Multi-Objective Metamorphic Follow-up Test Case Selection for Deep Learning Systems

Aitor Arrieta
Mondragon University
Mondragon, Spain
aarrieta@mondragon.edu

ABSTRACT

Deep Learning (DL) components are increasing their presence in safety and mission-critical software systems. To ensure a high dependability of DL systems, robust verification methods are required, for which automation is highly beneficial (e.g., more test cases can be executed). Metamorphic Testing (MT) is a technique that has shown to alleviate the test oracle problem when testing DL systems, and therefore, increasing test automation. However, a drawback of this technique lies into the need of multiple test executions to obtain the test verdict (named as the source and the follow-up test cases), requiring additional testing cost. In this paper we propose an approach based on multi-objective search to select follow-up test cases. Our approach makes use of source test cases to measure the uncertainty provoked by such test inputs in the DL model, and based on that, select failure-revealing follow-up test cases. We integrate our approach with the NSGA-II algorithm. An empirical evaluation on three DL models tackling the image classification problem, along with five different metamorphic relations demonstrates that our approach outperformed the baseline algorithm between 17.09 to 59.20% on average when considering the revisited Hypervolume quality indicator.

CCS CONCEPTS

• **Software and its engineering → Search-based software engineering**.

KEYWORDS

Metamorphic Testing, Test Case Selection, Multi-Objective search, Deep Learning Systems

ACM Reference Format:
Aitor Arrieta. 2022. Multi-Objective Metamorphic Follow-up Test Case Selection for Deep Learning Systems. In *Genetic and Evolutionary Computation Conference (GECCO '22), July 9–13, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3512290.3528697

1 INTRODUCTION

Deep Learning (DL) components are commonly integrated into critical software systems that need to perform complex tasks, including image processing of autonomous vehicles and medical systems predicting diseases [50]. Thoroughly testing such systems is paramount to ensure a high dependability of critical systems. To allow a robust verification approach of such systems, automation is necessary, which therefore requires test oracles (i.e., the mechanism in charge of determining whether a test input passes or fails). Often, it is infeasible to determine which the test outcome should be, which is known as the test oracle problem. Metamorphic testing is a technique that alleviates the test oracle problem. This technique is based on the intuition that often it is simpler to reason about relations between the inputs/outputs of multiple, related test executions, rather than the relations between inputs/outputs of each individual test execution [7].

Metamorphic oracles, which take advantage of metamorphic relations between input values, have emerged as a viable approach to overcome the test oracle problem of DL systems [36]. In fact, in a recent systematic mapping on testing machine-learning based systems, Riccio et al., [36] found that out of 13 studies tackling the test oracle problem, 12 of them used metamorphic oracles. Such technique has demonstrated success in alleviating the oracle problem in a wide range of DL applications, including autonomous driving systems [48], image classification [13, 41] and object detection [41].

However, the application of metamorphic testing consumes additional testing resources as it involves multiple program executions [35]. To increase the cost-effectiveness of metamorphic testing, prior studies have proposed to reduce the number of Metamorphic Relations (MRs) through *MR composition* [21, 22]. However, whether the fault detection capability of the composite MR is higher or similar to its corresponding component MRs has been controversial. While Dong et al., [21] reported that the fault detection capability remains unchanged, Liu et al., [22] found that the fault detection capability may be reduced after MR composition. Our recent study showed that such an approach reduces the failure detection capability of MRs when testing DL systems [3].

To overcome this problem, this paper proposes a test case selection approach of DL systems that uses metamorphic testing in their verification process. Specifically, metamorphic testing predicates on the input and output of multiple test inputs (source and follow-up test cases). We conjecture that a high confidence of the DL model when executing the source test case leads to a high probability of the MR holding with respect to its corresponding follow-up test case too. Therefore, our approach aims at favoring the selection of follow-up test cases with high probability of violating the MR by selecting follow-up test cases whose source test cases have shown a low confidence. Since the test case selection problem is multi-objective in nature, we use a Pareto-based test case selection approach based on the NSGA-II algorithm, a widely used algorithm

GECCO '22, July 9–13, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9237-2/22/07…$15.00
https://doi.org/10.1145/3512290.3528697

for solving the test case selection problem [4–6, 31, 33, 34, 42, 44]. Specifically, the main contributions of this paper can be summarized as follows:

- We propose a multi-objective test case selection approach for selecting metamorphic follow-up test cases for testing DL systems based on information obtained from the execution of source test cases. To the best of our knowledge, this is the first paper that proposes a similar approach both in the context of metamorphic testing and in the application domain of DL systems.
- We apply this approach to the context of image classification DL models. We used a total of five Metamorphic Relations (MRs) defined by Spieker and Gotlieb [41] and three state-of-the-art DL models. We empirically evaluated the approach with the Imagenette dataset, containing a total of 3,845 images (i.e., test cases in our context). Our approach outperformed with statistical significance the baseline techniques for all the defined MRs in the three selected models.
- We make all our sources and experimental material available for replication and reproducibility purposes.

The rest of the paper is structured as follows: Section 2 explains basic background related to metamorphic testing and DL systems. In Section 3 we explain our approach for metamorhpic follow-up test case selection of DL systems. In Section 4 we introduce the application domain at which we evaluated our approach and the selected metamorphic relations. We empirically evaluate our approach in Section 5. We position our work with respect to the state-of-the-art in Section 6. Lastly, we conclude our paper and discuss future research avenues in Section 7.

2 BACKGROUND

This section explains basic concepts of metamorphic testing and DL systems.

2.1 Metamorphic Testing

Metamorphic testing (MT) [8, 40] is a technique that aims to detect faults by looking at the relations among the inputs and outputs of two or more executions of the system under test (SUT), so called *metamorphic relations* (MRs). For instance, consider the program $search(T)$ that searchers for a string T in Google, as shown in Figure 1, where T refers to the string "software". Naturally, checking all the results returned by Google is unfeasible. This is an instance of the oracle problem. Suppose we create a new string T' by adding an independent text fragment S at the end of T: $T' = T + \{S\}$. In the example of Figure 1, "S" refers to the string " product lines". Intuitively, the number of results returned for the string T' should be less than or equal to those in T, as can be seen in Figure 1. In this relation, (T) is the *source test case*, and (T'), which is created by extending the input text in T, is the *follow-up test case*. This MR can be instantiated into one or more *metamorphic tests* by using specific input values and checking whether the relation holds. The metamorphic test is said to have failed if the relation is violated. MT has been applied in multiple domains, including compilers, cybersecurity and bioinformatics, among others [9, 39]. In the last few years, industrial applications of MT have also been reported at high-tech companies like Facebook [1] and Google [12].

