

If a Human Can See It, So Should Your System: Reliability Requirements for Machine Vision Components

Boyue Caroline Hu
boyue@cs.toronto.edu
University of Toronto
Toronto, Ontario, Canada

Lina Marsso
lina.marsso@utoronto.ca
University of Toronto
Toronto, Ontario, Canada

Krzysztof Czarnecki
kczarnec@gsd.uwaterloo.ca
University of Waterloo
Waterloo, Ontario, Canada

Rick Salay
rsalay@gsd.uwaterloo.ca
University of Waterloo
Waterloo, Ontario, Canada

Huakun Shen
huakun.shen@mail.utoronto.ca
University of Toronto
Toronto, Ontario, Canada

Marsha Chechik
chechik@cs.toronto.edu
University of Toronto
Toronto, Ontario, Canada

ABSTRACT

Machine Vision Components (MVC) are becoming safety-critical. Assuring their quality, including safety, is essential for their successful deployment. Assurance relies on the availability of precisely specified and, ideally, machine-verifiable requirements. MVCs with state-of-the-art performance rely on machine learning (ML) and training data, but largely lack such requirements.

In this paper, we address the need for defining machine-verifiable reliability requirements for MVCs against transformations that simulate the full range of realistic and safety-critical changes in the environment. Using human performance as a baseline, we define reliability requirements as: ‘if the changes in an image do not affect a human’s decision, neither should they affect the MVC’s.’ To this end, we provide: (1) a class of safety-related image transformations; (2) reliability requirement classes to specify correctness-preservation and prediction-preservation for MVCs; (3) a method to instantiate machine-verifiable requirements from these requirements classes using human performance experiment data; (4) human performance experiment data for image recognition involving eight commonly used transformations, from about 2000 human participants; and (5) a method for automatically checking whether an MVC satisfies our requirements. Further, we show that our reliability requirements are feasible and reusable by evaluating our methods on 13 state-of-the-art pre-trained image classification models. Finally, we demonstrate that our approach detects reliability gaps in MVCs that other existing methods are unable to detect.

CCS CONCEPTS

• **Software and its engineering** → **Requirements analysis**; • **Computing methodologies** → *Computer vision*.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ICSE '22, May 21–29, 2022, Pittsburgh, PA, USA

© 2022 Association for Computing Machinery.

ACM ISBN 978-1-4503-9221-1/22/05...\$15.00

<https://doi.org/10.1145/3510003.3510109>

KEYWORDS

Software Engineering for Artificial Intelligence, Requirements Engineering, Software Analysis, Machine Learning, Computer Vision

ACM Reference Format:

Boyue Caroline Hu, Lina Marsso, Krzysztof Czarnecki, Rick Salay, Huakun Shen, and Marsha Chechik. 2022. If a Human Can See It, So Should Your System: Reliability Requirements for Machine Vision Components. In *44th International Conference on Software Engineering (ICSE '22)*, May 21–29, 2022, Pittsburgh, PA, USA. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3510003.3510109>

1 INTRODUCTION

The use of Machine Vision Components (MVCs) in safety-critical systems, such as self-driving cars, creates major safety concerns, since undesired behaviors can lead to fatal accidents [56]. For example, recently, Tesla self-driving cars misclassified emergency vehicles and caused multiple crashes [4, 5]. Knowing how to analyze these components, provide safety assurance, and ensure their quality becomes a must for their usability in safety-critical domains. Particularly, in systems that automate tasks normally performed by humans, such as driving, the vision task is performed by MVCs which represent a critical function for the overall system safety. However, vision tasks are difficult to specify; thus, they are usually performed using machine learning (ML) [53]. Defining requirements for ML is not trivial because the inability to specify clear requirements is the reason to use ML in the first place [8, 24, 46]. Yet such requirements are necessary for verification and providing safety guarantees. As a first step towards safe MVCs, one needs to define what it means for an MVC to be correct and then check its correctness prior to system deployment.

