods and severity levels to simulate a generalized real-world testing environment. We also select commonly used datasets from the literature to simulate natural covariate shifts and temporal shifts [33, 44]. For label shifts, we randomly chose the label ratio, as the system under test may encounter arbitrary label distributions in reality. To further reduce the threat, we explore alternative label distributions with statistical analysis. We take MNIST with LeNet-5 and IMDb with Transformer as experiment subjects. Specifically, for MNIST, we simulate label shifts by sampling an opposite skewed distribution with 34% for digit 9, 15% each for digits 6-8, 5% each for digits 3-5, 2% each for digits 0-2. For IMDb, we sample the reviews with 20% positive and 80% negative. The results are shown in Table 9, where the results are averaged over four budgets (50, 100, 150, 200). For MNIST (**MNIST-label2**), surprise-based and uncertainty-based metrics (DSA, Gini, Ent) achieve the best fault detection, yielding the highest #Mis. and #Clu. results. For performance estimation, EST, despite being designed for this objective, performs poorly, while diversity-encouraging metrics generally provide better estimates. Metrics tailored for retraining guidance (e.g., MCP, DAT) can be outperformed by others (e.g., STD for MNIST, Rand for IMDb). For IMDb (**IMDb-label2**), DSA again achieves the highest #Mis., followed by Gini and Ent, which also lead in #Clu. Diversity-based metrics achieve the top performance estimation, whereas EST remains ineffective. Overall, these results are consistent with our main findings, indicating that our conclusions remain robust under varied label distributions encountered in deployment.

(3) Finally, we evaluate budgets from 50 to 200 and observe that metric rankings are largely insensitive within this range. Since these budgets are relatively small, we further assess all 15 metrics under larger budgets using the IMDb–Transformer setting (ID 13), with selection sizes from 500 to 1,000, with an interval of 100. As shown in Table 9 (**IMDb-LB**), the conclusions remain consistent: DSA and DAT achieve the best #Mis. and #Clu., respectively, diversity-based metrics

Table 9. Performance of 15 metrics under alternative label distributions (MNIST-label2 and IMDb-label2) and large selection budgets (IMDb-LB).

	Eval.	Rand	Gini	Ent	NC	KMNC	GD	STD	LSA	DSA	CES	PACE	EST	DR	MCP	DAT
MNIST-label2	#Mis.	2.25	59.0	60.75	**5.25**	4.0	3.0	2.25	1.0	62.25	2.5	3.0	14.5	0.75	41.25	2.5
	#Clu.	0.0	10.75	9.25	1.0	0.5	0.5	0.25	0.0	13.25	0.5	0.0	**3.0**	0.0	10.75	0.25
	AE%	**1.28**	45.85	47.48	2.81	2.36	1.2	1.13	0.8	51.85	**1.42**	0.56	8.85	1.15	33.64	0.53
	Acc.%	0.93	0.61	0.34	0.87	**1.05**	1.18	1.3	**1.07**	0.78	0.31	1.21	0.67	0.07	0.6	1.21
IMDb-label2	#Mis.	12.0	61.25	61.25	0.5	**35.75**	11.25	12.75	3.75	120.25	11.0	9.75	39.75	57.25	58.5	18.75
	#Clu.	**2.25**	3.25	**2.25**	0.0	4.0	1.5	**2.0**	0.75	3.25	2.75	**2.0**	0.5	3.5	2.75	3.25
	AE%	**3.96**	39.91	39.91	9.25	22.66	2.71	1.98	7.25	88.0	1.83	2.13	20.71	40.58	36.79	**4.54**
	Acc.%	1.28	-0.02	-1.27	**0.66**	0.1	0.11	-0.49	-3.68	-3.07	0.32	0.9	0.82	-1.61	0.22	0.1
IMDb-LB	#Mis.	109.33	356.17	356.17	2.0	96.33	114.33	117.0	53.5	610.17	113.83	91.17	284.5	148.17	358.67	**193.33**
	#Clu.	**2.83**	2.33	2.0	0.0	3.83	2.5	3.0	3.67	**2.67**	**2.83**	3.5	2.17	2.33	2.17	16.67
	AE%	1.29	32.11	32.11	15.23	**2.77**	0.44	0.68	8.41	66.37	1.26	3.23	22.28	4.57	32.48	10.36
	Acc.%	4.5	3.42	**3.63**	-0.1	-0.18	4.9	1.43	-1.03	-3.43	0.37	2.9	2.23	-0.53	0.89	3.9

(e.g., GD, STD) provide accurate performance estimation, and MCP and DAT do not yield top retraining performance.

