Cluster-Based Multi-Objective Metamorphic Test Case Pair Selection for Deep Neural Networks

Jingling Wang
jinglingwang@smail.nju.edu.cn
State Key Laboratory for Novel Software Technology
Nanjing University, China

Shuwei Qiu
swchiu@smail.nju.edu.cn
State Key Laboratory for Novel Software Technology
Nanjing University, China

Peng Wang
pengwang@smail.nju.edu.cn
State Key Laboratory for Novel Software Technology
Nanjing University, China

Jiyuan Song
jiyuansong@smail.nju.edu.cn
State Key Laboratory for Novel Software Technology
Nanjing University, China

Huayao Wu*
hywu@nju.edu.cn
State Key Laboratory for Novel Software Technology
Nanjing University, China

Xintao Niu
niuxintao@nju.edu.cn
State Key Laboratory for Novel Software Technology
Nanjing University, China

Changhai Nie
changhainie@nju.edu.cn
State Key Laboratory for Novel Software Technology
Nanjing University, China

Abstract

Due to the rapid development of deep neural networks (DNNs), ensuring their quality has become increasingly important. However, the test oracle problem poses an obstacle to DNN testing because of the massive unlabeled data. Metamorphic Testing (MT) has proven effective in alleviating the test oracle problem, and many efforts have been made to improve the cost-effectiveness of MT for DNNs. Some approaches focus on selecting good metamorphic relations (MRs), while others target the selection of suspicious source test cases. Since follow-up test cases are generated by combining source test cases with MRs, selecting effective pairs of source test cases and MRs is also quite essential and beneficial for MT. In this paper, we propose CMPS, a multi-objective black-box approach for metamorphic test case pair selection. Considering both uncertainty and diversity, CMPS aims to select pairs that can detect more unique faults in the model. It evaluates uncertainty based on model outputs and assesses diversity through clustering source test cases. Furthermore, CMPS can adaptively optimize the selection process based on feedback from the execution results of the selected pairs. We conduct extensive experiments on three datasets and five DNN models to evaluate CMPS's performance. The experimental results demonstrate that CMPS significantly outperforms baseline approaches in both failure triggering and fault detection.

*Corresponding author

Conference acronym 'XX, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

CCS Concepts

• **Software and its engineering → Software testing and debugging**.

Keywords

Deep Neural Networks, Metamorphic Testing, Test Selection, Multi-Objective, Cluster-Based

ACM Reference Format:
Jingling Wang, Shuwei Qiu, Peng Wang, Jiyuan Song, Huayao Wu, Xintao Niu, and Changhai Nie. 2025. Cluster-Based Multi-Objective Metamorphic Test Case Pair Selection for Deep Neural Networks. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Software powered by deep neural networks (DNNs) has become an integral part of our daily lives, with applications spanning healthcare [36], transportation [57], education [23], and beyond. However, like traditional software, even minor errors in DNN-based software can lead to serious accidents or even pose significant risks to human safety [17, 46, 49]. Therefore, extensively testing various DNN-based software and effectively uncovering their defects is crucial for ensuring their quality.

To this end, testers need to collect a large number of test data for various testing scenarios and label them to establish test oracles [4], which is the mechanism used to verify the correctness of test outputs. Although establishing test oracles through manual labeling is extremely time-consuming and often requires the assistance of domain experts, it becomes impossible to determine whether the output is correct without labeling, which is known as the test oracle problem [52]. In some instances, labeling is even unfeasible, particularly when the correct test output is difficult to define. Therefore,

researchers have investigated some techniques to reduce reliance on manual annotation while effectively alleviating the test oracle problem.

Among these techniques, Metamorphic Testing (MT) has emerged as the most widely adopted, enabling testers to evaluate DNN-based software with unlabeled test cases. In 13 studies mitigating the test oracle problem, 12 of them utilized MT [35]. The core idea behind MT is to identify and utilize metamorphic relations (MRs), which are properties or transformations that describe how the output of a system should change when its input is altered in certain ways. In MT for DNNs, instead of directly comparing the output of the source test case to a known correct result (as in traditional testing), testers use some MRs to generate follow-up test cases and run them on the model under test to check if the model's output adheres to the expected transformations. If the outputs do not meet the MRs, it indicates a potential defect in the model.

However, implementing MT requires additional testing resources, as it entails executing follow-up test cases generated from source test cases and MRs [33]. Multiple executions of the model can consume significant computational resources. As the number of MRs increases, more follow-up test cases will be executed for each source test case, further increasing the overall testing cost. In order to enhance the cost-effectiveness of MT for DNNs, numerous efforts have been undertaken, which can be roughly divided into two categories: MR selection and test case selection. MR selection aims to identify a relatively small set of high-quality MRs to reduce the generation of follow-up test cases while preserving the effectiveness of MT. Chen et al. [10] argued that good MRs should ensure the outputs of the program are as diverse as possible. In contrast, test case selection focuses on choosing a subset of more suspicious test cases to generate follow-up test cases, thereby triggering more failures with reduced test execution costs.

These two categories of approaches both overlook the interaction between source test cases and MRs, which also plays a crucial role in the effectiveness of MT, as follow-up test cases are generated through their combinations. Consequently, researchers have investigated and developed several metamorphic test case pair selection approaches, which focus on selecting good pairs of source test cases and MRs (each pair consists of a source test case and an MR) to generate more challenging follow-up test cases, thereby enhancing the performance of MT for DNNs. Xie et al. [55] proposed a white-box approach to prioritize pairs of source and follow-up test cases (each pair consists of a source test case and one corresponding follow-up test case) for subsequent selection. This approach essentially belongs to metamorphic test case pair selection as when a follow-up test case is chosen, the corresponding MR is determined as well. It evaluates and ranks each pair based on the outputs of the neurons at a certain intermediate layer of the DNN model. However, accessing the internal structures of DNN models is often not feasible in many real-world scenarios, limiting the practicality of this approach. In contrast, Arrieta et al. [2] proposed a black-box approach utilizing the Non-dominated Sorting Genetic Algorithm (NSGA-II) [11], a widely-used genetic algorithm for multi-objective optimization [1, 16], to select follow-up test cases for different MRs. Despite its black-box advantages, this approach can only select source test cases for a single MR per execution, limiting its applicability in realistic testing scenarios where multiple MRs need to be addressed concurrently.

To address the aforementioned challenges and enhance the performance of MT for DNNs, we propose a novel black-box approach called **CMPS** (**C**luster-based **M**ulti-objective **P**air **S**election) for metamorphic test case pair selection. As a multi-objective approach, CMPS considers both uncertainty and diversity, aiming to select pairs that can detect a greater number of unique faults in the model with a given test budget. Meanwhile, CMPS is capable of handling multiple MRs simultaneously, enabling it to prioritize those with higher effectiveness during selection, especially when the fault detection abilities of different MRs varies significantly. Specifically, CMPS first evaluates the uncertainty of the DNN model with respect to the source test cases, prioritizing those with high uncertainty. The underlying insight is that the greater uncertainty the model has about a source test case, the more likely the source test case is to trigger incorrect behavior of the model. Next, CPMS iteratively selects high-uncertainty source test cases from different clusters to promote diversity in the selection. For each selected source test case, CMPS chooses the best MR based on the Euclidean distance between the source and follow-up test cases. Moreover, CMPS leverages feedback information from execution results of the selected pairs to adaptively optimize the selection process, further enhancing its effectiveness.