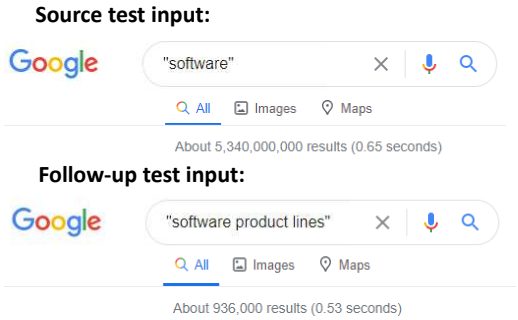

Figure 1: Example of a metamorphic test for testing the Google search engine

2.2 Deep Learning Systems

Unlike traditional software development, where the logic of a program is crafted by developers in the form of source code, DL follows a data-driven programming paradigm [25]. The logic of a DL system is encoded in a deep neural network (DNN), which is represented by sets of weights fed into non-linear activation functions [25]. The DNN is built by running the training program on a set of training data that should be prepared by the DL developer [25]. The learned functions can later be applied in unseen data, allowing to make predictions about unknown properties of observed data [36]. When learned, these functions can be represented and stored as a set of hyper-parameters and variables in a *model*, which is defined as an instance of a specific machine-learning algorithm [36].

3 TEST CASE SELECTION OF METAMORPHIC FOLLOW-UP TEST CASES

Approach overview: Figure 2 shows the overview of our approach, which contemplates five steps. In *step 1*, the source test cases are executed. In this execution, the test output (i.e., the prediction, such as labeling of an image) and the confidence level is obtained. The confidence level refers to the maximum prediction probability score of an input in a DL model [26]. Despite its simplicity, this metric has been found to be effective at quantifying the uncertainty provoked by an input in a DL model [26]. This confidence level is obtained by the test selection algorithm in *Step 2*, which uses such information to predict the uncertainty provoked by the source test cases in the DL model. The test case selection approach uses a search algorithm, NSGA-II in our case, to select a subset of test cases from an initial test suite (selected test suite). With this information, the follow up test cases are generated applying the corresponding test transformation based on the selected MR. In *step 4*, the selected follow-up test cases are executed. In a last step (i.e., *Step 5*), the output of both source and follow-up test cases are compared based on the MR. For instance, in the application we used in the evaluation (i.e., image classification), the test output is referred to the label of the DL model for an image. Based on the defined MRs, both outputs (i.e., labels) shall be the same.

Formalization: Let $TS = \{tc_1, tc_2, ..., tc_N\}$ be a test suite (TS) composed by N test cases (tc). The cost-effectiveness of a test suite is measured through a set of p objective functions (of) to be satisfied when selecting test cases: $OF = \{of_1, of_2, ..., of_p\}$ [31]. The

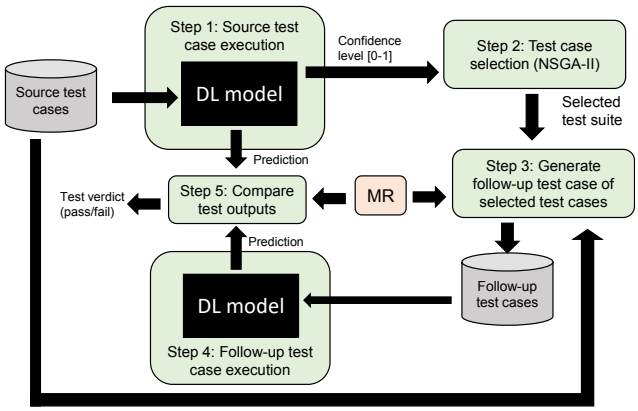

Figure 2: Overview of the proposed approach

test case selection problem aims at selecting a subset of test cases from TS, such that $TS' = \{tc_1, tc_2, ..., tc_M\}$ is a subset of TS (i.e., $TS' \subseteq TS$), that is Pareto-optimal with respect to the objective functions in OF and $M \leq N$. Our test case selection approach is slightly different, because in our case, we are aiming to select metamorphic follow-up test cases. Therefore, let $STS = \{stc_1, stc_2, ..., stc_N\}$ be the source test suite (STS) composed by N source test cases (stc). For each of these test cases, a follow-up test case (ftc) can be generated by applying a transformation to the source test case based on a metamorphic relation (MR). Thus, for any test case i in STS a transformation function tf can generate the follow-up test case based on an MR: $ftc_i = tf(stc_i, MR)$. Therefore, let $FTS_{MR} = \{ftc_1, ftc_2, ..., ftc_N\}$ be the follow-up test suite composed by N follow-up test cases, which are obtained by transforming the source test cases from STS based on a specific MR. Our test case selection approach aims at selecting test cases from FTS_{MR}, such that $FTS'_{MR} = \{ftc_1, ftc_2, ..., ftc_M\}$ is a subset of FTS_{MR} (i.e., $FTS' \subseteq FTS$).

Solution encoding: We use a binary coding representation of solution, like most multi-objective test case selection studies [4–6, 30, 31, 33, 34, 42, 44]. This means that if the i-th digit of the binary string is 1, the follow-up test case ftc_i from FTS_{MR} is included in the solution. Conversely, if the i-th digit of the binary string is 0, ftc_i will not be included.

Search space: The search space for the test case selection problem is 2^N, N being the number of test cases in the initial test suite. A test suite of 1,000 test cases has a total of 1.0715×10^{301} potential solutions. In our experiments we used a total of 3,845 test cases. Thus, the search space is huge in order brute force to be applied. Therefore, the use of search algorithms is justified.