Construct Validity concerns whether the study correctly identifies the operational measures for the concepts being investigated [83]. In our context, this means the selected evaluation criteria must reliably reflect the intended testing objectives. For fault detection, we employ the number of mispredictions (#Mis.) and the number of clusters (#Clu.) to evaluate the effectiveness. These two criteria are widely used in the literature [1, 7, 20, 79]. Specifically, #Clu. alleviates the potential bias introduced by #Mis. when mispredictions are attributed to the same fault. For retraining guidance and performance estimation, we assess the accuracy improvement (Acc.%) and absolute error (AE%), which are well-established and straightforward measurements [12, 25, 26, 33, 37, 79, 82]. Meanwhile, we also identify a more effective retraining process to accurately capture the improvement margin.

7 Conclusion

Our study presents an extensive study of 15 existing test selection metrics on benchmarking their performance on three testing objectives: fault detection, performance estimation, and retraining guidance. Through empirical analysis on five OOD types, four datasets with three modalities, and 13 DNNs, encompassing 1,640 unique experimental scenarios, we have derived critical insights for practitioners. For example, Metrics that encourage input diversity (Rand, GD, STD) are effective in performance estimation, outperforming those specifically designed for this objective. Meanwhile, metrics' performance on fault detection and performance estimation is largely stable across budgets, while there is considerable fluctuation for retraining guidance.

8 Data Availability

Our implementation is publicly available at https://github.com/MetricsBenchmark/TestingBenchmark.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant 62502550, the Research Grants Council of Hong Kong (9229029), funds from CityU HK (9229192, 6000871), Shenzhen Science and Technology Program (KJZD20240903095700001), JST CRONOS Grant (No. JPMJCS24K8), and JSPS KAKENHI Grant (No.JP21H04877, No.JP23H03372, and No.JP24K02920), Canada CIFAR AI Chairs Program, the Natural Sciences and Engineering Research Council of Canada.

References

[1] Zohreh Aghababaeyan, Manel Abdellatif, Lionel Briand, S Ramesh, and Mojtaba Bagherzadeh. 2023. Black-box testing of deep neural networks through test case diversity. *IEEE Transactions on Software Engineering* 49, 5 (2023), 3182–3204. doi:10.1109/TSE.2023.3243522

[2] Zohreh Aghababaeyan, Manel Abdellatif, Mahboubeh Dadkhah, and Lionel Briand. 2024. Deepgd: A multi-objective black-box test selection approach for deep neural networks. *ACM Transactions on Software Engineering and Methodology* 33, 6 (2024), 1–29. doi:10.1145/3644388

[3] Kevin Allix, Tegawendé F Bissyandé, Jacques Klein, and Yves Le Traon. 2016. Androzoo: Collecting millions of Android apps for the research community. In *Proceedings of the 13th international conference on mining software repositories*. 468–471. doi:10.1145/2901739.2903508

[4] Kylie Anglin, Qing Liu, and Vivian C Wong. 2024. A primer on the validity typology and threats to validity in education research. *Asia Pacific Education Review* 25, 3 (2024), 557–574. doi:10.1007/s12564-024-09955-4

[5] Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, Konrad Rieck, and CERT Siemens. 2014. Drebin: Effective and explainable detection of android malware in your pocket.. In *Ndss*, Vol. 14. 23–26. doi:10.14722/ndss.2014.23247

[6] Mohammed Attaoui, Hazem Fahmy, Fabrizio Pastore, and Lionel Briand. 2023. Black-box safety analysis and retraining of DNNs based on feature extraction and clustering. *ACM Transactions on Software Engineering and Methodology* 32, 3 (2023), 1–40. doi:10.1145/3550271