To evaluate the effectiveness of CMPS, we conduct extensive experiments on six subjects, covering three datasets and five DNN models, comparing CMPS with the existing black-box selection approach proposed by Arrieta et al. [2] and Random Selection (RS). The experimental results demonstrate that CMPS significantly outperforms baseline approaches in triggering more failures and detecting more diverse faults.

The main contributions of this work can be summarized as follows:

- We propose a novel black-box metamorphic test case pair selection approach called CMPS for DNNs. As a multi-objective approach, CMPS is the first to incorporate both uncertainty and diversity as optimization goals in metamorphic test case pair selection. Furthermore, CMPS leverages the feedback information from execution results of the selected pairs to adaptively optimize the selection process.
- As a black-box approach, CMPS can simultaneously handle multiple MRs in one execution, addressing the limitations of existing black-box approaches, which can only process one MR per execution.
- We perform an extensive experiment to evaluate the capability of CMPS with three datasets and five DNN models. The results show that our approach is significantly superior to the baseline approaches in terms of triggering failures and detecting diverse faults, indicating that CMPS can effectively enhance the performance of MT for DNNs.

The rest of the paper is structured as follows: Section 2 explains basic background related to DNNs and MT. Section 3 explains the details of our approach CMPS. Section 4 introduces the design details of our experiments. Sections 5 describes the results of the experiments and the analysis on them. Section 6 discusses some

threats to the validity of our work. Section 7 presents some related works. Lastly, Section 8 draws a conclusion.

2 Background

This section includes several basic knowledge of deep neural network (DNN) and Metamorphic Testing (MT).

2.1 Deep Neural Network (DNN)

A deep neural network (DNN) [28] is structured as a composite function that maps an input x to an output result y, represented as:

$$y = f(x) = f^{(0)}\left(f^{(1)}\left(\ldots\left(f^{(D)}(x)\right)\right)\right) \tag{1}$$

In the above formula, $f^{(j)}$ represents the function at the j-th layer of the network, with D being the total number of layers. Each layer of $f^{(j)}$ transforms the input into an output vector by applying weights and biases (which can be adjusted during training to minimize the discrepancy between predictions and actual results), followed by an activation function [47]. As the layers go deeper, the model is able to learn more and more complex features.

In classification tasks, the output layer typically uses a *softmax* function to map the final transformed vector into class probabilities, summing up to 1, ensuring it represents a valid probability distribution. The goal of the DNN is to approximate the true classification function $f^*(x)$, providing an output y close to the one-hot encoded ground truth vector $l = f^*(x)$.

2.2 Metamorphic Testing (MT)

Metamorphic Testing (MT) [9] is a testing technique used to alleviate the test oracle problem, particularly when it is difficult or impossible to determine the correct output for a given input. The core idea behind MT is to identify metamorphic relations (MRs) [60], which are properties or relationships between the inputs and outputs of a system that should hold across different test cases, even if the exact expected output is unknown. Formally, let f be a function or system under test that takes an input x and produces an output $y = f(x)$. For multiple inputs $x_1, x_2, \ldots, x_n$ and their corresponding outputs $y_1, y_2, \ldots, y_n$, a MR R is defined as $R(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n)$, where R expresses the expected relationship between inputs and outputs, such as consistency (the outputs change predictably with the inputs) and transformation (modifying an input should lead to a consistent transformation in the output).

Instead of relying on a test oracle, MT checks whether the output adheres to certain expected behaviors described by MRs. As a result, MT can detect faults without requiring explicit knowledge of the correct output. This capability makes the selection of effective metamorphic test case pairs particularly crucial for maximizing the fault detection potential of MT. Formally, given a DNN model F, a set of source test cases $X = \{x_1, x_2, \ldots, x_n\}$, a set of MRs $M = \{MR_1, MR_2, \ldots, MR_m\}$, a test budget k and an objective function $f : 2^{n\times m} \rightarrow \mathbb{R}$ (e.g., number of failures triggered). The metamorphic test case pair selection problem is to identify a subset $S \subseteq \{(x, mr)|x \in X \text{ and } mr \in M\}$ with $|S| = k$, such that for any $S' \subseteq \{(x, mr)|x \in X \text{ and } mr \in M\}$ ($|S'| = k$ and $S \neq S'$), $f(S) \geq f(S')$.

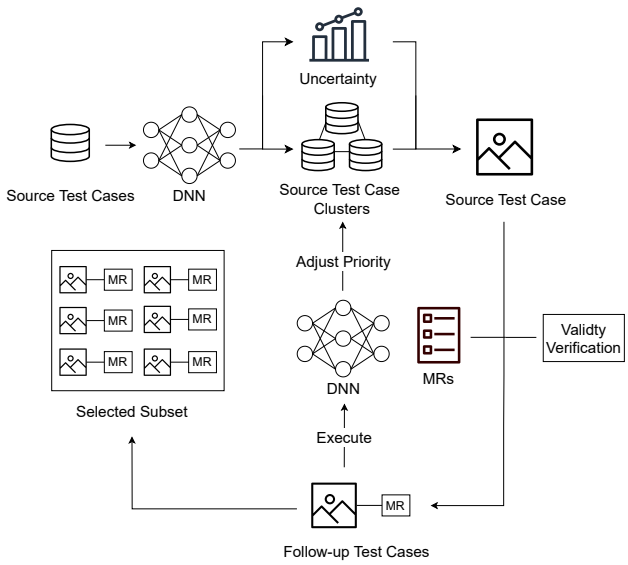


Figure 1: Workflow of CMPS

3 Approach

In this section, we describe the design details of our approach CMPS.

3.1 Overview of CMPS

Figure 1 illustrates the workflow of CMPS. First, CMPS runs all the source test cases on the DNN model under test to obtain their output probabilities. Using these output probabilities, CMPS not only evaluates the uncertainty of each source test case, but also clusters them by grouping together those with similar probability distributions. Considering both uncertainty and diversity, CMPS then iteratively selects a source test case with the highest uncertainty from different clusters, along with the best MR for it. After each selection, CMPS executes the follow-up test case generated by the selected pair (i.e., the selected source test case and MR) to examine whether the pair triggers a failure or not. Based on the execution result, it dynamically adjusts the priority of the cluster to which the source test case belongs in the next iteration (a round of iteration ends when all clusters have been selected once). This process ensures an adaptive and effective selection of metamorphic test case pairs to trigger more failures and detect more unique faults in the model.

3.2 Algorithm

Algorithm 1 outlines the detailed process of the CMPS approach. Given a set of source test cases X and a set of MRs M, CMPS aims to select a subset $S \subseteq \{(x, mr)|x \in X \text{ and } mr \in M\}$ with a pre-defined size k (i.e., the test budget). The goal is for the pairs in S to reveal as many diverse faults as possible in the DNN model under test, F.

Intuitively, the more uncertain the DNN model is about the source test case, the greater uncertainty the DNN model will have regarding the follow-up test case generated from it. Therefore, the model's outputs for the source and follow-up test cases are more likely to violate the MR, that is, triggering failures in the model.