In this paper, we focus on one aspect of correctness: *reliability*, which measures the ability of a system or component to perform its required functions in a specified environment [1], as it enables ensuring the quality of the deployed system. We are specifically interested in whether the performance of an MVC remains reliably unaffected by image transformations that commonly occur in real-world scenarios. This question has been studied in SE and ML literature as *model robustness*, including testing [52] and verification [23] techniques. Yet, given the lack of detailed reliability requirements, these approaches are limited to checking the models within a *small* neighbourhood of the original input image, i.e., by applying perturbations that are almost imperceptible to humans.

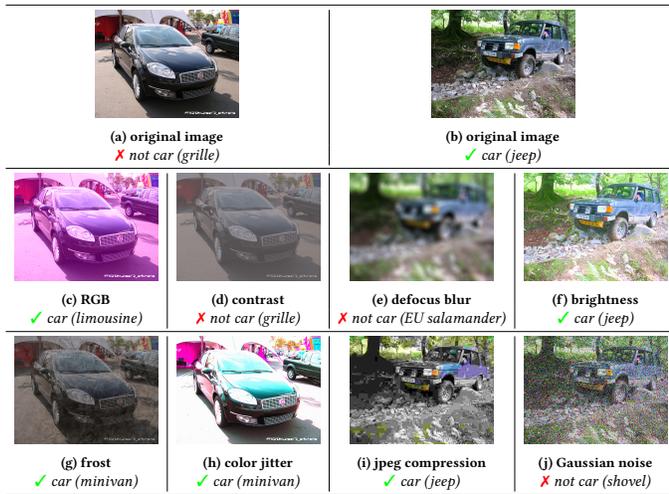


Figure 1: Image recognition on original and transformed images. The top row displays original images containing cars from the ILSVRC'12 dataset [38]. The transformation applied to the image is specified under the image. Images from (c) to (j) present all the safety-related transformations considered in this paper. The classification result of a state-of-the-art MVC ResNet50 [17] is shown in brackets in italics under each image. We further specify whether the predicted category is considered as a car (✓), or not (✗), based on the ILSVRC'12 class hierarchy. Transformations are implemented by Albumentations [6] and Imagenet-c [19].

While considering only the small perturbations allows for requirement analysis of model reliability [22], its applicability is limited in the real-world scenarios, with a much broader range of possible changes. For example, consider the problem of recognizing cars in images – see a few examples in Fig. 1. We are interested in being able to recognize cars under such transformations as frost (see Fig. 1g) and different brightness levels (Fig. 3d and Fig. 3g).

The range of transformation magnitudes in images in Fig. 1 is not considered small or imperceptible. While humans have no problem recognizing cars in these images, the state-of-the-art image classification model ResNet50 [17] failed to do so on the examples in Figs. 1d, 1e and 1j. Since MVCs are used in systems that automate tasks normally performed by humans, MVCs, like ResNet50, are *at the minimum* expected to consistently classify objects across range of changes that do not affect human perception. Thus, we seek a method to establish human performance as a reference for defining reliability, and an automated method to check MVC against a justified range of changes that do not affect human performance.

In this paper, we formally define two classes of machine-verifiable reliability requirements for MVCs: *correctness-preservation* and *prediction-preservation*. For both requirements classes, the range of image changes we consider (i.e., the human-tolerated range), is a parameter estimated using experiments with human participants. Intuitively, within the human-tolerated range of changes, *correctness-preservation* requires that the MVC's predictions after changes in images should be correct, and *prediction-preservation* requires that the predictions on original images and on images that underwent transformations should be the same. Specifically, this paper

makes the following contributions: (1) We identify a class of safety-related image transformations; (2) We provide a formal specification of two classes of input-output reliability requirements for MVCs, with parameters representing human performance; (3) We present a method to instantiate our requirements classes into machine-verifiable requirements. This method estimates ranges of changes to images that do not affect human vision using results of experiments with human participants; (4) We provide human experiment performance data for image recognition; (5) We provide an automated method for checking MVCs against our machine-verifiable requirements.

While our criteria are defined for any computer-vision task (including object detection and semantic segmentation), in this paper, we demonstrate the feasibility of our approach on the image classification task. We show that our approach captures reliability gaps that existing methods are unable to detect using 13 state-of-the-art pre-trained image classification models on two image classification datasets (Imagenet [38] and CIFAR-10 [30]).

