Towards Robust Object Detection Models by Metamorphic Testing

JIANHONG WANG[1,a)] YUTA ISHIMOTO[1,b)] MASANARI KONDO [1,c)] YASUTAKA KAMEI[1,d)]
NAOYASU UBAYASHI[1,e)]

Abstract: Object detection models are widely used in safety-critical systems in industrial fields such as autonomous driving. It is essential to improve the robustness of object detection models, which represents the capability of avoiding failures under realistic scenarios. It is difficult to evaluate and improve the robustness of most current object detection models. This research proposes an approach to evaluate and improve their robustness by metamorphic testing. More specifically, we use the following metamorphic relation: even if inserting new instances into the original image, the object detection model behaves the same before and after the insertion. Bayesian uncertainty and the failure rate are used as evaluation criteria for the robustness. The experiment uses a large-scale dataset, the COCO dataset, with over 160 thousand images. Our proposed approach improves the robustness by reducing the uncertainty of the data by 12.8% and the failure rate by 26.3% on average.

1. Introduction

The object detection system is an important application system of computer vision in recent years. Object detection systems using models such as convolutional neural networks or transformers as the framework are widely used in various industrial fields. The dynamic tracking of autonomous driving [1] requires the usage of object detection techniques to assist in determining road conditions. Intelligent medical image analysis [2] uses object detection to pinpoint the location of cancerous cells. Object detection systems play an essential role in those applications in many industrial fields.

However, object detection systems still face severe issues regarding reliability and safety. In real-world scenarios, it has been found that object detection systems are susceptible to small disturbances triggering failures [3]. For instance, operational autopilot systems often overlook or misjudge sudden flying objects (birds), resulting in temporary system failure. These failures are fatal and unacceptable. A robust object detection system needs to avoid failures as much as possible, and it can be achieved by improving the robustness of the model, which is pivotal in the object detection system.

Software testing can be used to verify the robustness of the object detection model. Wang, S. and Su, Z [4] applied metamorphic testing, a software testing technique to object detectors. Metamorphic testing of object detection is proposed to test the performance of a model in a real production environment: the original data image interfered by inserting an instance similar to the background, is transformed into metamorphic testing, which verifies the model maintains the predictability. However, existing researches have not discussed how metamorphic testing is used to improve the model robustness.

The motivation of this research is to evaluate and improve the robustness of the object detection model using the metamorphic testing framework. The robustness of the model is reflected in the model's performance against realistic attacks under the production environment. In our research, metamorphic testing mainly uses Bayesian uncertainty [5] (how much confidence the model itself has in the prediction results) and failure rate (how often each image fails during metamorphic testing). A robust model shall have lower uncertainty and failure rate.

More specifically, metamorphic testing evaluates the robustness by asserting the metamorphic relations. A metamorphic relation is established when the model still keeps the same prediction results before and after the tested images are inserted by extracted instances. In the experiments, the tested images and extracted instances are selected by their different characteristics from the whole dataset, COCO dataset [6], with over 160 thousand images. Then, metamorphic images are synthesized by pasting the extracted instance onto the tested images after determining the specified location for pasting, which as much as possible challenges the original model robustness. During the metamorphic testing, models are tested by predicting both the original tested images and their metamorphic ones in order to see whether the model maintains the metamorphic relations or not. Metamorphic images that become unpredictable and break the metamorphic relations are called failed test cases. After the metamorphic testing, those failed test cases are used to create a new training dataset to retrain the model in order to improve its robustness.

2. Testing object detection models

2.1 Object detection model

Current object detection combines deep learning models with traditional image processing technology. Deep learning system is

[1] Kyushu University
[a)] wang@posl.ait.kyushu-u.ac.jp
[b)] ishimoto@posl.ait.kyushu-u.a
[c)] kondo@posl.ait.kyushu-u.ac
[d)] kamei@posl.ait.kyushu-u.ac
[e)] ubayashi@ait.kyushu-u.ac

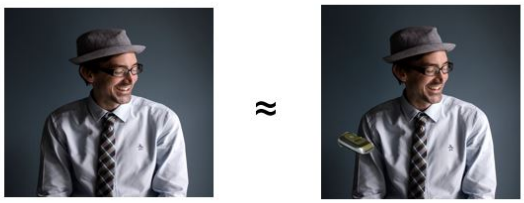

Fig. 1 Synthesizing images by pasting an inserting instance. The left is the original image. The right with the cellphone pasted is the metamorphic image.

the embodiment of advanced technology and the key to intelligent detection. An object detection system detects the location, contour, classes, and semantic information of various instances in an image. The most used are the classes and locations of the instances. To evaluate the quality of an object detection system, it is necessary to cover the testing of many aspects which are related to the prediction results. Our research mainly concentrates on the robustness of object detection models.

2.2 Testing Robustness of deep learning models

Model robustness refers to how much model performance varies when using new data. Model robustness ensures the reliability and safety of deep learning systems [7]. To ensure that a model is working as expected, it's critical to monitor and manage robustness. Therefore, almost all applications and studies of deep learning models require the testing of robustness. Currently, robustness testing is mainly conducted based on adversarial samples. One of the most effective robustness testing techniques [8] is to generate adversarial samples by generating algorithms and retrain them. The robustness testing based on adversarial samples allows the model to change from a native state to a defensive state. For testing the robustness of the object detection models under a realistic environment, we decide to adopt the method of metamorphic testing for our research, which simulates the realistic attack like in a production environment.