[7] Mohammed Oualid Attaoui, Hazem Fahmy, Fabrizio Pastore, and Lionel Briand. 2024. Supporting safety analysis of image-processing dnns through clustering-based approaches. *ACM Transactions on Software Engineering and Methodology* 33, 5 (2024), 1–48. doi:10.1145/3643671

[8] David Berend, Xiaofei Xie, Lei Ma, Lingjun Zhou, Yang Liu, Chi Xu, and Jianjun Zhao. 2020. Cats are not fish: Deep learning testing calls for out-of-distribution awareness. In *Proceedings of the 35th IEEE/ACM international conference on automated software engineering*. 1041–1052. doi:10.1145/3324884.3416609

[9] Ella Bingham and Heikki Mannila. 2001. Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. 245–250. doi:10.1145/502512.502546

[10] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* (2016). https://arxiv.org/abs/1604.07316

[11] Taejoon Byun, Vaibhav Sharma, Abhishek Vijayakumar, Sanjai Rayadurgam, and Darren Cofer. 2019. Input prioritization for testing neural networks. In *2019 IEEE International Conference On Artificial Intelligence Testing (AITest)*. IEEE, 63–70. doi:10.1109/AITest.2019.000-6

[12] Junjie Chen, Zhuo Wu, Zan Wang, Hanmo You, Lingming Zhang, and Ming Yan. 2020. Practical accuracy estimation for efficient deep neural network testing. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 29, 4 (2020), 1–35. doi:10.1145/3394112

[13] Lingjiao Chen, Matei Zaharia, and James Y Zou. 2022. Estimating and explaining model performance when both covariates and labels shift. *Advances in Neural Information Processing Systems* 35 (2022), 11467–11479. https://dl.acm.org/doi/10.5555/3600270.3601103

[14] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555* (2020). https://arxiv.org/abs/2003.10555

[15] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. 2017. EMNIST: Extending MNIST to handwritten letters. *2017 International Joint Conference on Neural Networks (IJCNN)* (2017). doi:10.1109/ijcnn.2017.7966217

[16] Demet Demir, Aysu Betin Can, and Elif Surer. 2024. Test selection for deep neural networks using meta-models with uncertainty metrics. In *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*. 678–690. doi:10.1145/3650212.3680312

[17] K. P. Devakumar. 2020. IMDB Review Classification LSTM, GRU, CNN, GloVe. *https://www.kaggle.com/code/imdevskp/imdb-review-classification-lstm-gru-cnn-glove/*.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 4171–4186. doi:10.18653/V1/N19-1423

[19] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, Vol. 96. 226–231. https://dl.acm.org/doi/10.5555/3001460.3001507

[20] Yang Feng, Qingkai Shi, Xinyu Gao, Jun Wan, Chunrong Fang, and Zhenyu Chen. 2020. Deepgini: prioritizing massive tests to enhance the robustness of deep neural networks. In *Proceedings of the 29th ACM SIGSOFT international symposium on software testing and analysis*. 177–188. doi:10.1145/3395363.3397357

[21] Xinyu Gao, Yang Feng, Yining Yin, Zixi Liu, Zhenyu Chen, and Baowen Xu. 2022. Adaptive test selection for deep neural networks. In *Proceedings of the 44th international conference on software engineering*. 73–85. doi:10.1145/3510003.3510232

[22] Jacob Gildenblat. 2016. Visualizations for understanding the regressed wheel steering angle for self-driving cars. *https://github.com/jacobgil/keras-steering-angle-visualizations*.

[23] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014). https://doi.org/10.48550/arXiv.1412.6572

[24] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. 2017. Adversarial examples for malware detection. In *Computer Security–ESORICS 2017: 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II 22*. Springer, 62–79. doi:10.1007/978-3-319-66399-9_4

[25] Antonio Guerriero, Roberto Pietrantuono, and Stefano Russo. 2021. Operation is the hardest teacher: estimating DNN accuracy looking for mispredictions. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 348–358. doi:10.1109/ICSE43902.2021.00042

[26] Antonio Guerriero, Roberto Pietrantuono, and Stefano Russo. 2024. DeepSample: DNN sampling-based testing for operational accuracy assessment. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 1–12. doi:10.1109/ICSE43902.2021.00042

[27] Chris Gundling. 2016. Steering angle model: Cg32. *https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/cg23*.