Significance: To the best of our knowledge, we are the first to define reliability requirements for MVCs using a human-justified range of changes over realistic safety-related transformations. Our requirements and the method for checking their satisfaction can be reused by software engineers for analyzing system reliability of MVCs before deployment.

The rest of the paper is organized as follows: Sec. 2 gives an overview of our approach for creating and checking our reliability requirements. Sec. 3 presents the safety-related image transformations and a generic metric for measuring changes in images. Sec. 4 presents a formal specification of our reliability requirement classes. Sec. 5 presents our experiment for measuring human recognition performance with human participants, and demonstrates an automated approach for estimating parameters of the requirements, using data from this experiment. Sec. 6 introduces an automated method for checking MVC's against our reliability requirements. We evaluate our approach in Sec. 7. Sec. 8 compares our work with related approaches and we conclude in Sec. 9.

2 APPROACH OVERVIEW

Fig. 2 gives an overview of our approach. Given (i) a vision task for the MVC, (ii) a safety-related transformation and (iii) experimental data for estimating the ranges of visual changes that do not affect human performance, we provide a process for instantiating machine-verifiable reliability requirements for MVC (requirement instantiation) and a process for checking whether an MVC satisfies these instantiated requirements (requirement checking).

The vision task and the transformation need to be selected based on the application of the MVC. To help with the selection of transformations that represent changes likely to happen in the operating environment, we identified a class of safety-related image transformations that represent potentially risky input modifications in real world situations. For example, frost shown in Fig. 1 is a safety-related transformation because it can reduce lighting in the scene which, in turn, can cause machine vision errors. Note that since transformations have different parameter domains and can have different visual effects on different images to humans, we defined a generic metric called a *visual change* and denoted by Δ_v , which

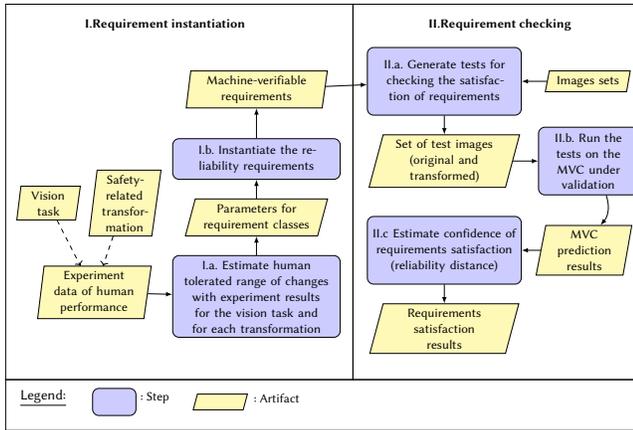


Figure 2: A process for instantiating two reliability requirements classes for MVCs (requirement instantiation) and a process for checking their satisfactions (requirement checking).

decouples the perceptible visual change to the image from the transformation parameters and thus allows stating the reliability requirements on the MVC more abstractly.

Requirement instantiation: This automated step enables users to instantiate the reliability requirements for the vision task with human tolerated range of visual changes for each selected transformation. The human-tolerated range is the requirement parameter that describes the range of changes from a transformation that should not affect the MVC’s behavior. This requirement parameter is measured with Δ_v and estimated using results from experiments with human participants. The output of this step is a set of machine-verifiable requirements. The resulting correctness-preservation [resp. prediction-preservation] requirement states: for a vision task and a transformation, if the changes in the images are within the estimated human tolerated range, then an MVC should preserve the correctness [resp. prediction] after applying the changes to its input from before.

For example, for the transformation adding frost artificially, our resulting requirements are as follows:

- the recognition accuracy of an MVC should not decrease if the visual change in the images is within the range $\Delta_v \leq 0.84$ (*correctness-preservation*); and
- the percentage of labels the MVC can preserve after adding frost should not decrease if visual change in the images is within the range $\Delta_v \leq 0.91$ (*prediction-preservation*).

Note that our requirements do not depend on the state-of-art ML techniques since they treat the MVC as a black box.