Fitness functions: We used two fitness functions to guide the multi-objective algorithm towards optimal solutions: one measures the cost, and the second is the uncertainty provoked when executing a set of source test cases. Both fitness functions conflict with each other.

Let $FTS_{init} = \{ftc_1, ftc_2, ..., ftc_N\}$ be the initial test suite. Define a set of follow-up test cases $FTS_{select} = \{ftc_1, ftc_2, ..., ftc_M\}$, that are selected from FTS_{init}. As a cost function we used the normalized number of test cases to be selected. Prior studies have used the test execution time [5, 6] as cost function. This fitness function is good especially when there are high differences in the test execution times between test cases (e.g., Cyber-Physical Systems [5, 6]). In our case, since all test cases are images, and there is no clear difference in the execution of tests among them, we resort to the total number of selected test cases as the cost function of the test case selection problem. Thus, the normalized number of test cases score of a set of selected test cases is:

$$NTC_{select} = M/N \tag{1}$$

The search algorithm aims at **minimizing** this metric, favoring solutions with fewer test cases.

For the effectiveness fitness function, we conjecture that an MR is more likely to be violated if the uncertainty provoked by the source test case in the DL model is higher. A previous study observed that there was a positive correlation between uncertainty metrics and missclassification on both real and artificial (adversarial) test inputs [26]. However, in that study, the missclassification was intended to be verified manually. In our case, we consider a missclassification to be a violation of an MR, allowing full test automation. Thus, we need to validate such hypothesis again because, a missclassification can occur with both the source and follow-up test cases executed in the DL model, but both can classify the input with the same label and not violate the MR. To quantify this uncertainty, we make use of the confidence level of a DL model when doing a prediction (i.e., in our case, since we have applied our approach to an image classification problem, the prediction is the label of the image). The confidence is a value between 0 and 1. The higher this value, the lower the uncertainty. Therefore, when executing a source test case i (stc_i) in Step 1 (Figure 2), we quantify its uncertainty as follows:

$$un_{stc_i} = 1 - conf_{stc_i} \tag{2}$$

where $conf_{stc_i}$ is the confidence value returned by the DL model when executing stc_i.

Based on the above definition, let un_{s_a} denote the uncertainty of the source test case corresponding to the follow-up test case ftc_a in FTS_{select}. To normalize such a function, let un_{s_b} be the uncertainty of the source test case corresponding to the follow-up test case ftc_b in FTS_{init}. The normalized uncertainty score of a set of selected test cases is:

$$UN_{select} = \frac{\sum_{a=1}^{M} un_{s_a}}{\sum_{b=1}^{N} un_{s_b}} \tag{3}$$

The search algorithm aims at **maximizing** this metric, favoring the inclusion of follow-up test cases whose source test case exhibited a high uncertainty in the DL model.

4 APPLICATION DOMAIN AND SELECTED METAMORPHIC RELATIONS

While our approach could be generalized to any kind of DL system using metamorphic testing as a technique to alleviate the test oracle problem, our experiments were limited to the context of image classification. DL models in this application receive an image as input and assign a single class label to the image [41]. We chose this

(a) Original image **(b) MR_1 – Flip Left Right** **(c) MR_2 – Flip Up Down**

(d) MR_3 – Rotate 5° **(e) MR_4 – Rotate -5°** **(f) MR_5 – Shear**

Figure 3: Examples of the selected MRs applied to an image of *Lur*, the author's dog

application domain due to different reasons. First, this application has been widely investigated in the past in the context of metamorphic testing [13, 41]. This allowed us to analyze and select different metamorphic relations (MRs) proposed by other researchers. Secondly, there are many pre-trained DL models available targeting the image classification problem. Such DL models are possible to use in many different programming languages. Therefore, our approach can easily be replicated by other researchers. Lastly, it is easy to find a dataset of labeled images in this application context.

We defined a total of 5 MRs, one of which was configured to two different MRs. Following the MRs between source and follow-up test cases, a test is considered failed if the transformed image leads to a different classification result as the original image. Figure 3 shows the effect of the MRs in the image transformation from the original image (i.e., source test case), to their transformation (i.e., follow-up test cases). These proposed MRs were considered from a previous work [41] and relate to:

- ***MR1 – Flip left-right:*** The classification of the DL model shall be the same when flipping the image from the left to the right.
- ***MR2 – Flip up-down:*** The classification of the DL model shall be the same when flipping the image upside down.
- ***MR3 – Rotate 5°:*** The classification of the DL model shall be the same when rotating the image by 5°.
- ***MR4 – Rotate -5°:*** The classification of the DL model shall be the same when rotating the image by -5°.
- ***MR5 – Shear:*** The classification of the DL model shall be the same when the image is sheared.

5 EMPIRICAL EVALUATION

In our empirical evaluation, we aimed at answering the following three research questions (RQs):

- ***RQ1 – Fitness Function Assessment:*** *Is the uncertainty metric based on confidence appropriate for the test case selection problem?* Our approach conjectures that the confidence level of a DL model when labeling an input can help increase the cost-effectiveness of DL systems testing by selecting an appropriate subset of follow-up test cases. This RQ aims at validating this hypothesis. Similar to other regression testing studies [17, 28], we modified the effectiveness fitness function with the goal of favoring follow-up test cases whose source test cases showed a high confidence. Therefore, we integrated the NSGA-II algorithm with such modified fitness function. Thorough the rest of the paper, we refer to this version as NSGA-II(RQ1).
- ***RQ2 – Sanity Check:*** *How does the proposed test case selection approach together with NSGA-II perform when compared to Random Search (RS)?* The second RQ aims at answering whether the problem to solve is non-trivial. To this end, we integrated the proposed fitness functions with RS, as done in other studies that evaluate test selection techniques [6, 42].
- ***RQ3 – Improvement Extent:*** *To what extent can NGSA-II together with the proposed fitness functions outperform RS?* The third RQ is defined to quantify the average percentage of improvement of the NSGA-II algorithm along with the proposed fitness functions when compared with RS.

5.1 Experimental Setup

We now explain the experimental setup we carried out to answer the defined RQs.