[28] Yao Hao, Zhiqiu Huang, Hongjing Guo, and Guohua Shen. 2023. Test input selection for deep neural network enhancement based on multiple-objective optimization. In *2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 534–545. doi:10.1109/SANER56733.2023.00056

[29] Fabrice Harel-Canada, Lingxiao Wang, Muhammad Ali Gulzar, Quanquan Gu, and Miryung Kim. 2020. Is neuron coverage a meaningful measure for testing deep neural networks?. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 851–862. doi:10.1145/3368089.3409754

[30] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778. doi:10.1109/CVPR.2016.90

[31] Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136* (2016). doi:10.48550/arXiv.1610.02136

[32] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 168–177. doi:10.1145/1014052.1014073

[33] Qiang Hu, Yuejun Guo, Maxime Cordy, Xiaofei Xie, Lei Ma, Mike Papadakis, and Yves Le Traon. 2022. An empirical study on data distribution-aware test selection for deep learning enhancement. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 31, 4 (2022), 1–30. doi:10.1145/3511598

[34] Qiang Hu, Yuejun Guo, Xiaofei Xie, Maxime Cordy, Lei Ma, Mike Papadakis, and Yves Le Traon. 2024. Test optimization in dnn testing: A survey. *ACM Transactions on Software Engineering and Methodology* 33, 4 (2024), 1–42. doi:10.1145/3643678

[35] Qiang Hu, Yuejun Guo, Xiaofei Xie, Maxime Cordy, Wei Ma, Mike Papadakis, Lei Ma, and Yves Le Traon. 2025. Assessing the Robustness of Test Selection Methods for Deep Neural Networks. *ACM Transactions on Software Engineering and Methodology* (2025). doi:10.1145/3715693

[36] Shachar Kaufman, Saharon Rosset, Claudia Perlich, and Ori Stitelman. 2012. Leakage in data mining: Formulation, detection, and avoidance. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 6, 4 (2012), 1–21. doi:10.1145/2382577.2382579

[37] Jinhan Kim, Robert Feldt, and Shin Yoo. 2019. Guiding deep learning system testing using surprise adequacy. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 1039–1049. doi:10.1109/ICSE.2019.00108

[38] 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*. 1466–1476. doi:10.1145/3368089.3417065

[39] Ronald S King. 2013. Cluster analysis and data mining: An introduction. (2013). https://dl.acm.org/doi/abs/10.5555/2823861

[40] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. 2018. Adversarial examples in the physical world. In *Artificial intelligence safety and security*. Chapman and Hall/CRC, 99–112. doi:10.1201/9781351251389-8

[41] Yann LeCun. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/* (1998). doi:10.24432/C53K8Q

[42] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324. doi:10.1109/5.726791

[43] Deqiang Li and Qianmu Li. 2020. Adversarial deep ensemble: Evasion attacks and defenses for malware detection. *IEEE Transactions on Information Forensics and Security* 15 (2020), 3886–3900. doi:10.1109/TIFS.2020.3003571

[44] Deqiang Li, Tian Qiu, Shuo Chen, Qianmu Li, and Shouhuai Xu. 2021. Can we leverage predictive uncertainty to detect dataset shift and adversarial examples in android malware detection?. In *Proceedings of the 37th Annual Computer Security Applications Conference*. 596–608. doi:10.1145/3485832.3485916

[45] Yinghua Li, Xueqi Dang, Lei Ma, Jacques Klein, and Tegawendé F Bissyandé. 2024. Prioritizing test cases for deep learning-based video classifiers. *Empirical Software Engineering* 29, 5 (2024), 111. doi:10.1007/s10664-024-10520-1