2.3 Metamorphic testing

Metamorphic testing is a property-based software testing technique [10]. In the field of machine learning, metamorphic testing describes and evaluates system functionality through metamorphic relations. A metamorphic relation for executing a system ensures that the output keeps the consistency in terms of the inputs and metamorphic inputs. In our research, we discuss the consistency of predictions for object detection models:

- class consistency: The prediction before and after the insertions should be consistent.
- localization consistency: The predicted bounding box (the predicted rectangle containing the instance) should remain similar shapes and coordinates before and after the insertions.
- recognition consistency: The consistency of whether the model still retains the ability of successful prediction.

The above are the consistencies for all aspects of the prediction of an object detection model. The metamorphic relation is kept if all three consistencies are kept, otherwise, it is broken. When the metamorphic relations of predictions are broken, the model robustness is challenged. Metamorphic testing was used to evaluate the performance of object detectors [4]. They explored the technique of instance segmentation to create a data pool of instances after extracting high-quality object instances from the images, as shown in Figure 1 [*1]. These instances are inserted into the original images and combined with the original images to synthesize new images. Metamorphic relations between the model predictions are established: all model predictions for the *metamorphic images* (images synthesized by pasting instances onto the original image during metamorphic testing) and the original image (the raw images without the insertion) should remain equivalent. They use delta-debugging styled insertion to find locations where the metamorphic relations are broken. Delta-debugging styled insertions is constantly bringing the instances closer to the base instance's bounding box boundaries to be detected.

The advantage of metamorphic testing is that it reveals the vulnerabilities of the object detector against interference under a production environment. This is more relevant in reality than other deep learning techniques such as FGSM [11]. In our research, we further improve the delta-debugging styled insertions by Grad-CAM [14], considering that many instances are of special shapes and not proper for being a reference point merely by their boundaries. Moreover, to evaluate the model robustness regarding a single image but not the accuracy of the whole dataset, we mainly use two metrics as the evaluation criteria, Bayesian uncertainty, which is introduced in the next subsection, and the failure rate, which is introduced in Section 3.4.

2.4 Bayesian uncertainty

Bayesian uncertainty [5] originates from Bayesian neural networks and it is to assess the confidence of the model prediction. Bayesian neural networks convert the determined parameters and predicted outcomes into corresponding probability distributions, where the variance of these distributions can be called Bayesian uncertainty. To simplify the complexity of deriving Bayesian uncertainty, the MC Dropout [12] treats the variance in the probability distribution of the final prediction outcome as uncertainty. During the model training process, variance is similarly embedded in the loss function for training.

Bayesian uncertainty is applied to object detection models [5]. Bayesian uncertainty in object detection includes class prediction uncertainty and bounding box uncertainty. This study uses a probabilistic model of object detection to enhance the training effect, improve the overall accuracy of the bounding box prediction, and obtain fewer incorrect predictions of class. In our study, we use MC Dropout to rewrite the YOLO [9] into Bayesian YOLO to derive uncertainty and use uncertainty as one of the evaluation metrics.

3. Our approach

3.1 Overview

In this section, the methods and the mechanism of metamorphic testing to evaluate the robustness of object detection models are introduced.

We introduce the approaches of conducting metamorphic testing on the object detection model in the order of preparation of data, localization of insertion instances, and designed metamorphic testing workflow in the following subsections. As shown in Figure 2, metamorphic testing is performing the prediction of a series of *data characteristics* (characteristics of data that can be explored after the prediction) and detecting the failed test cases from the prediction bases, which record the data characteristics of both successful test cases and failed test cases. In our research, through metamorphic testing of object detection models, we can analyze what data

[*1] © Attribution-NonCommercial 2.0 Generic (CC BY-NC 2.0)

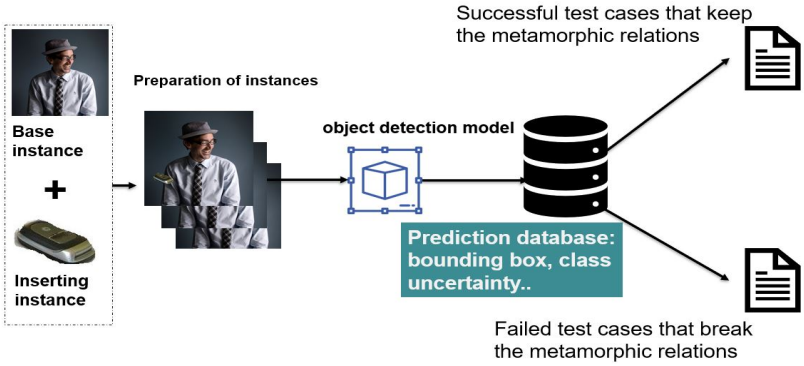

Fig. 2 The metamorphic testing workflow: generation of databases of successful test cases and failed test cases.

characteristics influence the robustness of models by evaluating the difference of the prediction uncertainty and failure rate. Moreover, we can retrain the failed test cases to improve the robustness of the object detection model.