Algorithm 1 The CMPS Approach

Input: the DNN under test F, the set of source test cases X, the set of metamorphic relations M and the budget k
Output: the selected subset S

1: output probabilities $P \leftarrow F(X)$; // run all the source test cases on the model F to obtain their output probabilities P
2: uncertainty value $Un_x \leftarrow DeepGini(P_x)$; // calculate the uncertainty value Un_x of each source test case x based on the DeepGini metric
3: cluster set $C \leftarrow DBSCAN(X, P)$; // cluster source test cases based on P using the DBSCAN algorithm
4: $G \leftarrow C$; $B \leftarrow \emptyset$; // G: good cluster set B: bad cluster set
5: $G_{next} \leftarrow \emptyset$; $B_{next} \leftarrow \emptyset$;
6: $S \leftarrow \emptyset$;
7: **while** $k > 0$ **do**
8: **if** $B = \emptyset \wedge G = \emptyset$ **then**
9: $G \leftarrow G_{next}$; $B \leftarrow B_{next}$;
10: $G_{next} \leftarrow \emptyset$; $B_{next} \leftarrow \emptyset$;
11: **end if**
12: **if** $G \neq \emptyset$ **then** // prioritize good clusters over bad clusters
13: candidate cluster set $L \leftarrow G$;
14: **else**
15: $L \leftarrow B$;
16: **end if**
17: $x = \underset{x \in \bigcup_{C_i \in L} C_i}{\arg\max}\ Un_x$; //select the source test case with the highest uncertainty from the candidate clusters
18: $C_i = C_i \setminus \{x\}$;
19: $mr = \underset{mr \in M}{\arg\max}\ EuclideanDistance(x, mr(x))$;
20: $S = S \cup \{(x, mr)\}$;
21: **if** $F(x) = F(mr(x))$ **then**
22: $B_{next} = B_{next} \cup \{C_i\}$; // this pair cannot trigger a failure and thus the priority of the cluster the source test case belongs to is reduced
23: **else**
24: $G_{next} = G_{next} \cup \{C_i\}$; // this pair can trigger a failure and thus the priority of the cluster the source test case belongs to is increased
25: **end if**
26: **if** $C_i \in G$ **then**
27: $G = G \setminus \{C_i\}$;
28: **else**
29: $B = B \setminus \{C_i\}$;
30: **end if**
31: $k \leftarrow k - 1$;
32: **end while**
33: **return** S

Inspired by this, CMPS first evaluates the uncertainty of all the source test cases based on their output probabilities (Line 1–2). It employs the DeepGini [14] metric, a widely used measure of uncertainty [1, 2, 16], to calculate the uncertainty value for each source test case. Given the output probability $P_x = \langle p_1, p_2, \ldots, p_c \rangle$ of a test case x (p_i represents the probability that the model predicts x to the i-th class, and c is the total number of classes), the DeepGini value of x can be calculated as $1 - \sum_{i=1}^{c} p_i^2$.

However, focusing solely on uncertainty may lead to redundant selection, as many failures triggered by the selected pairs may stem from the same root fault of the DNN model. Therefore, CMPS treats diversity as another optimization objective, aiming to increasing the diversity of the source test cases in the selected pairs, thereby increasing the diversity of the fault that can be revealed. Given that test cases with similar probability distributions may trigger the same behavior of the DNN model [15], test cases with different probability distributions are more likely to expose distinct faults within the model. To this end, CMPS applies a density-based clustering algorithm, DBSCAN [12], to group source test cases based on their output probabilities and obtains the cluster set C (Line 3). Within each cluster $C_i \in C$, test cases share similar probability distributions, while test cases from different clusters exhibit distinct distributions. For example, given a test set $X = \{x_1, x_2, x_3, x_4\}$ with output probabilities: $P_{x_1} = \langle 0.7, 0, 0.3 \rangle$, $P_{x_2} = \langle 0.68, 0, 0.32 \rangle$, $P_{x_3} = \langle 0, 0.6, 0.4 \rangle$, $P_{x_4} = \langle 0, 0.59, 0.41 \rangle$. Using this clustering algorithm, CMPS will divide them into two subsets: $\{x_1, x_2\}$ and $\{x_3, x_4\}$.

Then, CMPS performs some initialization before the selection process. It first initializes two sets: G for good cluster set and B for bad cluster set. Clusters in G are more likely to trigger failures, while those in B less likely to do so. These two sets provide the priorities of the clusters in the current iteration. Before the selection begins, CMPS assumes that all clusters in the candidate set C are good clusters by default, so it initializes G as C, and B as empty (Line 4). Next, two auxiliary sets, G_{next} and B_{next}, are initialized as empty (Line 5), to record the adjusted priority of each cluster in the current iteration. In addition, the target subset S is also initialized as empty (Line 6).

During selection, CMPS first checks whether the current iteration has ended, that is, whether all clusters have been selected once. If both G and B are empty, meaning all clusters have been selected, CMPS updates G and B with G_{next} and B_{next}, respectively, and then resets G_{next} and B_{next} as empty to start the next iteration (Line 8–11). Otherwise, it continues with the current iteration. This strategy enables the selection of more diverse clusters within the given test budget. Next, CMPS sets G as the candidate cluster set L if G is not empty; otherwise, it sets B as L (Line 12–16), indicating that CMPS prioritizes the clusters in G. This prioritization ensures that clusters more likely to trigger failures are processed first. CMPS then selects the source test case x with the highest uncertainty from the cluster $C_i \in L$ and removes x from the cluster C_i to prevent it from being selected again (Line 17–18).

After selecting the source test case x, CMPS then selects the best MR mr from the set M for x (Line 19). According to Chen et al. [10], good MRs should be those that maximize the differences between multiple executions of the program. Additionally, Asrafi et al. [3] demonstrated that MRs are more effective in detecting failures when the source and follow-up test cases caused more varied execution behaviors. Inspired by these studies, we believe that an ideal MR should not only generate valid follow-up test cases but also maximize the difference between the source and follow-up test cases. Therefore, CMPS first verifies the validity of the follow-up test cases generated by all the MRs in M using SSIM [50], a metric widely used to assess quality of the test cases [34, 37]. And then

it uses Euclidean distance, a well-established and straightforward metric, to measure the differences between the source test case and its valid follow-ups. The MR mr that maximizes the distance between the source and follow-up test cases is selected. The selected pair of the source test case and the MR, (x, mr), is then added to the subset S (Line 20).

Then, CMPS executes the follow-up test case $mr(x)$ on the model F and compares the two outputs, $F(x)$ and $F(mr(x))$ (noting that the source test case x has already been executed before). If the model outputs remain unchanged, i.e., $F(x) = F(mr(x))$, it indicates that no failure is triggered by the pair. As a result, the priority of the cluster C_i should be reduced in the next iteration, and C_i is added to B_{next}. Conversely, if the model outputs are different, i.e., $F(x) \neq F(mr(x))$, it suggests the pair triggers a failure. Hence, the priority of the cluster C_i should be increased in the next iteration, and C_i is added to G_{next} (Line 21–25). CMPS then removes the selected cluster C_i from G or B (Line 26–30), to avoid repeated selection in the current iteration. The selection process continues until the test budget k is reached. Finally, CMPS outputs the carefully selected subset S, consisting of pairs of source test cases and MRs (Line 33).

The entire process considers both uncertainty and diversity, aiming to optimize two objectives: 1) triggering as many failures as possible, and 2) maximizing the diversity of the underlying faults causing these failures.