[46] Yinghua Li, Xueqi Dang, Lei Ma, Jacques Klein, Yves Le Traon, and Tegawende F Bissyande. 2024. Test input prioritization for 3d point clouds. *ACM Transactions on Software Engineering and Methodology* 33, 5 (2024), 1–44. doi:10.1145/3643676

[47] Zenan Li, Xiaoxing Ma, Chang Xu, Chun Cao, Jingwei Xu, and Jian Lü. 2019. Boosting operational dnn testing efficiency through conditioning. In *Proceedings of the 2019 27th ACM Joint meeting on European software engineering conference and symposium on the foundations of software engineering*. 499–509. doi:10.1145/3338906.3338930

[48] Jingjing Liang, Sebastian Elbaum, and Gregg Rothermel. 2018. Redefining prioritization: continuous prioritization for continuous integration. In *Proceedings of the 40th International Conference on Software Engineering*. 688–698. doi:10.1145/3180155.3180213

[49] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019). https://doi.org/10.48550/arXiv.1907.11692

[50] Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, et al. 2018. Deepgauge: Multi-granularity testing criteria for deep learning systems. In *Proceedings of the 33rd ACM/IEEE international conference on automated software engineering*. 120–131. doi:10.1145/3238147.3238202

[51] Wei Ma, Mike Papadakis, Anestis Tsakmalis, Maxime Cordy, and Yves Le Traon. 2021. Test selection for deep learning systems. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 30, 2 (2021), 1–22. doi:10.1145/3417330

[52] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*. 142–150. https://dl.acm.org/doi/10.5555/2002472.2002491

[53] James MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, Vol. 5. University of California press, 281–298. https://api.semanticscholar.org/CorpusID:6278891

[54] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017). https://doi.org/10.48550/arXiv.1706.06083

[55] Leland McInnes, John Healy, Steve Astels, et al. 2017. hdbscan: Hierarchical density based clustering. *J. Open Source Softw.* 2, 11 (2017), 205. doi:10.21105/joss.00205

[56] Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018). doi:10.21105/joss.00861

[57] Niall McLaughlin, Jesus Martinez del Rincon, BooJoong Kang, Suleiman Yerima, Paul Miller, Sakir Sezer, Yeganeh Safaei, Erik Trickel, Ziming Zhao, Adam Doupé, et al. 2017. Deep android malware detection. In *Proceedings of the seventh ACM on conference on data and application security and privacy*. 301–308. doi:10.1145/3029806.3029823

[58] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S. Ecker, Matthias Bethge, and Wieland Brendel. 2019. Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming. *arXiv preprint arXiv:1907.07484* (2019). https://doi.org/10.48550/arXiv.1907.07484

[59] Jose G Moreno-Torres, Troy Raeder, Rocío Alaiz-Rodríguez, Nitesh V Chawla, and Francisco Herrera. 2012. A unifying view on dataset shift in classification. *Pattern recognition* 45, 1 (2012), 521–530. doi:10.1016/j.patcog.2011.06.019

[60] Vasilii Mosin, Miroslaw Staron, Darko Durisic, Francisco Gomes de Oliveira Neto, Sushant Kumar Pandey, and Ashok Chaitanya Koppisetty. 2022. Comparing input prioritization techniques for testing deep learning algorithms. In *2022 48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 76–83. doi:10.1109/SEAA56994.2022.00020

[61] Davoud Moulavi, Pablo A Jaskowiak, Ricardo JGB Campello, Arthur Zimek, and Jörg Sander. 2014. Density-based clustering validation. In *Proceedings of the 2014 SIAM international conference on data mining*. SIAM, 839–847. doi:10.1137/1.9781611973440.96

[62] Norman Mu and Justin Gilmer. 2019. Mnist-c: A robustness benchmark for computer vision. *arXiv preprint arXiv:1906.02337* (2019). https://doi.org/10.48550/arXiv.1906.02337

[63] Apoorv Nandan. 2020. Text classification with Transformer. *https://keras.io/examples/nlp/text_classification_with_transformer/*.