3.2 Preparation of data

Data in our study are the instances on the image data. In this study, we use two types of instances to synthesize metamorphic images. Instances are classified as the instances to be tested and the instances to be inserted. Instances to be tested are called *base instances*. In all experiments, the model robustness is evaluated by observing the uncertainty and testing failure rate of the base instance, which are described later in the section. On the other hand, the inserted instances are called *inserting instances*, which are extracted from all images in the dataset by the Bayesian instance segmentation model YOLACT and pasted onto the background containing the base instance. Like in the Fig 1, the man is the base instance, and the cellphone placed near the man is the inserting instance. For the inserting instances, we adopt the following criteria to extract high-quality inserting instances for each base instance used for metamorphic testing:

- Instance size: It is necessary to control the size of the inserting instance compared to the base instance.
- Uncertainty: We use the Bayesian YOLACT [15] to derive the prediction uncertainty for all instances. For the following experiments, instances of different values of bounding box uncertainty and class uncertainty are used in metamorphic testing to evaluate the model.
- Scores of the instances (related to the ground truth value): Scores describe the accuracy of the prediction. Scores of the instances also reflect the correctness of the model's prediction on the instances w.r.t the ground truth. Similar to the uncertainty, the instances of different scores are to be used in metamorphic testing.
- Semantic similarity: Keeping the similarity of the image background and the inserting instance is to simulate realistic scenarios. We use phash [13], an image hashing algorithm.
- No occlusions: After recording the coordinates of base instance and inserting the instance, the instances should be prevented from occluding with each other as much as possible.

For each base instance, the inserting instances for it are filtered and chosen by the above criteria, thus each base instance's inserting instances are chosen accordingly. Besides, the uncertainty and scores of instances are considered as the data characteristics used for research questions, as described in Section 4.5.

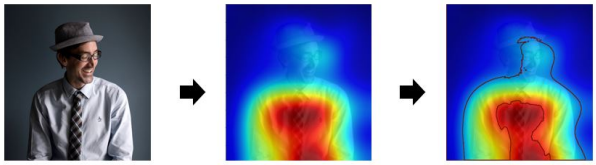

Fig. 3 Generation of Grad-CAM heatmap's contours.

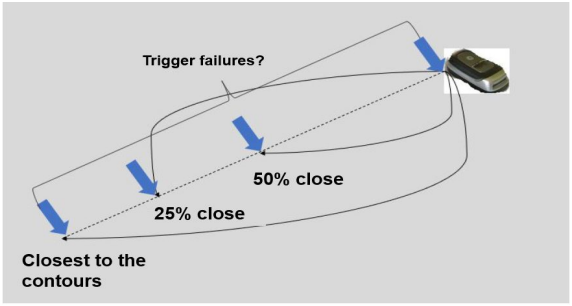

Fig. 4 Delta-debugging styled insertions. For instance, 25% close shows the instances, 25% of the preset distances close to the contours.

3.3 Localization of inserting instances

We adopt a method of delta-debugging styled insertion to determine the locations of inserting instances as the research [4]: gradually moving the inserting instances closer to the base instance from a distance. In our study, we note the characteristics of the object detection models, where the CNN-based model prediction has confidence for each pixel of the image, and these confidences are aggregated to the image as a Grad-CAM heatmap [14]. Delta-debugging styled insertions based on the Grad-CAM better utilize these characteristics of the object detection model, as the previous research determines the locations of insertions merely based on the bounding box, which is not accurate in many cases (e.g. triangular-shaped instances). As shown in Figure 3, the warm-colored areas of the heat map have a high influence on predictions, while the cool-colored areas represent no predictions at all. We use the edges of the warm regions as contours, and then sample a number of points on the contours. There is a point among those sampled points with the closest distance to the inserting instance, and that distance is the distance between the contour and inserting instances. As shown in Figure 4, we use delta-debugging styled insertions to gradually shift the instances closer to the contours. With the distance between inserting instances and the contour halved, the metamorphic testing locates the same inserting instances on the metamorphic images and detect failures in a way of binary search.

3.4 Metamorphic testing workflow

In this section, the workflow of metamorphic testing is introduced.
(1) Prepare a data pool of instances including base instance and inserting instance after synthesizing the images, and their corresponding data characteristics (categories, coordinates).
(2) For each base instances to be tested, the metamorphic testing generates hundreds of metamorphic synthesized images, where the insertions are done in a delta-debugging styled way on different locations on the background of the images. Therefore, failure rate F_r is used to evaluate how frequently the metamorphic testing for one image fails, and it is one of the evaluation criteria of the model robustness.

$$F_r = \frac{the\ number\ of\ failures}{preset\ total\ number\ of\ metamorphic\ testing\ on\ one\ image}$$

(3) Use the object detection model to predict all synthesized images and record the predictions (Bayesian uncertainty, scores...) into the prediction databases. Then, compare the difference with the predicted result of the corresponding original tested image.
(4) The images that are successfully predicted and maintain the metamorphic relations [4] are classified as successful test cases, while the images that fail to be predicted or break the metamorphic relations are classified as failed test cases.
(5) By analyzing the variation of the uncertainty and the failure rate, metamorphic testing can be used to evaluate the robustness of the model according to data with different characteristics. Bayesian uncertainty Unc describes whether the model has consistent confidence in predicting the same base instance. Failure rate F_r shows the robustness of the object detection model against realistic attacks.

4. Experimental Setups

In this section, the experiment settings and two research questions are introduced.