Requirement checking: This automated method checks whether an MVC satisfies the instantiated reliability requirements. Given a set with original images, our process generates test cases (step II.a) by transforming the original images within the range specified in the instantiated requirements, runs the tests on the model (step II.b), and checks whether the MVC satisfies our requirements (step II.c).

To summarize, our proposed approach can be used to automatically generate machine-verifiable reliability requirements for a vision task and a list of transformations, given human experiment

results; and then automatically evaluate whether an MVC satisfies these requirements. In the above example, the requirement checking method will generate a set of test case images within the ranges (0.84 and 0.91) to check whether an MLC satisfies these requirements.

An implementation of our method is available online.¹ For the purpose of demonstration and evaluation, we conducted image classification experiments with 2000 human participants for the vision task of recognizing car images for 8 transformations: RGB, contrast, defocus blur, brightness, frost, color jitter, jpeg compression, and Gaussian noise (see images in Fig. 1(c)-(j)). In the rest of the paper, we describe the technical details of each step of Fig. 2 using this experiment.

3 VISUAL CHANGES IN IMAGES

In this section, we start with establishing the definition of the metric Δ_v , which measures human visual changes in images caused by transformations. Then, we identify a class of safety-related image transformations that are used to instantiate our *correctness-preservation* and *prediction-preservation* requirements (see Sec. 4).

A key idea in our work is to define reliability requirements relative to Δ_v ranges rather than the transformation parameter ranges to be tolerated. This is important since each transformation may have one or more parameters, and each parameter may affect the transformed image to a different degree—also depending on the input image. For example, brightening an already bright image makes the objects harder to see; on the other hand, making a dark image brighter will have the opposite effect. Further, small changes to one parameter may cause small changes or large changes to the transformed image depending on the values of other parameters. The visual change metric Δ_v allows us to abstract from these complexities of the transformation parameter space. Also, a simple image distance metric such as mean squared error, which is often used to define robustness, e.g., [3, 9], does not adequately reflect the human-perceived visual change in images [48]. Thus, we base Δ_v on image quality assessment metrics.

Background: Image Quality Assessment (IQA). IQA metrics are quantitative measures of human objective image quality [49]. Given the original image, the IQA metrics automatically predict the perceived image quality by measuring the perceptual ‘distance’ between the two images [42]. This ‘distance’ is different from pixel distance and its calculation depends on the design of the IQA metric. VSNR (*Visual-Signal-to-Noise-Ratio* [7]) checks the visibility of the changes in images and returns infinity (∞) if they are not visible to humans [7]. VIF (*Visual Information Fidelity* [42]) measures the information fidelity by analyzing the statistics of the natural scenes in the images. VIF returns a value between 0 and 1 if the changes degrade perceived image quality, with 1 indicating the perfect quality compared to the original image; and it returns a value > 1 if the changes enhances image quality [42]. VIF is empirically shown to be the closest to human opinions when compared to all other IQA metrics [43] and VSNR has been shown to be effective to detect non-visible changes [22]. VIF is applicable to transformations

¹ See https://carolineeeeeee.github.io/automating_requirements for implementation, more results and information.

that can be described locally by a combination of signal attenuation and additive Gaussian noise in the sub-bands in the wavelet domain [42].

Measuring visual change in images. We now use IQA metrics to define a generic metric Δ_v . Our definition of Δ_v shares the same applicability characteristics as VNSR and VIF. For transformations that satisfy this characteristic (e.g., noise, blur, brightness and contrast changes, color change, etc.), the definition is as follows:

Definition 1: Visual change Δ_v

Let an image x , an applicable transformation T_X with a parameter domain C and a parameter $c \in C$, s.t. $x' = T_X(x, c)$ be given. $\Delta_v(x, x')$ is a function defined as follows:

$$\begin{cases} 0 & \text{If VSNR}(x, x') = \infty \\ & \text{or VIF}(x, x') > 1 \\ 1 - \text{VIF}(x, x') & \text{Otherwise} \end{cases}$$