[64] Karl Pearson. 1901. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science* 2, 11 (1901), 559–572. doi:10.1080/14786440109462720
[65] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2017. Deepxplore: Automated whitebox testing of deep learning systems. In *proceedings of the 26th Symposium on Operating Systems Principles*. 1–18. doi:10.1145/3361566
[66] Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th annual meeting of the association for computational linguistics*. 1085–1097. https://api.semanticscholar.org/CorpusID:196202909
[67] Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20 (1987), 53–65. doi:10.1016/0377-0427(87)90125-7
[68] Alexandru Constantin Serban, Erik Poll, and Joost Visser. 2018. Adversarial examples-a complete characterisation of the phenomenon. *arXiv preprint arXiv:1810.01185* (2018). doi:10.48550/arXiv.1810.01185
[69] Weijun Shen, Yanhui Li, Lin Chen, Yuanlei Han, Yuming Zhou, and Baowen Xu. 2020. Multiple-boundary clustering and prioritization to promote neural network retraining. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*. 410–422. doi:10.1145/3324884.3416621
[70] Ying Shi, Beibei Yin, Zheng Zheng, and Tiancheng Li. 2021. An empirical study on test case prioritization metrics for deep neural networks. In *2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*. IEEE, 157–166. doi:10.1109/QRS54544.2021.00027
[71] Alex Staravoitau. 2016. Behavioral cloning: end-to-end learning for self-driving cars. *https://github.com/navoshta/behavioral-cloning*.
[72] Sully-Chen. 2016. Autopilot-Tensorflow. *https://github.com/SullyChen/Autopilot-TensorFlow*.
[73] Weifeng Sun, Meng Yan, Zhongxin Liu, and David Lo. 2023. Robust test selection for deep neural networks. *IEEE Transactions on Software Engineering* 49, 12 (2023), 5250–5278. doi:10.1109/TSE.2023.3330982
[74] Chakkrit Tantithamthavorn, Shane McIntosh, Ahmed E Hassan, and Kenichi Matsumoto. 2016. An empirical comparison of model validation techniques for defect prediction models. *IEEE Transactions on Software Engineering* 43, 1 (2016), 1–18. doi:10.1109/TSE.2016.2584050
[75] TextAttack. [n. d.]. Transforms an input by replacing any word with 'banana'. *https://textattack.readthedocs.io/en/latest/*.
[76] Udacity. 2016. Using Deep Learning to Predict Steering Angles. https://medium.com/udacity/challenge-2-using-deep-learning-to-predict-steering-angles-f42004a36ff3
[77] Dan Wang and Yi Shang. 2014. A new active labeling method for deep learning. In *2014 International joint conference on neural networks (IJCNN)*. IEEE, 112–119. doi:10.1109/IJCNN.2014.6889457
[78] Zhiyu Wang, Sihan Xu, Lingling Fan, Xiangrui Cai, Linyu Li, and Zheli Liu. 2024. Can coverage criteria guide failure discovery for image classifiers? an empirical study. *ACM Transactions on Software Engineering and Methodology* 33, 7 (2024), 1–28. doi:10.1145/3672446
[79] Michael Weiss and Paolo Tonella. 2022. Simple techniques work surprisingly well for neural network test prioritization and active learning (replicability study). In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*. 139–150. doi:10.1145/3533767.3534375
[80] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? *Advances in neural information processing systems* 27 (2014). https://api.semanticscholar.org/CorpusID:362467
[81] Chunyu Zhao, Yanzhou Mu, Xiang Chen, Jingke Zhao, Xiaolin Ju, and Gan Wang. 2022. Can test input selection methods for deep neural network guarantee test diversity? A large-scale empirical study. *Information and Software Technology* 150 (2022), 106982. doi:10.1016/j.infsof.2022.106982
[82] Jianyi Zhou, Feng Li, Jinhao Dong, Hongyu Zhang, and Dan Hao. 2020. Cost-effective testing of a deep learning model through input reduction. In *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 289–300. doi:10.1109/ISSRE5003.2020.00035
[83] Xin Zhou, Yuqin Jin, He Zhang, Shanshan Li, and Xin Huang. 2016. A map of threats to validity of systematic literature reviews in software engineering. In *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 153–160. doi:10.1109/APSEC.2016.031

Received 2025-09-11; accepted 2025-12-22