5.1.1 Models. We used three pre-trained DL models for the object classification problem: (1) *ResNet50*, (2) *ResNet101* and (3) *GoogleNet*. These three models are Convolutional Neural Networks (CNN) of 50, 101 and 22 layers deep respectively. The three of them classify

images into 1,000 object categories and their image input size is of 224-by-224.

5.1.2 Dataset. Spieker and Gotlieb used the CIFAR-10 dataset [41]. However, the authors of the dataset removed it due to a variety of reasons, and asked the community not to use it.[1] Therefore, we used the Imagenette dataset,[2] which is a subset of 10 classified classes from Imagenet [20], a widely used dataset for image classification algorithms. We used a total of 3,845 images from the validation dataset of Imagenette. Table 1 shows the obtained missclassification rates by the defined metamorphic relations for the three models used in our study. As can be seen, MR2 obtained the highest missclassification rate while MR1 showed the lowest one. Despite the dataset being different, these results were in line with the results obtained by Spieker and Gotlieb for such MRs with the CIFAR-10 dataset [41].

Table 1: Failure rate of each of the defined MRs for the 3,845 images and the three DL models used in our evaluation

	MR1	MR2	MR3	MR4	MR5
ResNet50	0.15	0.52	0.30	0.30	0.24
ResNet101	0.12	0.52	0.31	0.31	0.19
GoogleNet	0.24	0.69	0.35	0.36	0.30

5.1.3 Evaluation metrics. As proposed by Panichella et al. [31], and used in other test case selection studies (e.g., [4–6]), we used the revisited hypervolume (HV) metric to measure the effectiveness of our approach. The revisited HV, instead of measuring the HV at the algorithm's objective level, first obtains the Pareto-frontier returned by the algorithm (i.e., in our case NSGA-II or RS) and measures (1) the cost and (2) the effectiveness of each individual solution. With the obtained information, a second Pareto-frontier is obtained aiming to minimize the cost and maximize the effectiveness. In our case, we measured the cost as the test reduction rate provided by the solution (i.e., # of selected test cases/ # of total test cases). Thus, this metric is in the range [0,1], where the lower, the better. As for the effectiveness, we measured the selected failure rate. To this end, we first applied the entire test suite in each MR and obtained the number of MR violations by the entire test suite. In a second step, for each of the solutions, we measured the number of MR violations by the selected test cases in a solution. Therefore, we measured the normalized fault detection rate as # of failures detected by the selected test cases/ # of failures detected by the entire test suite. Thus, this metric is also in the range [0,1], where the higher, the better. It is noteworthy that the revisited HV was applied individually in the five MRs.

5.1.4 Statistical tests. The used search algorithms involve random variations. We therefore run each algorithm 50 times, as recommended by Arcuri and Briand [2]. We kept control over the seed in order our results to be reproducible by other researchers. After executing the experiments, we assessed the normality of the data samples through the Shapiro-Wilk test. Since the data was normally distributed, we used the t-test (i.e., significance test) to determine the significance of the results produced by different algorithms. The significance level was set to 95%, which means that a p-value lower than 0.05 means that there was statistical significance. Furthermore, to assess the difference between the different algorithms, we employed the Vargha and Delaney $\hat{A}_{12}$ value.

5.1.5 Algorithm setup. Similar to other studies where the NSGA-II was the algorithm for test case selection [5, 6, 44], the population was set to 100 and the number of fitness evaluations to 25,000. The crossover rate was set to 0.8. We used a one-point crossover operator. Lastly, the mutation rate was 0.01. Unlike other studies [5, 6], we did not use the probability rate of 1/N, N being number of test cases, because the number of test cases in our study was much larger (i.e., 3,845) than those previous studies (i.e., around 150 [5, 6]). We used the bit-flip mutation operator. Therefore, adding/removing 1 test case on average by the mutation operator does not have a large impact in the search.

5.2 Analysis of the Results and Discussion

In this section we analyze and discuss the obtained results to answer our RQs. Figure 4 depicts the obtained distribution for the revisited HV metric. As can be observed, for the NSGA-II algorithm, the values for MR1 are the highest ones, whereas the MR2 are the lowest ones. Similarly, the values were higher in general for the Resnet 50 and Resnet101 DL models than the Googlenet DL model. This could be due to the failure rates of each model and MR. As can be observed, MR1 has the lowest failure rates for the three DL models, whereas MR2 has the highest ones. On the other hand, GoogleNet showed higher failure rates than the Resnet50 and Resnet101 DL models.

RQ1 – Fitness function assessment: Figure 4 shows the obtained distribution for the revisited *HV* for each MR and each DL model. As can be seen, the results of our approach (NSGA-II) outperformed in all cases the NSGA-II with the fitness function configured to promote the inclusion of follow-up test cases with low uncertainty (i.e., NSGA-II(RQ1)). These results were further corroborated through statistical tests (Table 2), where all the $\hat{A}_{12}$ values were 1, meaning that the probability of our approach being better than the baseline is of 100%. Furthermore, all p-values were very low. This can also be appreciated from the boxplots in Figure 4, where there was no single value of the NSGA-II below the values of the NSGA-II(RQ1). Furthermore, it can be observed that in 12 out of 15 cases (i.e., all except the cases of MR2) RS outperformed the NSGA-II(RQ1). This means that the uncertainty of source test cases measured through the confidence of a DL model to classify an object is a good proxy to select follow-up test cases. Therefore, we can answer the first RQ as follows:

> The uncertainty based on confidence is an appropriate metric to integrate as an objective function of a multi-objective algorithm for selecting follow-up test cases of DL systems when using metamorphic testing.