Basing Δ_v on IQA metrics means that it provides a generalized quantitative measure for visual changes in the images that is independent of particular images and transformations. We split this definition into two cases. The first corresponds to changes imperceptible to humans (when $\text{VSNR}(x, x') = \infty$) and changes that enhance the visual quality (when $\text{VIF}(x, x') > 1$). In this case, $\Delta_v = 0$ because such changes do not impact human recognition of the images negatively. The other case deals with visible changes that degrade visual quality. Since VIF returns 1 for perfect quality compared to the original image, the degradation is one minus the image quality score. For example, the visual change of the example in Fig. 1f compared to its original image in Fig. 1b is 0.507. The visual change of the example in Fig. 1e compared to the one in Fig. 1b is 0.985. This suggests that the transformation in Fig. 1e causes more change visually than the one in Fig. 1f.

Safety-related image transformations. We say that a transformation is *safety-related* if it can lead to a hazard in a real-world machine-vision application scenario. To assess this in a systematic manner, we utilize the CV-HAZOP checklist [55]. This checklist comprehensively identifies the potentially hazardous impacts of different modes of interference in the computer vision (CV) process, which is comprised of light sources, transmission medium, object, observer, and algorithm. A transformation that can produce such impacts is considered safety-related. For example, contrast adjustment, defocus blur, and added Gaussian noise shown in Fig. 1 are safety-related transformations because they can reduce lighting in the scene, cause blurring, and add noise in the images, which can cause machine vision errors and subsequent system failures. Since the scope of CV-HAZOP is broader than the image transformation assessment task, we remove non-image-related hazard scenarios entries from the checklist. In particular, entries related to *Algorithm* in the vision process are not relevant because they modify the CV algorithm and not the images. Entries concerned with the *Number of Observers* are also not relevant since they focus on the interaction between the observers and cannot be represented by single image transformations. Finally, image transformations

cannot make temporal changes; therefore, entries which deal with time are not relevant either.

To determine whether a given image transformation belongs to our safety-related class, one should first identify the location in the vision process to which the transformation corresponds; then the property of the process location that the transformation is affecting (CV-HAZOP parameters); and finally, how the transformation is changing the property (CV-HAZOP guide words). For example, defocus blur is changing the focus of the observer (CV-HAZOP entry No.1018), i.e., camera, and therefore belongs in our class. Supplementary material¹ includes the full list of CV-HAZOP safety-related entries (954 entries chosen from the overall 1470).

In this paper, we consider transformations provided by the state-of-the-art library Albumentations [6] and the ML robustness benchmark Imagenet-c [19], which consist of 50 unique transformations. Of these, 45 are safety-related. We further remove those transformations that cannot produce a continuous range of transformed images, yielding 31 transformations—see supplementary material¹ for the full list. Since multiple transformations can correspond to a single CV-HAZOP entry, we only instantiate our approach on one transformation per CV-HAZOP entry, resulting in the eight transformations illustrated in Fig. 1c-j: *RGB, contrast, defocus blur, brightness, frost, color jitter, jpeg compression* and *Gaussian noise addition*.

4 SPECIFICATION OF RELIABILITY REQUIREMENTS CLASSES

In this section, we provide a formal specification of our two reliability requirements classes: *correctness-preservation* and *prediction-preservation*.

Let us assume that we are given an MVC f , a distribution of input images P_X , a ground-truth labeling function f^* , a transformation T_X with parameter domain C and parameter distribution P_C . Our requirements use the joint distribution of pairs of original and transformed images, defined as $P_{T_X}(x, x') = P_X(x) \sum_{c \in C, x' = T_X(x, c)} P_C(c)$.

We first introduce our *correctness-preservation* reliability requirement class. It assumes a performance measure $m(f, f^*, P_X)$, which is typically a measure of similarity between the output of f and f^* given that the input $x \sim P_X$. Note that m should be adequate for the vision task, such as classification accuracy for image classification, intersection over union (IoU) for image segmentation, and average precision for object detection. We define the marginal distribution of transformed images with changes within the human tolerated-range $\Delta_v \leq t_c$ as $P_{T_X, t_c}(x') = \sum_{x \in X} P_{T_X}(x, x' | \Delta_v(x, x') \leq t_c)$.