4 Experiment Design

In this study, we set up the following research questions to evaluate the performance of CMPS:

- RQ1: Can CMPS trigger more failures than existing approaches?
- RQ2: Can CMPS detect more diverse faults than existing approaches?
- RQ3: How effective is the clustering component of CMPS?

4.1 Experiment Setup

All the experiments were performed on MacOS 14.5 with AppleM2 Pro and 16GB memory, as well as an NVIDIA GeForce RTX 3090 GPU. We implemented CMPS upon Python 3.8.19 with Keras 2.13.1 and TensorFlow 2.13.0.

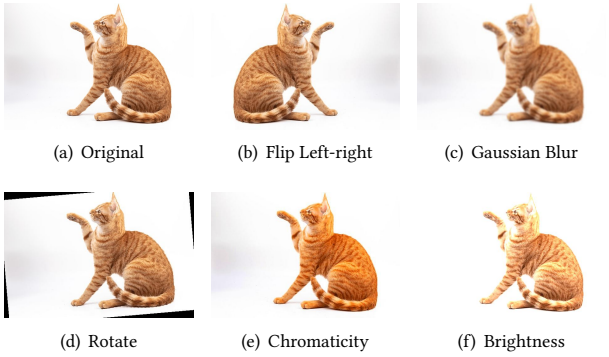
4.2 Datasets and DNN Models

Although the idea of CMPS can be generalized to MT for any type of DNN model, our experiments focused on MT for image classification DNN models. On one hand, using the same type of DNN model facilitates a fair comparison, as previous work [2, 55] also focused on image classification models. On the other hand, the properties of images are relatively intuitive, making the MRs designed for image data easily understandable, even for non-experts. More importantly, recent years have seen extensive research on MT in the image classification domain [13, 39], with a wide variety of valid MRs proposed that can be selected for use.

We selected three widely-used datasets for image classification tasks, and for each dataset, we chose two pre-trained DNN models, resulting in a total of six experimental subjects. Table 1 provides a summary of each dataset's name, the number of images in its corresponding test set, as well as the name and layer count of each DNN model. The Fashion-MNIST [53] dataset consists of 28×28 gray-scale images of clothing items, spanning 10 classes. The CIFAR-10 [24] dataset contains 32×32 color images, divided into 10 classes. The ImageNet [25] dataset is a large-scale collection with 1,000 classes and millions of images. To optimize experimental resources, we randomly selected 10 images from each class as the test set for ImageNet, ensuring that all test sets have the same size.

Table 1: Datasets and Models Used for Experiments

Dataset	Test Set	DNN Model	Layers
Fashion-MNIST	10,000	LeNet1	7
		LeNet5	14
CIFAR-10	10,000	VGG19	27
		ResNet50	177
ImageNet	10,000	GoogleNet	144
		ResNet50	177



(a) Original (b) Flip Left-right (c) Gaussian Blur

(d) Rotate (e) Chromaticity (f) Brightness

Figure 2: Examples of the Selected MRs Applied to an Image of Cat

The selected DNN models consist of five in total. LeNet1 [6] consists of two convolutional layers, sub-sampling layers, and fully connected layers. LeNet5 [26] builds on LeNet1 by adding extra layers to improve digit classification performance. VGG19 [38] is a deep convolutional neural network (CNN) with 19 layers, utilizing small convolution filters throughout the network. GoogleNet [48] introduces the Inception module, which combines multiple filter sizes in parallel to enhance computational efficiency. ResNet50 [5] is a 50-layer deep residual network that uses skip connections to address the vanishing gradient problem, making it particularly effective for image recognition tasks.

4.3 Metamorphic Relations

We selected five different MRs proposed by previous work [2, 39]. Figure 2 shows the effect of MRs on transforming the source test case into its follow-up test cases. A test is deemed to have failed (i.e., a failure) if the follow-up test case generated by the MR yields a different classification result than the source test case.

- ***MR1 - Flip Left-right:*** The DNN model's outputs should remain consistent when the image is flipped from left to right.
- ***MR2 - Gaussian Blur:*** The DNN model's outputs should remain consistent when the image undergoes Gaussian blurring.
- ***MR3 - Rotate 5°:*** The DNN model's outputs should remain consistent when the image is rotated by 5°.
- ***MR4 - Change Chromaticity:*** The DNN model's outputs should remain consistent when the chromaticity of the image is increased.
- ***MR5 - Adjust Brightness:*** The DNN model's outputs should remain consistent when the brightness of the image is increased.

4.4 Baseline Approaches

In our experiment, we compared CMPS with the approach proposed by Arrieta et al. [2], as it is currently the only black-box approach for metamorphic test case pair selection. Since Arrieta et al. implemented this approach based on the NSGA-II [11] algorithm without assigning a specific name to it, we referred to it as **NSGA-II** for the convenience of subsequent discussion. We also use **random selection (RS)** as a standard baseline to provide a useful benchmark for assessing the effectiveness of the two selection approaches.

NSGA-II. As a multi-objective approach, NSGA-II optimizes both uncertainty and size (the number of selected source test cases) simultaneously, aiming to minimize the size while maximizing the uncertainty in the selected source test cases. Before starting the search, NSGA-II runs all source test cases to obtain their output probabilities for uncertainty estimation. It then searches the entire solution space to identify target source test cases for the given MRs. However, since cost is one of NSGA-II's optimization objectives, it cannot select a fixed number of pairs. Furthermore, unlike CMPS, which can handle multiple MRs simultaneously, NSGA-II can select source test cases for only one MR per execution.

To ensure a fair comparison between the two approaches, we randomly sampled $\frac{k}{5}$ (where k is the test budget and 5 is the total number of MRs) source test cases from the optimal subset (i.e., the solution closest to the ideal point on the Pareto front [7]) identified by NSGA-II for each MR. These sampled test cases were paired with the corresponding MR to form the k pairs, which served as the selection results of NSGA-II for the experimental subjects. Given the randomness in both the search algorithm and random sampling, we executed NSGA-II 10 times, with each execution repeating the random sampling 10 times, to obtain its average performance. Additionally, to ensure the accuracy of experimental replication, we directly used the open-source implementation of NSGA-II, configuring the parameters according to the settings recommended by the authors.

RS. RS randomly selects the pairs of source test cases and MRs with the given test budget k. To mitigate the randomness of RS, we executed it 10 times to obtain its average performance.

4.5 Research Questions

4.5.1 ***RQ1: Failure-Triggering Capability.*** Intuitively, the more failures a metamorphic test case pair selection approach can trigger within a given test budget, the more effective it is at identifying weaknesses in the DNN model under test. Therefore, the first research question focuses on evaluating the failure-triggering capability of CMPS. To compare the failure-triggering capability of different metamorphic test case pair selection approaches with a given test budget, we introduced the test relative coverage (TRC) metric proposed by Li et al. [27]. The calculation of TRC is expressed by Eq. 2,

$$TRC = \frac{\#Detected\ Failures}{min(\#Budget, \#Total\ Failures)} \tag{2}$$

where $\#DetectedFailures$ denotes the number of failures triggered by the selected pairs, $\#TotalFailures$ refers to the total number of failures that can be triggered by all possible pairs, and $\#Budget$ refers to the test budget. The purpose of $min(\#Budget, \#TotalFailures)$ is to identify the maximum potential number of failures that the selected subset can trigger.