4.1 Models in use

We use an instance segmentation model to extract the instances in the first step of metamorphic testing and another object detection model to conduct the metamorphic testing.
Instance segmentation model: YOLACT, the model used to extract the instances as described in Section 3.2.
Training and Prediction model: Bayesian YOLO model, the model used to be trained, retrained, and derive the uncertainty and failure rate in metamorphic testing. This model is also to be retrained in RQ2.

4.2 Data in use

Data in our research refer to instances of two types: base instances and inserting instances. At the beginning of metamorphic testing, we prepare the base instance to be tested (test cases). Then, We use the criteria as described in Section 3.2 to choose inserting instances for them. All instances are from the COCO dataset [6].

For each research question, we use different base instances (test cases), which are introduced in the later sections. Since the exact configurations in experiments for choosing the inserting instances for base instances are different and of little importance for the research questions, it is omitted to describe here the exact figures of configuration on how to choose the inserting instances. The experiments are conducted on Google Colaboratory using PyTorch.

4.3 Prediction databases in metamorphic testing

After the metamorphic testing, the prediction databases, like in Figure 5, record the data characteristics such as categories, uncertainty, scores, and so forth. Prediction databases record the following information of data:
(1) Firstly, we select the base instance whose scores are over the recognizable threshold (instances' scores greater than 0.7). Since the test cases need clear baselines to evaluate the model robustness in metamorphic testing, we only use the instances which are steadily detected by model with the scores over 0.70. Instances, whose scores is below the recognizable threshold, can not be steadily detected and provide the baselines for metamorphic testing. Hence, there are a total of 183361 instances that can be regarded as tested images.
(2) Then, for each base instance, we select its proper inserting instances from the instance pool by the criteria described in Section 3.2. For each base instance, there are variable numbers of available inserting instances for them to synthesize into a metamorphic image.
(3) Thirdly, we use delta-debugging styled insertions, which determine the locations of different distances to the contour of base instances' Grad-CAM heatmap central region. Delta-debugging styled insertions [4] is moving the inserting instances from a very remote place to the contour, with each time the distances halved (in a manner like 1, 50%, 25%...). We set a series of levels like 0, 12.5%, 25%, 50%, and 100% distances, like in Figure 4, to the contour as level 1, 2, 3, 4, 5... In short, these levels represent the closeness of insertion locations to the contour. When testing a base instance, we conduct 100 times of synthesizing new metamorphic images for each level, and there are totally hundreds of times of metamorphic testing on one single image. After that, we record the number of failure occurrences and divide it by the preset total number (e.g. 500 times) of metamorphic testing to calculate the failure rate F_r.

The prediction databases record the data characteristics of all these metamorphic synthesized images. They are used in both research questions.

4.4 Metrics in use

For most experiments, we evaluate the model robustness by Bayesian uncertainty Unc, which is introduced in Section 2.4, and failure rate F_r, which is introduced in Section 3.4.

4.5 Research questions

4.5.1 RQ1. What are the main data characteristics that influence the metamorphic testing results of the model.

The objective of this research question is to find whether certain characteristics in metamorphic testing, including uncertainty and scores, affect the test results of metamorphic testing. Since the failure rate is an indication of the model robustness, we use the failure rate as a benchmark to find the most relevant data characteristics. For testing instances with different data characteristics, the metamorphic relation of model's predictions might be kept or broken. The model shows different robustness against data with different characteristics, and it deserves investigation of which data characteristics are more influential.

4.5.2 RQ2. How to improve the robustness of the model against realistic insertion attack after data augmentation with failed test cases found in metamorphic testing.

This research question aims to reorganize the training dataset by

Table 1 Average instances' uncertainty and scores of each preset levels. *The intervals* show the equivalent numerical intervals divided from the numerical interval from the highest value and the lowest values. *Collected number* are the numbers of instances for each levels of uncertainty or scores.

The intervals	Average uncertainty	Collected number
all instances	2.214	183,361
0% - 20%	1.435	66,872
20% - 40%	1.973	51,634
40% - 60%	2.305	32,308
60% - 80%	2.456	27,860
80% - 100%	2.720	4,687
The intervals	**Average scores**	**Collected number**
all instances	0.9075	183,361
0% - 20%	0.7623	9,901
20% - 40%	0.8328	23,984
40% - 60%	0.8990	44,355
60% - 80%	0.9251	59,720
80% - 100%	0.9632	45,401

data augmentation with metamorphic images and retrain the model to improve its robustness of the model in a production environment. Also in this research question, we focus on the usage of the data characteristics in RQ1 to improve the effectiveness of retraining in terms of repairing existing failures and reducing the occurrence of new failures which do not occur in the metamorphic testing for the original prediction model, which is called *overfitting*.

5. Experimental Results

In this section, the approaches and results of the experiments for each research questions are described.

5.1 RQ1. What are the main data characteristics that influence the metamorphic testing results of the model.

In this section, the influence of different data characteristics including uncertainty and score is investigated. Uncertainty is the property of the model that describes the model's confidence in the prediction. The score represents the accuracy of the model's prediction compared with the ground truth values. Through experiments, it deserves investigation on which is more influential to model robustness by counting the occurrence of failures in metamorphic testing.

5.1.1 Approach

We analyze the relationship between the failure rate and two data characteristics (uncertainty and scores) of two types of instances (base instances and inserting instances) separately. For each experiment, we synthesize metamorphic images (each image contains one base instance and several inserting instances) by controlling the base instances of different data characteristics. Then, we conduct the metamorphic testing on those different metamorphic images to be tested. The workflow of RQ1 is almost the same as the workflow introduced in Section 3, as shown in Figure 5.