Definition 2: Correctness-preservation requirement, with parameters T_X and t_c

Intuitively: For the range of changes in images that do not affect human performance ($\Delta_v \leq t_c$), the *performance* of machine vision component f should not be affected as well. Note that ground truth is required.

Formally: We require the performance m of f for transformed images to be equal to or larger than that for original images: $m(f, f^*, P_{T_X, t_c}) \geq m(f, f^*, P_X)$.

Example: For the transformation brightness, *correctness-preservation* requires $m_{t_c} = m(f, f^*, P_{T_X, t_c})$, the classification accuracy of ResNet50 on all transformed images (which is the percentage of *correct* predictions in Fig. 3d-3i), to be at least $m_0 = m(f, f^*, P_X)$, the classification accuracy of the model on all original images, which is the percentage of *correct* predictions in Fig. 3a-3c. Both accuracies are $1/3 \approx 33\%$, and the requirement is satisfied.

We then introduce our *prediction-preservation* requirement class. Given a distance measure $d(f(x), f(x'))$, which measures distance between the output of f on two input images, we define a *prediction-similarity* measure $s(f, P_{X \times X}) = 1 - E_{(x, x') \sim P_{X \times X}} [d(f(x), f(x'))]$, which measures the expected similarity between the output of f on two images drawn from $P_{X \times X}$, a distribution of image pairs. Note that d should also be adequate for the vision task; for example, for image classification, $d(f(x), f(x')) = 0$ if $f(x) = f(x')$ and 1 otherwise. We define the distribution of *pairs* of original and transformed images that are within the human-tolerated range for prediction-preservation $\Delta_v \leq t_p$ by conditioning the joint distribution P_{T_X} as follows: $P_{T_X, t_p}(x, x') = P_{T_X}(x, x' | \Delta_v(x, x') \leq t_p)$. Since s compares outputs with the original outputs, s of original images would always be 1, which is not necessarily achievable. As an alternative, we estimate s of original images with s of images with minimal image changes ($\Delta_v \leq \epsilon$). More precisely, we rank the image pairs by Δ_v , determine ϵ as a lower q -th quantile in the ranking, and define the distribution of image pairs with $\Delta_v \leq \epsilon$ as $P_{T_X, \epsilon}(x, x') = P_{T_X}(x, x' | \Delta_v(x, x') \leq \epsilon)$.

Definition 3: Prediction-preservation requirement, with parameters T_X and t_p

Intuitively: For the range of changes in images that do not affect human predictions ($\Delta_v \leq t_p$), the *predictions* of machine vision component f should stay unaffected as well. Note that ground truth *is not* required.

Formally: We require the prediction similarity s of f for all transformed images to be equal to or larger than that of images transformed with $\Delta_v \leq \epsilon$, which is: $s(f, P_{T_X, t_p}) \geq s(f, P_{X, \epsilon})$.

Example: For the transformation brightness, *prediction-preservation* requires $s_{t_p} = s(f, P_{T_X, t_p})$, the prediction similarity of ResNet50 for all transformed images vs. originals, which is the percentage of predictions in images in Fig. 3d-3i *preserved* from images in Fig. 3a-3c and thus $5/6 \approx 83\%$, to be at least equal to $s_0 = s(f, P_{X, \epsilon})$, the prediction similarity of the model for images transformed with $\Delta_v \leq \epsilon$. Given the very small sample, we set ϵ to the median, and thus s_0 is the percentage of predictions preserved for images in Fig. 3d-3f, and $s_0 = 3/3 = 100\%$. Thus, the requirement is not satisfied.

Definitions of the two reliability requirements are similar, with two main differences. First, correctness-preservation relies on a performance metric to compare predictions to ground truth, whereas prediction-preservation uses prediction similarity to compare predictions on transformed images vs. originals. Second, correctness-preservation compares performance on the full range of transformed images with that on the originals, whereas prediction-preservation compares the prediction similarity for the full range



Figure 3: Image recognition on original and transformed images. Images from (d) to (i) display the same transformation, brightness, applied with different magnitudes. The classification result of ResNet50 is shown in *italics* under each image.

of transformed images vs. originals to that for the minimally transformed images (i.e., $\Delta_v \leq \epsilon$) vs. originals. The design choices for the prediction-preservation requirement completely remove the need for human labels on test images, and make this requirement applicable in environments where such labels are unavailable.