We considered two different test budgets for selection: $k = 500$ and $k = 1000$, i.e., selecting k pairs, where $k \in \{500,\ 1000\}$. And we applied all the approaches to each subject in Table 1 for comparison. Furthermore, we conducted a series of experiments using the Wilcoxon Signed-Rank Test [31], a non-parametric statistical test to assess the differences in performance between CMPS and baseline approaches. We set the significance level at 0.05. A p-value less than 0.05 indicates a statistically significant difference in performance between these approaches.

4.5.2 ***RQ2: Fault Detection Ability.*** It is insufficient to evaluate the effectiveness of a metamorphic test case pair selection approach solely based on the number of failures it triggers, as many failures may stem from the same root fault of the DNN model. With a given budget, a good metamorphic test case pair selection approach should not only trigger more failures, but also detect more distinct root faults in the DNN model. In this research question, we adopted the definition of ATS [15] to categorize the types of faults. The fault type is defined as follows:

$$Fault\_type(x) = (Label(x),\ Label(x')) \tag{3}$$

For a given source test case x, $Label(x)$ denotes the label predicted by the DNN model. The follow-up test case x' is generated from x and the selected MR. $Label(x')$ denotes the label of x' predicted by the DNN model. A fault occurs when x' leads to a different classification result as x, that is, $Label(x') \neq Label(x)$. For example, a source test case is predicted as "cat" by the DNN model, while its follow-up test case is predicted as "dog". Then the fault type can be denoted as: $(cat,\ dog)$. It's worth noting that $(cat,\ dog)$ and $(dog,\ cat)$ are two different types of faults.

Based on the definition of the fault type, we defined the Fault Detection Rate (FDR) as Eq. 4,

$$FDR = \frac{\#Detected\ Fault\_Types}{min(\#Budget, \#Total\ Fault\_Types)} \tag{4}$$

where $\#Detected\ Fault\_Types$ denotes the number of unique faults detected by the selected pairs and $\#Total\ Fault\_Types$ refers to the total number of possible unique faults detected by the entire test set.

Table 2: Test Relative Coverage (TRC) of CMPS, NSGA-II and RS Approaches

Test Relative Coverage (TRC)		Test Budget k = 500			Test Budget k = 1000		
Dataset	**DNN Model**	**CMPS**	**NSGA-II**	**RS**	**CMPS**	**NSGA-II**	**RS**
Fashion-MNIST	LeNet1	**63%**	36%	36%	**58%**	36%	37%
	LeNet5	**50%**	34%	35%	**45%**	34%	34%
CIFAR-10	VGG19	**53%**	25%	25%	**53%**	24%	26%
	ResNet50	**50%**	22%	20%	**45%**	22%	21%
ImageNet	GoogleNet	**55%**	38%	38%	**53%**	39%	39%
	ResNet50	**59%**	37%	37%	**54%**	36%	37%

Taking the CIFAR-10 dataset as an example, the $\#Total\ Fault\_Types$ is $10 \times 9 = 90$. The purpose of $min(\#Budget, \#Total\ Fault\_Types)$ is to identify the maximum potential number of distinct faults that the selected subset can detect.

Similar to RQ1, we considered two test budgets, $k \in \{500,\ 1000\}$, for selection and applied all the approaches to each subject in Table 1. We also used the Wilcoxon Signed-Rank Test with significance level $\alpha = 0.05$ to evaluate the differences in performance between CMPS and baseline approaches.

4.5.3 ***RQ3: Ablation Experiment.*** Since uncertainty has been proven to be an effective selection criterion in previous work [1, 2, 14, 30], the last research question aims to investigate the impact of the clustering on the effectiveness of CMPS. To this end, we conducted an ablation experiment to compare the performance of CMPS with and without clustering (the version of CMPS without clustering corresponds to Lines 1–2 and 17–20 of Algorithm 1). For clarity in subsequent discussions, we referred to CMPS without clustering as CMPS-NC. We used not only the TRC metric introduced in Section 4.5.1 to compare the failure-triggering capability of CMPS and CMPS-NC, but also the FDR metric defined in Section 4.5.2 to compare the fault detection ability of the two.

As with the previous RQs, we performed experiments on all the subjects listed in Table 1, considering two different budgets for selection: $k \in \{500,\ 1000\}$.

5 Result Analysis

In this section, we analyze the experimental results to answer our research questions.

5.1 RQ_1 (Failure-Triggering Capability)

Table 2 presents the TRCs of CMPS and baseline approaches across all subjects. In this table, bold numbers highlight the highest TRC values, while numbers with a gray background indicate statistically significant differences between the corresponding approach and the other two (i.e., p-values < 0.05). As shown in Table 2, CMPS achieves the highest TRC values across all cases (a case refers to a combination of subject and test budget). With the exception of the case where 1000 pairs are selected for the CIFAR-10&ResNet50 and Fashion-MNIST&LeNet5 subjects, the TRCs of CMPS exceed 50% in all other cases. Among these, the highest TRC value, 63%, is achieved on the Fashion-MNIST&LeNet1 subject at $k = 500$.

Furthermore, CMPS significantly outperforms both the NSGA-II and RS approaches in triggering failures across all cases. Compared to NSGA-II, CMPS improves TRC by an average of 23% at $k = 500$ and 19.5% at $k = 1000$. The greatest improvement is observed on the CIFAR-10&VGG19 subject, where CMPS achieves a 29% increase in TRC at $k = 1000$. In comparison to RS, CMPS improves TRC by an average of 23.17% at $k = 500$ and 19% at $k = 1000$. The most significant improvement is also observed on the CIFAR-10&ResNet50 subject, where CMPS achieves a 30% increase in TRC when selecting 500 pairs.

Notably, NSGA-II performs similarly to RS in most cases and even slightly worse in some, such as selecting 1000 pairs on the ImageNet&ResNet50 subject. This may be due to NSGA-II's variable performance across different MRs. The pairs selected for some MRs can trigger more failures, while those selected for other MRs perform poorly in failure triggering, which impacts NSGA-II's overall performance. This suggests that NSGA-II is highly sensitive to MRs, making it less effective when handling multiple MRs simultaneously.

> **Answer to RQ1:** CMPS significantly outperforms baseline approaches in triggering failures across all subjects and test budgets. The results also indicate that CMPS is more effective than NSGA-II when handling multiple MRs.

5.2 RQ_2 (Fault Detection Ability)

Table 3 shows the FDRs of CMPS and baseline approaches across all subjects. Similar to Table 2, the numbers in bold highlight the highest FDR values, while those with a gray background denote statistically significant differences between the corresponding approach and the other two, with p-values < 0.05.