RQ2 – Sanity check: As sanity check of our study, we compared the NSGA-II algorithm with RS under the same conditions (i.e., number of fitness evaluations and same fitness functions). Usually,

[1] See the following page for the explanations given by the CIFAR-10 authors: http://groups.csail.mit.edu/vision/TinyImages/

[2] https://github.com/fastai/imagenette

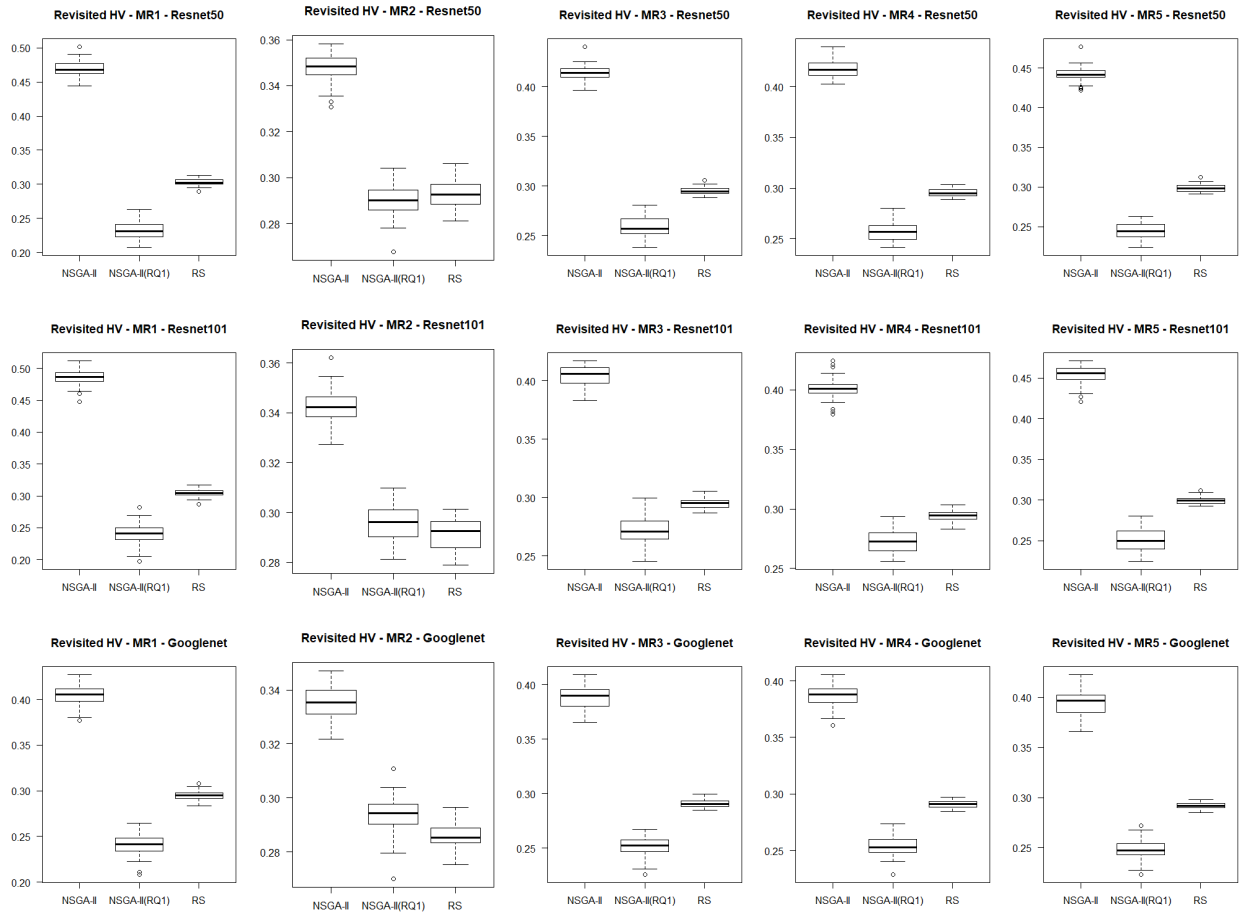

Figure 4: Distribution of the obtained Revisited HV metric for the different algorithms, MRs and DL models

Table 2: Results of the Vargha and Delaney $\hat{A}_{12}$ values and ttest p-values between our approach and the baselines (RQ1 and RQ2)

		MR1		MR2		MR3		MR4		MR5	
		$\hat{A}_{12}$	p-value	$\hat{A}_{12}$	p-value	$\hat{A}_{12}$	p-value	$\hat{A}_{12}$	p-value	$\hat{A}_{12}$	p-value
ResNet50	NSGA-II vs. NSGA-II (RQ1)	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001
	NSGA-II vs. RS	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001
ResNet101	NSGA-II vs. NSGA-II (RQ1)	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001
	NSGA-II vs. RS	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001
GoogleNet	NSGA-II vs. NSGA-II (RQ1)	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001
	NSGA-II vs. RS	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001	1	<0.0001

when RS outperforms NSGA-II, it means that the problem to solve is non-trivial. Another possibility might mean that the fitness functions are not effective. Neither happened in our case. As can be seen in Figure 4 and corroborated by the statistical tests (Table 2), the NSGA-II outperformed RS with statistical significance. Specifically, for all the 50 runs in all MRs and for all the used DL models, the NSGA-II provided higher HV values than RS. Thus, we can answer the second RQ as follows:

> The NSGA-II algorithm outperformed RS in all the cases with statistical significance. This means that the problem of metamorphic follow-up test case selection in the context of DL systems is a non-trivial problem. Therefore, search algorithms are recommended.

RQ3 – Improvement extent: Table 3 shows the percentage improvement extent of the NSGA-II over RS in terms of the revisited HV for the average values. The largest improvements are related to the first MR, whereas the lowest ones are related to MR2. It is noteworthy that this is consistent with the failure rates of the MRs that can be seen in Table 1. MR1 has the lowest failure rates whereas MR2 has the largest ones. A possible explanation could be that in

the case of MR2, the NSGA-II reduces the candidate test suites too much, therefore, missing many failure revelations. A potential improvement to fix this issue could be to seed the initial population of the search, as proposed in a previous work [4]. Nevertheless, the lowest improvement extent is 17.09%, which is still considered quite large. Having discussed this, the third RQ can be answered as follows:

> The improvement extent of NSGA-II over RS is between 17.09 to 59.20%. This improvement extent depends on the failure rate of the MRs. With MRs with lower failure rates (e.g., MR1), the improvement extents are larger when compared with MRs with higher failure rates.