1. Preparation of instances: Before the metamorphic testing, we prepare the instance pool for base instances and their available inserting instances. For instance, if performing the experiments on low uncertain base instances, we only synthesize metamorphic images with the low uncertain base instances. Then, we perform delta-debugging styled insertions to determine the insertion location. Finally, we synthesize the metamorphic images from the chosen base instances and insert instances to form a new dataset.

2. Prediction and testing: We feed the new dataset, which consists of the chosen metamorphic images and the original tested images, to the object detection model. In the meantime, their derived uncertainty, score, and other information are recorded in the prediction database.

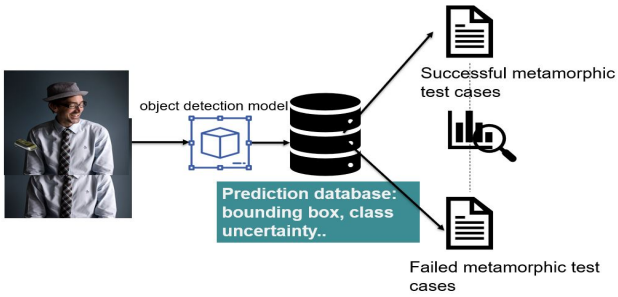

Fig. 5 RQ1: experiment workflow.

Lowest scores — Highest scores

Lowest uncertainty — Highest uncertainty

Uncertainty\Score	0-20%	20%-40%	40-60%	60%-80%	80%-100%
0-20%	0.3675	0.2712	0.2478	0.1854	0.1446
20%-40%	0.2581	0.2336	0.2045	0.1832	0.125
40%-60%	0.225	0.2653	0.1729	0.1564	0.05
60%-80%	0.1935	0.2431	0.2167	0.1	0
80%-100%	0.2	0.33	0.1415	0(no data)	0(no data)

Fig. 6 RQ1 results of 25 combinations of base instances and inserting instances. The numbers inside the table are the failure rate during the metamorphic testing. Each row represents the failure rate of instances of different numerical intervals of uncertainty. Each column represents that of different numerical intervals of scores. In fact, the amount of the high uncertain data and low scored data is small so that there is no data with both high uncertainty and high scores, and it is acceptable to see some unexpected failure rate for that parts of data.

3. Comparative analysis: After obtaining the prediction databases of metamorphic images and the original tested images, we can analyze the influence of the data characteristics on the metamorphic relations of prediction. To evaluate the robustness, we use the failure rate F_r as the criteria and compare the difference of the successful metamorphic test cases and failed ones in terms of their data characteristics. The workflow is demonstrated in Figure 5.

5.1.2 Results and analysis

For all base instances to be tested, we collect the uncertainty (both bounding box uncertainty and classification uncertainty) and scores. As we find in the preliminary experiments that classification uncertainty has a trivial variation of its value after the realistic insertions, we regard two kinds of uncertainty as one uncertainty by adding them together. For base instances' uncertainty or scores, we set five levels of uncertainty or scores to separate all instances, which represents the numerical intervals of their values. Five levels of same-sized numerical intervals are divided from the numerical interval between the lowest uncertainty and highest uncertainty. For instance, as Table 1 shows, 0% - 20% of uncertainty represents the instances that have the lowest uncertainty (highest confident predictions from the model). 0% - 20% of scores represent the instances that have the lowest accuracy compared to the ground truth values.

We use the two characteristics in experiments: There are 25 combinations of levels for the instances. We experiment with all 25 combinations of uncertainty and scores of the base instances and collect the results of their failure rate F_r during metamorphic testing, through which we can evaluate which characteristic is more influential to the metamorphic relations of model robustness.

In Figure 6, there is an obvious relation between the failure rate,

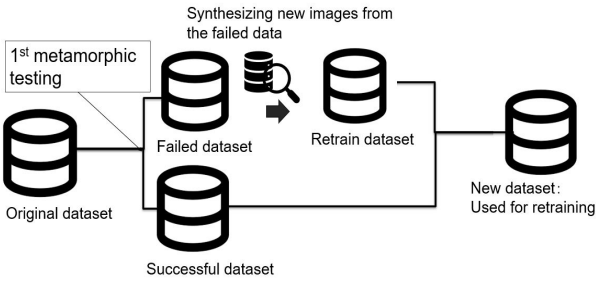

Fig. 7 RQ2: generation of new training dataset

uncertainty, and scores of the base instances. The instances with low uncertainty, in the metamorphic testing, have higher failure rate compared to instances with high uncertainty. The score shows a negative correlation to the failure rate. High-scored instances are usually behaving well, whereas the low-scored instances have higher failure rate. Especially, for high-scored instances, the failure rate tends to be zero when its instances' uncertainty is not low at the same time.

Last but not the least, we also investigate the relevance of failure rate and the inserting instances' scores and uncertainty. However, the correlation between inserting instances and the failure rate is weak with an absolute value of correlation coefficient smaller than 0.005.

5.2 RQ2. How to improve the robustness of the model against realistic insertion attack after data augmentation with failed test cases found in metamorphic testing.