In terms of fault detection, CMPS achieves the highest FDR values across all cases, with an average FDR of 58.33% at $k = 500$ and 60.5% at $k = 1000$. The FDR value even reaches 80% when selecting 1000 pairs on the CIFAR-10&VGG19 subject. And CMPS significantly outperforms both the NSGA-II and RS approaches in detecting more diverse faults across all cases. Compared to NSGA-II, CMPS improves FDR by an average of 23.33% at $k = 500$ and 19.17% at $k = 1000$. The largest improvement is seen on the CIFAR-10&VGG19 and CIFAR-10&ResNet50 subjects, where CMPS achieves a 43%

Table 3: Fault Detection Rate (FDR) of CMPS, NSGA-II and RS Approaches

Fault Detection Rate (FDR)		Test Budget k = 500			Test Budget k = 1000		
Dataset	**DNN Model**	**CMPS**	**NSGA-II**	**RS**	**CMPS**	**NSGA-II**	**RS**
Fashion-MNIST	LeNet1	**48%**	40%	40%	**54%**	52%	52%
	LeNet5	**46%**	40%	40%	**51%**	47%	47%
CIFAR-10	VGG19	**76%**	33%	34%	**80%**	46%	46%
	ResNet50	**66%**	23%	22%	**71%**	32%	31%
ImageNet	GoogleNet	**55%**	38%	37%	**53%**	37%	37%
	ResNet50	**59%**	36%	36%	**54%**	34%	35%

Table 4: Test Relative Coverage (TRC) and Fault Detection Rate (FDR) of CMPS and CMPS-NC

Dataset	DNN Model	Test Budget k = 500				Test Budget k = 1000			
		TRC		FDR		TRC		FDR	
		CMPS	**CMPS-NC**	**CMPS**	**CMPS-NC**	**CMPS**	**CMPS-NC**	**CMPS**	**CMPS-NC**
Fashion-MNIST	LeNet1	63%	48%	48%	43%	58%	40%	54%	49%
	LeNet5	50%	48%	46%	46%	45%	41%	51%	48%
CIFAR-10	VGG19	53%	57%	76%	76%	53%	54%	80%	79%
	ResNet50	50%	51%	66%	67%	45%	45%	71%	71%
ImageNet	GoogleNet	55%	38%	55%	38%	53%	46%	53%	46%
	ResNet50	59%	41%	59%	41%	54%	47%	54%	47%

increase in FDR at $k = 500$. In comparison to RS, CMPS enhances FDR by an average of 23.5% at $k = 500$ and 19.17% at $k = 1000$. The most notable improvement is also observed on the CIFAR-10&ResNet50 subject, where CMPS achieves a 44% increase in FDR when selecting 500 pairs.

It is worth noting that NSGA-II performs comparably to RS not only in terms of fault triggering (as shown in Table 2), but also in fault detection. This is primarily because NSGA-II's objectives focus on uncertainty and size, which means its fitness function is designed around these two goals while neglecting diversity. As a result, by concentrating solely on the number of failures and overlooking the variety of underlying faults that cause them, NSGA-II may lead to redundant selections.

Answer to RQ2: CMPS can significantly surpass baseline approaches in detecting more diverse faults across all subjects and test budgets. Given its superior fault detection ability, CMPS proves to be a more effective approach compared to the baselines.

5.3 RQ_3 (Ablation Experiment)

Table 4 presents the TRCs and FDRs of CMPS with and without clustering across all subjects. In seven out of all twelve cases, CMPS can not only trigger more failures but also detect more unique faults than CMPS-NC. The average improvement in TRC is 16.67% at $k = 500$ and 9% at $k = 1000$, while the average improvement in FDR is 13.33% at $k = 500$ and 5.5% at $k = 1000$. Notably, on the ImageNet&ResNet50 subject, the improvements in both TRC and FDR reach as high as 18% at $k = 500$.

In the remaining five cases, there are two cases, involving selecting pairs on the Fashion-MNIST&LeNet5 subject at $k = 500$, and on the CIFAR-10&ResNet50 subject at $k = 1000$, where, although the FDRs of CMPS and CMPS-NC are the same (i.e., the numbers of distinct faults detected are the same), the number of failures triggered by CMPS is no fewer than that of CMPS-NC. Additionally, there are two other cases, including the selection of pairs on the CIFAR-10&VGG19 subject at $k = 500$ and $k = 1000$, where, despite the TRCs of CMPS being lower than those of CMPS-NC (indicating fewer failures triggered), CMPS can still detect the same or a greater number of unique faults compared to CMPS-NC.

There is only one exception (selecting 500 pairs on the CIFAR-10& ResNet50 subjects) where CMPS triggers slightly fewer failures and detects fewer distinct faults. However, the differences in both TRC and FDR are only 1%, which is very small and can even be considered negligible. Overall, clustering can indeed enhance the performance of CMPS in both failure triggering and fault detection across various testing scenarios.

Answer to RQ3: The results demonstrate the effectiveness of clustering in improving the performance of CMPS. Clustering not only enables CMPS to trigger more failures, but also facilitates the detection of a wider variety of unique faults.

6 Threat To Validity

Internal validity. The primary threat to internal validity stems from the implementation of CMPS and the baseline approaches. To mitigate this threat, we utilize publicly available open-source implementations for the baseline approaches and configure the parameters recommended by the authors, ensuring their correctness and consistency. Additionally, we rigorously review and verify the source code of CMPS and baseline approaches to prevent potential implementation errors. Furthermore, since the baseline approaches involve randomness in their execution, we perform multiple experimental replications to minimize the impact of random variations and enhance the reliability of our results.

External validity. The main threat to external validity arises from the selection of datasets, DNN models and MRs. To alleviate this threat, we intentionally choose a diverse range of widely-used datasets and DNN models from prior studies [1, 2, 14, 16, 55], ensuring that our findings are generalizable across different scenarios. Furthermore, to ensure the quality of the MRs, we carefully select them from previous work [39], which has demonstrated their effectiveness in various testing scenarios. These measures allow us to strengthen the internal and external validity of our study, increasing the confidence in the applicability of our approach to real-world scenarios.

7 Related Work

In this section, we review related work on MT for DNNs, focusing on MR selection, test case selection, and metamorphic test case pair selection. We summarize the key contributions of these studies and highlight how our work differs from theirs.

7.1 MR Selection

Metamorphic Testing (MT), originating from traditional software testing, has seen the development of numerous MR selection approaches tailored for conventional MT, including those leveraging accumulative code coverage [3], coverage distance between test cases [8], sub-relations [19], and multiple-level execution diversity [40]. Furthermore, Huang et al. [18] established test adequacy criteria for MR selection. Srinivasan et al. [41] contributed with fault-based and coverage-based strategies, whereas Zhang et al. [58] employed mutation scores to guide MR selection. In the most recent advancements, Xie et al. [54] proposed the MUT Model to analyze MR diversity, and Ying et al. [56] developed MRGS-ART, a MR selection algorithm based on adaptive random testing.

In comparison, research on MT for DNN-based software is still in its nascent stages and remains relatively limited. Srinivasan et al. [42] pioneered an approach for prioritizing MRs in machine learning models, leveraging diversity between source and follow-up test cases. Subsequently, they expanded their work by introducing MR selection approaches based on diversity in the execution profiles of source and follow-up test cases [44], as well as data diversity [43].

However, these approaches primarily focus on MR selection and fail to explore the potential advantages of integrating test case selection into the process. In contrast, CMPS adopts a more comprehensive approach by combining both test case and MR selection. This strategy leverages clusters and uncertainty to refine and optimize the selection process, thereby enhancing the effectiveness of MT for DNN-based software.

7.2 Test Case Selection

Numerous studies have proposed various metrics for test case selection in DNNs. In the realm of white-box metrics, drawing inspiration from structural coverage techniques in traditional software testing, many neuron-coverage metrics have been developed, including neuron coverage (NC) [32], neuron boundary coverage (NBC) [29] and modified condition/decision coverage (MC/DC) [45]. Additionally, a range of metrics has been proposed to quantify the surprise adequacy (SA) between test cases and training data, such as likelihood-based surprise adequacy (LSA) [20] and distance-based surprise adequacy (DSA) [20], along with their variations like MDSA [21], MLSA [22], and MMDSA [22]. However, the white-box nature of these metrics poses limitations, as access to the internal structure of DNN models and training datasets is often restricted.