Table 3: Average improvement extent of the NSGA-II algorithm over RS (RQ3)

	MR1	MR2	MR3	MR4	MR5
Resnet50	55.14%	18.55%	39.99%	41.06%	47.98%
Resnet101	59.20%	17.26%	37.03%	36.36%	51.67%
GoogleNet	37.00%	17.09%	32.97%	32.50%	34.97%

5.3 Threats to validity

We now summarize the threats to validity of our evaluation and the actions we took to mitigate such risks:

Internal validity: An internal validity threat in our study could be related to the selected MRs. We selected these MRs based on a previous work [41]. Additional MRs were proposed in the same work. One of them was based on blurring the original image, but the implementation we found took a long time to generate follow-up test cases based on that MR. The inclusion of that MR would require the experimental set-up to be reduced (e.g., fewer DL models). Another one was based on inverting an image, but from the results from Spieker and Gotlieb [41] we found that violating MRs with such a transformation was too trivial. Therefore, we decided not to use these two MRs. Some of these MRs were configurable. For instance, MR3 and MR4 require a configuration of the degrees of the image to be rotated. Initially, we set this value to 20°, but the DL models missclassified most of the images, labeling them as "PC monitor" based on the edges this MR produces in the follow-up test cases. Therefore, we reduced this value to 5° and -5°, removing this problem. Another internal validity threat in our study refers to the parameters of the algorithms, which were not changed. To reduce this threat we configured the algorithms based on previous works that used the NSGA-II algorithm for test case selection purposes [6, 44].

External validity: An external validity threat in our evaluation relates to the generalization of the results. We tried to mitigate this threat by using three different DL models and five MRs for each of the models. Another external validity in all test selection and prioritization studies is related to how the failures are distributed, as found by Zhu et al., [49]. From Table 1, it can be observed that our MRs detected different amount of failures depending on the DL model and the MR. For instance, MR1 showed a low failure rate, whereas MR2 a large one. Meanwhile, the failure rates of MR2, MR3 and MR5 were modest, but also different among the DL models (e.g., the lowest failure rate for MR5 was 0.19 and the largest one 0.3). Therefore, this suggests that in our study we had a diverse distribution of failures.

Conclusion validity: As in any other study involving randomized algorithms, a conclusion validity threat involves the non-determinism of the employed algorithms. We mitigated this threat by running each algorithm 50 times to account for random variations, as recommended in related guidelines [2]. Moreover, we used the appropriate statistical tests to carefully analyze the results and differences among the proposed approach and the baseline techniques.

Construct validity: Construct validity threats arise when the measures used are not comparable across algorithms. To mitigate this threat we used the same stopping criterion for all the algorithms (i.e., the total number of fitness evaluations was set to 25,000).

6 RELATED WORK

Test case selection has been a widely studied technique in the current literature. Yoo et al., identified up to 12 different approaches based on an extensive analysis of the state-of-the-art [46]. Engström et al., surveyed 28 different techniques related to test case selection [14]. The test case selection problem is both (1) complex due to its huge search space and (2) multi-objective in nature [10]. For this reason, Pareto-based search algorithms have been found an appropriate technique to solve the test case selection problem in different domains [6, 31, 34, 42, 44]. Yoo and Harman were the first researchers in proposing the use of Pareto-based search algorithms to solve the test case selection problem [44]. Their original approach involved two effectiveness functions (i.e., coverage and historical information of faults) and a cost function (i.e., test execution cost). The same authors extended this study by proposing an hybrid approach [45].

Since then, the application of multi-objective test case selection algorithms has been widely studied in multiple domains (e.g., defense [15], maritime [33]) and type of systems (e.g., software product lines [42], cyber-physical systems [6]). Different types of studies are performed. Most of them focus on proposing cost-effective objective functions and study whether they act as reasonable surrogate to detect faults [6, 44]. Other studies focus on comparing different types of search algorithms [33, 34, 42]. There is a last group that proposes novel strategies inside the algorithms and their genetic operators, such as injecting diversity through the search [31], proposing novel crossover operators [30], or seeding the initial population of the algorithm [4]. However, to the best of our knowledge, there is no previous approach that has proposed multi-objective test case selection approach neither in the domain of DL systems nor in the context of metamorphic testing.

Research efforts on testing DL systems has considerably increased in the last few years from different perspectives [36, 47]. This research spans across different areas [36, 47], such as, new test adequacy criteria for DL systems testing [18, 19, 23–25, 32], test generation approaches [23, 37] and the comparison between on-line and off-line DL system testing [16]. A recent survey identified the test oracle problem as one of the fundamental challenges when testing ML systems [47]. Metamorphic oracles, which are the

types of oracles we target in this paper, have been found widely applied to solve the oracle problem in machine-learning systems [36]. For instance, Xie et al., [43] investigated the application of metamorphic testing to test unsupervised ML systems. Ding et al., [11] proposed an approach for validating the classification accuracy of a DL framework that included a CNN, a DL execution environment, and a massive image data set. Murphy et al., [29] proposed metamorphic testing to test different ML algorithms (e.g., Support Vector Machines). Saha and Kanewala compared multiple MRs from previous studies to test supervised classifiers [38]. Dwarakanath et al., proposed a set of MRs, which included rotation of images, to test image classifiers [13]. All these works explore the applicability of metamorphic testing to test machine learning and DL applications. However, unlike this study, their focus is not on increasing the cost-effectiveness of metamorphic testing by selecting an appropriate subset of test cases.

In the context of DL systems testing, similar to our approach, two recent studies cover both the execution cost and the test oracle problem. In particular, Ma et al., [26] investigate the correlation between different metrics and misclassification on both real and artificial data. These metrics involve both uncertainty-based and coverage-based metrics. They found that uncertainty-based metrics had medium to strong correlation with misclassification whereas coverage-based techniques showed weak or no correlation. On the other hand, Ma et al., [27] proposed a differential testing approach for selecting test cases. Specifically, they train different sub-specialized model instances of the DL system by training these models with sub-data of different categories. These sub-specialized models are later used for differential testing, i.e., the same test inputs are applied for both the sub-specialized models and the DL model under test. If one of the sub-specialized models differs with the DL model under test, that test input is selected to be manually inspected by the developer. Several differences exists between these two approaches and ours. Firstly, in both of these papers, the misclassification is intended to be checked manually, whereas we use metamorphic testing to provide a test verdict, making our approach fully automated. Secondly, they do not propose any search algorithm to select potential test cases. The first paper [26] limits their study to measuring the correlation between different metrics and misclassification. The second paper [27] uses multiple test executions with different DL models, which also increases the test execution cost. Conversely, in our study, we propose a multi-objective test case selection approach to select follow-up test cases by considering the uncertainty provoked by the source test cases in the DL model.