In this section, we aim to improve the robustness of the object detection model by retraining the model with failed data (failed test cases). As shown in Figure 7, we make new training datasets by adding the failed data or conducting some changes to the failed data. Through all experiments, the goal is to find an appropriate retraining dataset using test cases found in metamorphic testing. It is considered whether retraining the model with synthesized images whose insertions are localized onto the *failure locations* (location on the background where failures in metamorphic testing occur) is performing better than retraining with images that are synthesized randomly. In short, it is necessary to prove that the retraining dataset is better at improving the model robustness when its synthesized images are from the failed test cases but randomly synthesized images. Since there are often a number of failure locations for an image during the metamorphic testing, how many failure locations should be utilized to synthesize the training data remains the question.

5.2.1 Approach

Repairing the failures: We aim to find an effective solution to making training datasets to improve the model robustness through retraining the model. Figure 7 illustrates the generation of new training datasets for all experiments in this research question. The instances from the images in the dataset is separated into the successful dataset and failed dataset, whose images pass or fail the metamorphic testing in the first time metamorphic testing. Then, we use the original images of the failed test cases and their prediction databases to synthesize new training data. The prediction databases for the metamorphic testing record the uncertainty, scores of the metamorphic images' instances, and the locations where the failures are triggered. Next, the new training dataset is fed into the prediction model to retrain. The retraining is done in a few epochs, with 1,000 to 3,000 epochs by YOLOv3. Ultimately, we put the retrained model into the second time metamorphic testing and see whether the model is improved.

Table 2 Retraining dataset with synthesized images of different number of preset failure locations. Each column stands for the followings: Unc: average uncertainty. F_r: average failure rate. Pr_1: proportions of data with almost no failures. Pr_2: proportions of data with fewer failures than the original prediction, excluding Pr_1. Pr_3: proportions of data with more failures than original prediction. The sum of these proportions equals to 1. f and r represent the synthesized images are synthesized by how many number of inserting instances from the failure locations or random locations. Baseline are the uncertainty and failure rate after the first time metamorphic testing before retraining.

	Unc	F_r	Pr_1	Pr_2	Pr_3
$0f + 1r$	2.351	0.136	0.286	0.322	0.393
$0f + 2r$	2.463	0.131	0.260	0.339	0.402
$0f + 3r$	2.438	0.141	0.250	0.378	0.383
$0f + 4r$	2.890	0.150	0.267	0.401	0.333
$0f + 5r$	2.692	0.165	0.308	0.328	0.364
baseline	2.314	0.142	-	-	-
	Unc	F_r	Pr_1	Pr_2	Pr_3
$1f + 0r$	2.130	0.133	0.395	0.284	0.325
$2f + 0r$	1.875	0.125	0.351	0.343	0.307
$3f + 0r$	2.027	0.103	0.331	0.405	0.262
$4f + 0r$	2.257	0.105	0.233	0.493	0.274
$5f + 0r$	2.325	0.114	0.221	0.468	0.324
baseline	2.314	0.142	-	-	-

Reduction of the overfitting: New failures are detected during the second time metamorphic testing. It is worthy of notice that retraining can not always reduce model prediction failures for all instances. As Table 2 shows, in all experiments, there are over 20% (Pr_3) of instances whose failure rate is increased after the retraining. There are probably unremovable new failures for each tested image. These are called overfitting for retraining. In the following experiments, we aim to reduce overfitting by two methods:

(1) We find some *clusters* for the failure locations, where failure locations for a single image are likely to cluster near a close place on the background. We utilize these clusters in the experiments to see if the new failures of some instances can be further erased during the retraining.

(2) We utilize the data characteristics (uncertainty and scores) from the prediction databases of failed test cases into synthesizing the image data. We expand the research into finding the instances of which combinations of data characteristics can reduce overfitting.

5.2.2 Results and analysis

Repairing the failures: As Table 2 shows, we investigate the variation of the uncertainty, failure rate, and also repaired failures by investigating the proportions of instances with fewer failures. We compare the difference of these evaluations on model robustness between the random insertions and insertions of failure locations. It is obviously shown that retraining with synthesized images whose insertions on the failure locations performs better than the random locations. In the table, columns Unc and F_r show data characteristics, uncertainty, and failure rate. Columns Pr_1, Pr_2, Pr_3 separately show the proportions of data about how the retraining reduces the original failures in metamorphic testing. Column Pr_1 is the best and the most effective part of the data after retraining the model, whereas Column Pr_3 is the part of data, in which retraining triggers more failures unexpectedly. Since inserting too many instances is likely to overlapping with the base instance and the other inserting instances, in order to avoid this issue, we set a maximum

Table 3 Repairing the failures of the clusterable data and non-clusterable data for all instances of the category *animal*. The table demonstrates the changes of uncertainty and numbers of failures in the second time metamorphic testing. The first row are the specified methods of inserting instances. C_c is the insertions into the clusters of clusterable data. C_r is the insertions into the random failure locations of clusterable data. Nc_n is the insertions into regions near to the contours of non-clusterable data. Nc_f is the insertions into regions far from the contours of non-clusterable data. Baseline represents the average values of all data in the first time of metamorphic testing.