To overcome the limitations of white-box metrics, many black-box metrics based on uncertainty have been proposed, including DeepGini [14], Maxp [30], Entropy [51] and Prediction-Confidence Score (PCS) [51]. These metrics facilitate the selection of test cases exhibiting high uncertainty by assessing the confidence level of the DNN model in its predictions. The underlying idea is that the more uncertain the model is about a given test case, the more likely it is to make misprediction. These metrics leverages the model's own predictive behavior to identify potentially problematic inputs, thereby enhancing the effectiveness of test case selection without requiring access to the internal architecture or training data of the DNN.

However, these metrics are primarily designed for general DNN testing and lack specific adaptations for MT-based testing scenarios. Therefore, in the context of MT for DNNs, it is essential to account for the characteristics of MT and incorporate MR selection into the selection process. By doing so, the selection process can be better aligned with the goals of MT.

7.3 Metamorphic Test Case Pair Selection

Zhang and Xie [59] utilized several simple neuron coverage-based diversity metrics to conduct an initial investigation into the neuron activity across different metamorphic test case pairs. Subsequently, Xie et al. [55] introduced a white-box approach to prioritize metamorphic test case pairs for DNNs, aiming to enhance the effectiveness of MT. Their approach employs a designed metric with higher accuracy to measure the execution diversity of the DNN model on these pairs, based on the distribution discrepancy of neuron outputs, and prioritizes them with high fault sensitivity. However, the reliance on access to the internal structures of DNN models restricts its applicability in many real-world testing scenarios as such access is often unavailable.

In contrast, Arrieta et al. [2] proposed a search-based black-box approach using the Non-dominated Sorting Genetic Algorithm (NSGA-II [11]) to select source test cases for MRs. As a multi-objective approach, it aims to maximize the uncertainty of the selected test cases and minimize the size of the selected subset. However, this approach can only select test cases for a single MR per execution, limiting its practicality when dealing with multiple MRs.

Our approach, CMPS, addresses the limitations of existing black-box ones by enabling the simultaneous handling of multiple MRs. Furthermore, by optimizing both uncertainty and diversity, CMPS is capable of triggering more failures and detecting a greater number of unique faults than current black-box approaches. This enhancement significantly boosts its practicality and effectiveness in real-world testing scenarios.

8 Conclusion

In this paper, we propose CMPS, a novel black-box metamorphic test case pair selection approach designed to enhance the effectiveness of MT for DNNs. As a multi-objective approach, CMPS sets uncertainty and diversity as its optimization goals, aiming to maximize both the number of failures triggered by the selected pairs and the diversity of root faults from which these failures originate. By evaluating the model's uncertainty for each source test case, CMPS identifies candidates most likely to trigger incorrect behavior of the DNN model. Through clustering source test cases, CMPS selects diverse source test cases from those with high uncertainty, enabling a broader exploration of potential fault regions. For each selected source test case, CMPS carefully selects the most suitable MR to form the target pair. Additionally, CMPS utilizes feedback information from execution results of the selected pairs to adaptively refine its selection process. CMPS has demonstrated its potential as a promising approach to improving the effectiveness of MT for DNNs. In future work, we plan to explore the integration of additional model characteristics and further optimize the pair selection process to enhance its scalability and applicability to more complex scenarios.

9 Replication Package

The source code of our CMPS approach, experiment scripts, and all experimental data are available at GitHub: https://github.com/GIST-NJU/CMPS.

Acknowledgments

This work is supported in part by National Natural Science Foundation of China (No. 62472209), Natural Science Foundation of Jiangsu Province (No. BK20221439), and Primary Research and Development Plan of Jiangsu Province (No. BE2023025-2).

References

[1] 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.

[2] Aitor Arrieta. 2022. Multi-objective metamorphic follow-up test case selection for deep learning systems. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 1327–1335.

[3] Mahmuda Asrafi, Huai Liu, and Fei-Ching Kuo. 2011. On testing effectiveness of metamorphic relations: A case study. In *2011 fifth international conference on secure software integration and reliability improvement*. IEEE, 147–156.

[4] Earl T. Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo. 2015. The Oracle Problem in Software Testing: A Survey. *IEEE Transactions on Software Engineering* 41, 5 (2015), 507–525. doi:10.1109/TSE.2014.2372785

[5] Hanane Bennasar, Ahmed Bendahmane, and Mohammed Essaaidi. 2017. An overview of the state-of-the-art of cloud computing cyber-security. In *Codes, Cryptology and Information Security: Second International Conference, C2SI 2017, Rabat, Morocco, April 10–12, 2017, Proceedings-In Honor of Claude Carlet 2*. Springer, 56–67.

[6] Léon Bottou, Corinna Cortes, John S Denker, Harris Drucker, Isabelle Guyon, Larry D Jackel, Yann LeCun, Urs A Muller, Edward Sackinger, Patrice Simard, et al. 1994. Comparison of classifier methods: a case study in handwritten digit recognition. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3-Conference C: Signal Processing (Cat. No. 94CH3440-5)*, Vol. 2. IEEE, 77–82.

[7] Jürgen Branke, Kalyanmoy Deb, Henning Dierolf, and Matthias Osswald. 2004. Finding knees in multi-objective optimization. In *Parallel Problem Solving from Nature-PPSN VIII: 8th International Conference, Birmingham, UK, September 18-22, 2004. Proceedings 8*. Springer, 722–731.

[8] Yuxiang Cao, Zhi Quan Zhou, and Tsong Yueh Chen. 2013. On the correlation between the effectiveness of metamorphic relations and dissimilarities of test case executions. In *2013 13th International Conference on Quality Software*. IEEE, 153–162.

[9] Tsong Y Chen, Shing C Cheung, and Shiu Ming Yiu. 2020. Metamorphic testing: a new approach for generating next test cases. *arXiv preprint arXiv:2002.12543* (2020).

[10] Tsong Yueh Chen, DH Huang, TH Tse, and Zhi Quan Zhou. 2004. Case studies on the selection of useful relations in metamorphic testing. In *Proceedings of the 4th Ibero-American Symposium on Software Engineering and Knowledge Engineering (JIISIC 2004)*. Citeseer, 569–583.

[11] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation* 6, 2 (2002), 182–197.

[12] Dingsheng Deng. 2020. DBSCAN clustering algorithm based on density. In *2020 7th international forum on electrical engineering and automation (IFEEA)*. IEEE, 949–953.

[13] Anurag Dwarakanath, Manish Ahuja, Samarth Sikand, Raghotham M Rao, RP Jagadeesh Chandra Bose, Neville Dubash, and Sanjay Podder. 2018. Identifying implementation bugs in machine learning based image classifiers using metamorphic testing. In *Proceedings of the 27th ACM SIGSOFT international symposium on software testing and analysis*. 118–128.

[14] 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.

[15] 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.

[16] 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.

[17] The Last Driver License Holder. 2022. 2021 Disengagement Report from California. https://thelastdriverlicenseholder.com/2022/02/09/2021-disengagement-report-from-california/. Accessed: 2024-10-22.