7 CONCLUSION AND FUTURE WORK

Metamorphic testing alleviates the test oracle problem of several DL applications. However, this technique requires the execution of the source and follow-up test cases, requiring additional testing cost. In this paper, we propose a novel approach for increasing the cost-effectiveness of DL systems testing when using metamorphic tests. The approach uses information of the uncertainty provoked by the source test cases in the DL model to select follow-up test cases. The selection algorithm employs two competing fitness functions that are integrated with NSGA-II. We evaluated our approach with three different DL models intended to solve the image classification problem. We also used five different metamorphic relations that were defined in a previous work [41]. The results suggest that our approach is appropriate for selecting test cases of metamorphic follow-up test cases for testing DL systems.

There are different research avenues we would like to explore in the close future. One of them relates to the selection of multiple follow-up test cases. Our current version assumes there is only one follow-up test case per source test case (or that all follow-up test cases of a source test case will be selected and applied into multiple MRs). In the future we would like to implement an approach such that the algorithm could favor the selection of one or more follow-up test case per source test case. For instance, a source test case with a very high uncertainty might need all its follow-up test cases to be selected whereas a source test case with a lower uncertainty might need the selection of less MRs' follow-up test cases. Another research line we would like to explore is the evaluation of the technique in other DL applications, such as, DL for natural language processing or object identification. Lastly, a key drawback of the approach is that it only selects follow-up test cases, as it requires information on the execution of source test cases to measure their uncertainty. Future work includes the investigation of techniques to select source test cases too.

REPLICATION PACKAGE

The full replication package is available at Zenodo: https://doi.org/10.5281/zenodo.6389008

ACKNOWLEDGMENTS

This publication is part of a project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 871319.

REFERENCES

[1] John Ahlgren, Maria Eugenia Berezin, Kinga Bojarczuk, Elena Dulskyte, Inna Dvortsova, Johann George, Natalija Gucevska, Mark Harman, Maria Lomeli, Erik Meijer, Silvia Sapora, and Justin Spahr-Summers. 2021. Testing Web Enabled Simulation at Scale Using Metamorphic Testing. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. 140–149. https://doi.org/10.1109/ICSE-SEIP52600.2021.00023

[2] Andrea Arcuri and Lionel Briand. 2011. A practical guide for using statistical tests to assess randomized algorithms in software engineering. In *2011 33rd International Conference on Software Engineering (ICSE)*. IEEE, 1–10.

[3] Aitor Arrieta. 2022. On the Cost-Effectiveness of Composite Metamorphic Relations for Testing Deep Learning Systems. In *Metamorphic Testing Workshop (MET'22)*. ACM.

[4] Aitor Arrieta, Joseba Andoni Agirre, and Goiuria Sagardui. 2020. Seeding strategies for multi-objective test case selection: an application on simulation-based testing. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*. 1222–1231.

[5] Aitor Arrieta, Shuai Wang, Ainhoa Arruabarrena, Urtzi Markiegi, Goiuria Sagardui, and Leire Etxeberria. 2018. Multi-objective black-box test case selection for cost-effectively testing simulation models. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 1411–1418.

[6] Aitor Arrieta, Shuai Wang, Urtzi Markiegi, Ainhoa Arruabarrena, Leire Etxeberria, and Goiuria Sagardui. 2019. Pareto efficient multi-objective black-box test case selection for simulation-based testing. *Information and Software Technology* 114 (2019), 137–154.

[7] Jon Ayerdi, Valerio Terragni, Aitor Arrieta, Paolo Tonella, Goiuria Sagardui, and Maite Arratibel. 2021. Generating metamorphic relations for cyber-physical systems with genetic programming: an industrial case study. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 1264–1274.

[8] T. Y. Chen, S. C. Cheung, and S. M. Yiu. 1998. *Metamorphic Testing: A New Approach for Generating Next Test Cases.* Technical Report. Technical Report HKUST-CS98-01, Department of Computer Science, The Hong Kong University of Science and Technology.

[9] Tsong Yueh Chen, Fei-Ching Kuo, Huai Liu, Pak-Lok Poon, Dave Towey, T. H. Tse, and Zhi Quan Zhou. 2018. Metamorphic Testing: A Review of Challenges and Opportunities. *ACM Comput. Surv.* 51, 1, Article 4 (Jan. 2018), 27 pages. https://doi.org/10.1145/3143561

[10] Yiqun T Chen, Rahul Gopinath, Anita Tadakamalla, Michael D Ernst, Reid Holmes, Gordon Fraser, Paul Ammann, and René Just. 2020. Revisiting the relationship between fault detection, test adequacy criteria, and test set size. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*. 237–249.

[11] Junhua Ding, Xiaojun Kang, and Xin-Hua Hu. 2017. Validating a deep learning framework by metamorphic testing. In *2017 IEEE/ACM 2nd International Workshop on Metamorphic Testing (MET)*. IEEE, 28–34.

[12] Alastair F. Donaldson. 2019. Metamorphic Testing of Android Graphics Drivers. In *Proceedings of the 4th International Workshop on Metamorphic Testing* (Montreal, Quebec, Canada) *(MET '19)*. IEEE Press, 1. https://doi.org/10.1109/met.2019.00008

[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] Emelie Engström, Per Runeson, and Mats Skoglund. 2010. A systematic review on regression test selection techniques. *Information and Software Technology* 52, 1 (2010), 14–30.

[15] Vahid Garousi, Ramazan Özkan, and Aysu Betin-Can. 2018. Multi-objective regression test selection in practice: An empirical study in the defense software industry. *Information and Software Technology* 103 (2018), 40–54.