Insertion locations	C_c	C_r	Nc_n	Nc_f	Baseline
1. uncertainty	2.617	2.936	2.072	1.996	2.568
2. total number of failures (among 100)	7.52	11.41	8.15	7.43	12.39
3. repaired failures compared to the first time metamorphic testing	3.31	2.57	1.46	1.29	-
4. new failures in the second time metamorphic testing	1.84	2.66	1.62	1.37	-
5. proportions of data whose failures are mostly removed	23.61%	20.12%	25.61%	22.34%	-
6. proportions of data with almost no new failures	35.40%	19.50%	14.72%	16.88%	-

Table 4 New failures and repaired old failures for different instances used in second time retraining. For each cell, the left is the number of new failures, and the right the number of total repaired failures. These number are the average numbers per one hundred synthesized test cases.

	low uncertain inserting instance		high uncertain inserting instance		low score inserting instance		high score inserting instance	
low uncertain base instance	0.036	-3.451	0.052	-3.447	0.842	-3.325	0.151	-2.632
high uncertain base instance	1.825	-0.875	1.469	-0.924	1.536	-0.843	1.057	-0.617
low score base instance	1.215	-1.774	1.226	-1.554	2.536	-1.197	2.057	-0.383
high score base instance	1.657	-2.148	1.832	-1.843	1.317	-1.486	3.057	-2.57

of five times of insertions onto an image for each experiment's new training dataset to compare the difference.

As shown in Table 2, the baseline of the failed tested data is of 2.314 uncertainty and 14.24% failure rate. It is apparent that the retraining images with randomly localized inserting instances causes higher uncertainty to the model predictions. With increasing numbers of random locations, the uncertainty roars up, which points out the increasing prediction variance. The proportions of data with more failures increase to 40% ($0f + 2r$ in the first table), which is more than the counterpart in the failure localized insertions (30% around).

Retraining datasets with synthesized data whose insertions are localized on the failure locations can improve the model robustness by lowering the uncertainty by 12.8% and failure rate by 26.3% ($3f + 0r$ in the second table). The failure rates, as seen in the second table of Table 2 is lower than the baseline, which suggests that the model after the retraining reduces the possible failures than the original model. Moreover, only increasing the number of failures localized inserting instances does not have a positive influence, as seen from the last rows of each table in Table 2, on the improvement of model robustness. To improve the retraining effect, it is also important to control the number of instances used for data augmentation.

As seen in the results, it is inevitable that new failures occur due to the retraining of the model, which is seen as overfitting. In the next section, the research results of reducing overfitting are introduced.

1. Reduction of overfitting through the choices of failure locations.

These experiments focus on the usage of failure locations from the prediction databases of failed test cases. From observing the failure locations of some failed tested images, we find that there are some specific locations on which the prediction of base instances is more likely to be attacked during the metamorphic testing. Hence, these locations, which often locate near a single place on the background, are referred to as the *clusters* of the failure locations. Some of the images with the clusters are called *clusterable data*, whereas the others without the clusters are called the *non-clusterable data*. For clusterable data, we use the place of failure locations' cluster to insert the inserting instances rather than randomly choosing the failure locations, and synthesize the metamorphic images for retraining. For non-clusterable data, we randomly insert the inserting instances onto the failure locations. In the preliminary experiments, it is found that there are 36.4% clusterable data with 11.4% F_r, and 63.6% non-clusterable data with 18.9% F_r. It is investigated throughout all instances in the dataset about how many new failures occur due to retraining.

As shown in the table 3, it is seen that the overfitting (new failures due to the retraining) can be alleviated for the clusterable data, with new failures reduced from 2.66 to 1.84 (new failures in the second time metamorphic testing for C_c and C_r). Additionally, there are 35.4% of the clusterable data are retrained, almost without new failures in the second time metamorphic testing. Conversely, the failures of non-clusterable data are well reduced no matter how the insertions are located on the image.

2. Reduction of overfitting of data with different characteristics of instances.

In RQ1, we find that the low uncertain instances have higher failure rate in metamorphic testing. It is greatly likely that after the original training dataset is augmented, some failures of low uncertain instances are removed, with fewer new failures as well, as seen from Table 4. Through the table, low uncertain base instances when inserted by whatever instances are more likely to repair the old failures (failures that is in the prediction base for that image of the first time metamorphic testing) by -3.451, and reduce the new failures by 0.036, than the high uncertain instances. Comparatively, high uncertain base instances generate new failures by 1.825 when inserting low uncertain inserting instances and their failures are not easy to fix compared to the low uncertain base instances.

6. Discussions

6.1 About RQ1: The relation of failure rate and uncertainty reflects the latent problems of the models.

The uncertainty can be explained by the prediction of variance derived by the object detection model itself but any other standards, thus it is unstable for an object detection model to evaluate the robustness merely by uncertainty. During the metamorphic testing, the low uncertain instances highly trusted by the model are not reliable under realistic attacks. The instability of the model with predicting low uncertain instances during the metamorphic testing

manifests that a self-confident model needs further training to improve its robustness.

In the COCO dataset, the high uncertain instances are mostly from images with sophisticated semantic contexts (with many instances located on every corner). They are not vulnerable to realistic attacks in metamorphic testing. Thus, when retraining the high uncertain instances, it is acceptable that the failure rate for some metamorphic images are even lower than its original tested ones. The object detection model show a unstable predictions of the high uncertain instances.