[18] Dafei Huang, Yang Luo, and Meng Li. 2022. Metamorphic Relations Prioritization And Selection Based on Test Adequacy Criteria. In *2022 4th International Academic Exchange Conference on Science and Technology Innovation (IAECST)*. IEEE, 503–508.

[19] Zhan-Wei Hui, Song Huang, Hui Li, Jian-Hao Liu, and Li-Ping Rao. 2015. Measurable metrics for qualitative guidelines of metamorphic relation. In *2015 IEEE 39th Annual Computer Software and Applications Conference*, Vol. 3. IEEE, 417–422.

[20] 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.

[21] Jinhan Kim, Robert Feldt, and Shin Yoo. 2023. Evaluating surprise adequacy for deep learning system testing. *ACM Transactions on Software Engineering and Methodology* 32, 2 (2023), 1–29.

[22] Seah Kim and Shin Yoo. 2021. Multimodal surprise adequacy analysis of inputs for natural language processing DNN models. In *2021 IEEE/ACM International Conference on Automation of Software Test (AST)*. IEEE, 80–89.

[23] Hattori Kohei, Tsubasa Hirakawa, Takayoshi Yamashita, and Hironobu Fujiyoshi. 2023. Learning from AI: An Interactive Learning Method Using a DNN Model Incorporating Expert Knowledge as a Teacher. In *International Conference on Artificial Intelligence in Education*. Springer, 435–446.

[24] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. 2009. Cifar-10 and cifar-100 datasets. *URl: https://www. cs. toronto. edu/kriz/cifar. html* 6, 1 (2009), 1.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012).

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

[27] Yu Li, Min Li, Qiuxia Lai, Yannan Liu, and Qiang Xu. 2021. Testrank: Bringing order into unlabeled test instances for deep learning tasks. *Advances in Neural Information Processing Systems* 34 (2021), 20874–20886.

[28] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. 2017. A survey of deep neural network architectures and their applications. *Neurocomputing* 234 (2017), 11–26.

[29] 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.

[30] 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.

[31] Thomas W MacFarland, Jan M Yates, et al. 2016. *Introduction to nonparametric statistics for the biological sciences using R*. Springer.

[32] 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.

[33] Kun Qiu, Zheng Zheng, Tsong Yueh Chen, and Pak-Lok Poon. 2020. Theoretical and empirical analyses of the effectiveness of metamorphic relation composition. *IEEE Transactions on software engineering* 48, 3 (2020), 1001–1017.

[34] Megha Rani Raigonda et al. 2024. Signature Verification System Using SSIM In Image Processing. *Journal of Scientific Research and Technology* (2024), 5–11.

[35] Vincenzo Riccio, Gunel Jahangirova, Andrea Stocco, Nargiz Humbatova, Michael Weiss, and Paolo Tonella. 2020. Testing machine learning based systems: a systematic mapping. *Empirical Software Engineering* 25 (2020), 5193–5254.

[36] Madona B Sahaai et al. 2021. Brain tumor detection using DNN algorithm. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)* 12, 11 (2021), 3338–3345.

[37] Umme Sara, Morium Akter, and Mohammad Shorif Uddin. 2019. Image quality assessment through FSIM, SSIM, MSE and PSNR—a comparative study. *Journal of Computer and Communications* 7, 3 (2019), 8–18.

[38] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[39] Helge Spieker and Arnaud Gotlieb. 2020. Adaptive metamorphic testing with contextual bandits. *Journal of Systems and Software* 165 (2020), 110574.

[40] Madhusudan Srinivasan. 2018. Prioritization of metamorphic relations based on test case execution properties. In *2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 162–165.

[41] Madhusudan Srinivasan and Upulee Kanewala. 2022. Metamorphic relation prioritization for effective regression testing. *Software Testing, Verification and Reliability* 32, 3 (2022), e1807.

[42] Madhusudan Srinivasan and Upulee Kanewala. 2022. Prioritization of Metamorphic Relations to reduce the cost of testing. *arXiv preprint arXiv:2209.00162* (2022).

[43] Madhusudan Srinivasan and Upulee Kanewala. 2024. Improving Early Fault Detection in Machine Learning Systems Using Data Diversity-Driven Metamorphic Relation Prioritization. *Electronics* 13, 17 (2024), 3380.

[44] Madhusudan Srinivasan and Upulee Kanewala. 2024. Optimizing Metamorphic Testing: Prioritizing Relations Through Execution Profile Dissimilarity. *arXiv preprint arXiv:2411.09171* (2024).

[45] Youcheng Sun, Xiaowei Huang, Daniel Kroening, James Sharp, Matthew Hill, and Rob Ashmore. 2018. Testing deep neural networks. *arXiv preprint arXiv:1803.04792* (2018).

[46] Zeyu Sun, Jie M Zhang, Yingfei Xiong, Mark Harman, Mike Papadakis, and Lu Zhang. 2022. Improving machine translation systems via isotopic replacement. In *Proceedings of the 44th international conference on software engineering*. 1181–1192.

[47] Tomasz Szandała. 2021. Review and comparison of commonly used activation functions for deep neural networks. *Bio-inspired neurocomputing* (2021), 203–224.

[48] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[49] Shuai Wang and Zhendong Su. 2020. Metamorphic object insertion for testing object detection systems. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*. 1053–1065.

[50] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.

[51] 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.

[52] Elaine J Weyuker. 1982. On testing non-testable programs. *Comput. J.* 25, 4 (1982), 465–470.

[53] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).

[54] Xiaodong Xie, Zhehao Li, Jinfu Chen, Yue Zhang, Xiangxiang Wang, and Patrick Kwaku Kudjo. 2024. MUT Model: A metric for characterizing metamorphic relations diversity. *Software Quality Journal* (2024), 1–43.

[55] Xiaoyuan Xie, Pengbo Yin, and Songqiang Chen. 2022. Boosting the revealing of detected violations in deep learning testing: A diversity-guided method. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*. 1–13.

[56] Zhihao Ying, Dave Towey, Anthony Graham Bellotti, and Zhi Quan Zhou. 2025. MRGS-ART: Metamorphic Relation and Group Selection Based on Adaptive Random Testing. *Software Testing, Verification and Reliability* 35, 1 (2025), e1908.

[57] Yuan Yuan, Chunfu Shao, Zhichao Cao, Zhaocheng He, Changsheng Zhu, Yimin Wang, and Vlon Jang. 2020. Bus dynamic travel time prediction: using a deep feature extraction framework based on RNN and DNN. *Electronics* 9, 11 (2020), 1876.

[58] Jie Zhang, Jie Hong, Dafei Huang, Meng Li, Shiyu Yan, and Helin Gong. 2022. A Selection Method of Effective Metamorphic Relations. In *2022 13th International Conference on Reliability, Maintainability, and Safety (ICRMS)*. IEEE, 75–80.

[59] Zhiyi Zhang and Xiaoyuan Xie. 2019. On the investigation of essential diversities for deep learning testing criteria. In *2019 IEEE 19th International Conference on Software Quality, Reliability and Security (QRS)*. IEEE, 394–405.

[60] Zhi Quan Zhou, Liqun Sun, Tsong Yueh Chen, and Dave Towey. 2018. Metamorphic relations for enhancing system understanding and use. *IEEE Transactions on Software Engineering* 46, 10 (2018), 1120–1154.