[16] Fitash Ul Haq, Donghwan Shin, Shiva Nejati, and Lionel C Briand. 2020. Comparing offline and online testing of deep neural networks: An autonomous car case study. In *2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST)*. IEEE, 85–95.

[17] Christopher Henard, Mike Papadakis, Mark Harman, Yue Jia, and Yves Le Traon. 2016. Comparing white-box and black-box test prioritization. In *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*. IEEE, 523–534.

[18] Nargiz Humbatova, Gunel Jahangirova, and Paolo Tonella. 2021. Deepcrime: Mutation testing of deep learning systems based on real faults. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*. 67–78.

[19] 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.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012), 1097–1105.

[21] Dong Guowei Xu Baowen Chen Lin and Nie Changhai Wang Lulu. 2008. Case studies on testing with compositional metamorphic relations. *Journal of Southeast University (English Edition)* 4 (2008).

[22] Huai Liu, Xuan Liu, and Tsong Yueh Chen. 2012. A new method for constructing metamorphic relations. In *2012 12th International Conference on Quality Software*. IEEE, 59–68.

[23] Lei Ma, Felix Juefei-Xu, Minhui Xue, Bo Li, Li Li, Yang Liu, and Jianjun Zhao. 2019. Deepct: Tomographic combinatorial testing for deep learning systems. In *2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 614–618.

[24] 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.

[25] Lei Ma, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Felix Juefei-Xu, Chao Xie, Li Li, Yang Liu, Jianjun Zhao, et al. 2018. Deepmutation: Mutation testing of deep learning systems. In *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 100–111.

[26] 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.

[27] Yu-Seung Ma, Shin Yoo, and Taeho Kim. 2021. Selecting test inputs for DNNs using differential testing with subspecialized model instances. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 1467–1470.

[28] Urtzi Markiegi, Aitor Arrieta, Leire Etxeberria, and Goiuria Sagardui. 2021. Dynamic test prioritization of product lines: An application on configurable simulation models. *Software Quality Journal* 29, 4 (2021), 943–988.

[29] Christian Murphy, Kuang Shen, and Gail Kaiser. 2009. Automatic system testing of programs without test oracles. In *Proceedings of the eighteenth international symposium on Software testing and analysis*. 189–200.

[30] Mitchell Olsthoorn and Annibale Panichella. 2021. Multi-objective test case selection through linkage learning-based crossover. In *International Symposium on Search Based Software Engineering*. Springer, 87–102.

[31] Annibale Panichella, Rocco Oliveto, Massimiliano Di Penta, and Andrea De Lucia. 2014. Improving multi-objective test case selection by injecting diversity in genetic algorithms. *IEEE Transactions on Software Engineering* 41, 4 (2014), 358–383.

[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] Dipesh Pradhan, Shuai Wang, Shaukat Ali, and Tao Yue. 2016. Search-based cost-effective test case selection within a time budget: An empirical study. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. 1085–1092.

[34] Dipesh Pradhan, Shuai Wang, Shaukat Ali, Tao Yue, and Marius Liaaen. 2017. CBGA-ES: A cluster-based genetic algorithm with elitist selection for supporting multi-objective test optimization. In *2017 IEEE International Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 367–378.

[35] Kun Qiu, Zheng Zheng, Tsong Chen, and Pak-Lok Poon. 2020. Theoretical and Empirical Analyses of the Effectiveness of Metamorphic Relation Composition. *IEEE Transactions on Software Engineering* (2020).

[36] 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, 6 (2020), 5193–5254.

[37] Vincenzo Riccio and Paolo Tonella. 2020. Model-based exploration of the frontier of behaviours for deep learning system testing. In *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 876–888.

[38] Prashanta Saha and Upulee Kanewala. 2019. Fault detection effectiveness of metamorphic relations developed for testing supervised classifiers. In *2019 IEEE International conference on artificial intelligence testing (AITest)*. IEEE, 157–164.

[39] S. Segura, G. Fraser, A. Sanchez, and A. Ruiz-Cortés. 2016. A Survey on Metamorphic Testing. *IEEE Transactions on Software Engineering* 42, 9 (Sept 2016), 805–824. https://doi.org/10.1109/TSE.2016.2532875

[40] S. Segura, D. Towey, Z.Q. Zhou, and T.Y. Chen. 2020. Metamorphic Testing: Testing the Untestable. *IEEE Software* 37, 3 (2020), 46–53.

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

[42] Shuai Wang, Shaukat Ali, and Arnaud Gotlieb. 2015. Cost-effective test suite minimization in product lines using search techniques. *Journal of Systems and Software* 103 (2015), 370–391.

[43] Xiaoyuan Xie, Zhiyi Zhang, Tsong Yueh Chen, Yang Liu, Pak-Lok Poon, and Baowen Xu. 2020. METTLE: a METamorphic testing approach to assessing and validating unsupervised machine LEarning systems. *IEEE Transactions on Reliability* 69, 4 (2020), 1293–1322.

[44] Shin Yoo and Mark Harman. 2007. Pareto efficient multi-objective test case selection. In *Proceedings of the 2007 international symposium on Software testing and analysis*. 140–150.

[45] Shin Yoo and Mark Harman. 2010. Using hybrid algorithm for pareto efficient multi-objective test suite minimisation. *Journal of Systems and Software* 83, 4 (2010), 689–701.

[46] Shin Yoo and Mark Harman. 2012. Regression testing minimization, selection and prioritization: a survey. *Software testing, verification and reliability* 22, 2 (2012), 67–120.

[47] Jie M Zhang, Mark Harman, Lei Ma, and Yang Liu. 2020. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering* (2020).

[48] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. 2018. DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems. In *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 132–142.

[49] Yuecai Zhu, Emad Shihab, and Peter C Rigby. 2018. Test re-prioritization in continuous testing environments. In *2018 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 69–79.

[50] Tahereh Zohdinasab, Vincenzo Riccio, Alessio Gambi, and Paolo Tonella. 2021. Deephyperion: exploring the feature space of deep learning-based systems through illumination search. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*. 79–90.