6.2 About RQ2: Repairing through retraining is effective for only part of the data.

Retraining the model is a commonly-used method to improve the model. However, even retraining needs a practical solution. Currently, the questions discussed in RQ2 are related to the data characteristics of a realistic attack, like in the production environment. The situation of detecting instances in an image is always complicated. Some data are improved after the retraining, whereas some of the data as investigated in the research still retain their failures in metamorphic testing. From the results of RQ2, we find that the low uncertain data which are highly trusted by the model deserve the further retraining to improve the model robustness.

Those current repairable data which failed in the metamorphic testing have the potential to be learned by the models. We notice that the clusters of the failures help detecting the potential failures in metamorphic testing, but there is no direct method we can utilize the failure locations for the non-clusterable data. As a matter of fact, failure locations of non-clusterable data are stochastically distributed on the whole background. Those failure locations can often occur when inserting instances anywhere on their images' background. So far, we are unable to find out and make use of all possible data characteristics (such as the categories of inserting instances w.r.t the base instances) merely by metamorphic testing for these data, and we will figure them out in future research.

7. Threats and Validity

7.1 Internal validity

The metamorphic testing for the experiments requires the data to be easy to interpret with low uncertainty. Most of the instances to be tested are those which are easily detected by the model. For the undetectable instances with low scores, metamorphic testing is not effective to improve the robustness of the object detection model. There remains the issue of how to utilize the undetectable instances to improve the current metamorphic testing.

7.2 External validity

The metamorphic testing of the placing insertions currently does not evolve in the dynamic object detection scene, object tracking. The influence of the inserting instance might have a great difference between static images and dynamic images (videos). To ensure the validity of metamorphic testing for object detection, it is necessary to validate that insertions trigger failure in a dynamic process.

8. Conclusion

In this paper, the research of metamorphic testing to evaluate and improve the robustness of object detection models is discussed. Metamorphic testing utilizes the insertions of instances to synthesize the metamorphic images and assert the metamorphic relations of the model predictions. The model manifests different robustness in metamorphic testing due to the data with different characteristics, including the uncertainty and scores. Then, retraining the model with data augmentation of failed test cases in the first time metamorphic testing improves the model robustness against the realistic attack by repairing the old failures and reducing new failures.

In future research, we will find more effective methods to augment the training dataset to improve its robustness. The data characteristics are limited to describing all of the sophisticated situations under realistic scenarios since it is essential to improve the model robustness for high uncertain instances. In addition, from the perspective of machine learning, the pre-processing of metamorphic image data should be considered further in order for simulating realistic scenarios.

Acknowledgments This research was partially supported by JSPS KAKENHI Japan (Grant Numbers: JP18H04097, JP20H04167, JP21H04877, JP22K17874, JP22K18630).

References

[1] Guo, S., Wang, S., Yang, Z., Wang, L., Zhang, H., Guo, P., Gao, Y. and Guo, J. A Review of Deep Learning-Based Visual Multi-Object Tracking Algorithms for Autonomous Driving. *Applied Sciences*. **12**, 10741 (2022)

[2] Chang, W., Chen, L., Hsu, C., Lin, C. and Yang, T. A deep learning-based intelligent medicine recognition system for chronic patients. *IEEE Access*. **7** pp. 44441-44458 (2019)

[3] Li, D., Zhang, J. and Huang, K. Universal adversarial perturbations against object detection. *Pattern Recognition*. **110** pp. 107584 (2021)

[4] Wang, S. and Su, Z. Metamorphic object insertion for testing object detection systems. *2020 35th IEEE/ACM International Conference On Automated Software Engineering (ASE)*. pp. 1053-1065 (2020)

[5] Harakeh, A., Smart, M. and Waslander, S. Bayesod: A Bayesian approach for uncertainty estimation in deep object detectors. *2020 IEEE International Conference On Robotics And Automation (ICRA)*. pp. 87-93 (2020)

[6] Lin, T., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C. Microsoft coco: Common objects in context. *European Conference On Computer Vision*. pp. 740-755 (2014)

[7] Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. *2017 Ieee Symposium On Security And Privacy (sp)*. pp. 39-57 (2017)

[8] Katz, G., Barrett, C., Dill, D., Julian, K. and Kochenderfer, M. Towards proving the adversarial robustness of deep neural networks. *ArXiv Preprint ArXiv:1709.02802*. (2017)

[9] Redmon, J. and Farhadi, A. Yolov3: An incremental improvement. *ArXiv Preprint ArXiv:1804.02767*. (2018)

[10] Chen, T., Cheung, S. and Yiu, S. Metamorphic testing: a new approach for generating next test cases. *ArXiv Preprint ArXiv:2002.12543*. (2020)

[11] Madry, A., Makelov, A., Schmidt, L., Tsipras, D. and Vladu, A. Towards deep learning models resistant to adversarial attacks. *ArXiv Preprint ArXiv:1706.06083*. (2017)

[12] Gal, Y. and Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *International Conference On Machine Learning*. pp. 1050-1059 (2016)

[13] Niu, X. and Jiao, Y. An overview of perceptual hashing. *ACTA ELECTONICA SINICA*. **36**, 1405 (2008)

[14] Selvaraju, R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. and Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. *Proceedings Of The IEEE International Conference On Computer Vision*. pp. 618-626 (2017)

[15] Bolya, D., Zhou, C., Xiao, F. and Lee, Y. Yolact: Real-time instance segmentation. *Proceedings Of The IEEE/CVF International Conference On Computer Vision*. pp. 9157-9166 (2019)