Finally, we define *reliability distance* as the difference between the target and the actual correctness- or prediction-preservation, i.e., $\Delta m = m_0 - m_{t_c}$ and $\Delta s = s_0 - s_{t_p}$, respectively. This distance indicates how well the MVC satisfies the respective requirement: zero distance indicates just meeting it; negative distance indicates performing better than the target by a margin; and positive distance indicates how far the MVC is from meeting the target.

5 INSTANTIATING RELIABILITY REQUIREMENTS

To obtain the reliability requirements range parameters t_c , and t_p in Defs. 2 and 3, we perform two experiments with human participants and then estimate the parameters from the experimental results to obtain thresholds at which the human performance drops statistically significantly (step I.a of Fig. 2). This section first presents the experimental setup and procedure, and then introduces our method for instantiating the requirements from the experimental results (requirement instantiation, steps I.a-b).

Experiments with human participants. The objective of the human experiments, one per dataset, is to obtain human predictions on original and transformed images, to be used to estimate t_c , and t_p in step I.a. The experiment inputs are the task to be performed; the transformation T_X ; the dataset of original images $\{x_i\}$, $x_i \sim P_X$,

with their ground-truth labels $\{f^*(x_i)\}$; and distributions P_C for each transformation parameter. Given these inputs, we generate a sample of original-transformed images $\{(x_i, x'_{i,j})\}$, $(x_i, x'_{i,j}) \sim P_{TX}$, by randomly selecting x_i from $\{x_i\}$ and $c_j \sim P_C$, and transforming $x'_{i,j} = T_X(x_i, c_j)$. To obtain the human predictions for each image in $\{(x_i, x'_{i,j})\}$ for image classification, we follow the experimental design of Geirhos et al. [16]. The experiment is a *forced-choice image categorisation task*: humans are presented with the images with transformations applied, for 200 ms, and asked to choose one of the two categories (e.g., car or not car). Between images, a noise mask is shown to minimize feedback influence in the brain [16]. The tasks are timed to ensure fairness in the comparison between humans and machine [14]. However, in contrast to the work by Geirhos et al., we ensure that the full range of achievable Δ_v values is covered when sampling from P_C , and we also collect human predictions for the original images, so that we can estimate prediction preservation. A given subject is never shown more than one version of x_i , whether original or transformed. Note that human predictions for originals are different from their ground truth labels: labelers take as long as needed per image to classify it, but our subjects have only 200 ms to see each image. The human data is specific to a task, an image distribution, and a class of transformations, and thus has to be collected for the combination of the three. In other words, the data is reusable for different samples from the same distribution, or, intuitively, for images sharing the same characteristics, e.g., the same image resolution, the same objects, etc. For example, it is reusable across different sets of images of road scenes taken within the same geographic area using cameras with same the resolution and image quality.

To generate predictions for our evaluation, we perform the experiment on two datasets: ILSVRC'12 [38] and CIFAR-10 [30]. While CIFAR-10 has images of much lower resolution than ILSVRC'12, we include this dataset to compare our method to the existing work on robustness checking, which uses CIFAR-10. We also select the eight safety-related transformations (see images from Fig. 1c-j), as discussed in Sec. 3, and set P_C to be uniform. To fit our labeling budget, we limit the task to a binary classification problem of recognizing car instances. Also, while we apply the eight transformations to ILSVRC'12, we limit the experiment on CIFAR-10 to four transformations that are also used in the works we compare with. To differentiate between car and non-car instances, we use the class hierarchy from the ILSVRC'12 dataset. For each of the selected transformations, we sample 1000 pairs $(x_i, x'_{i,j})$, and have each image (original or transformed) labeled by five humans. To achieve this, we divide the 1,000 (pair samples) \times 8 (#considered) transformations into batches of 20 images. Each batch is shown five times to different participants using the platform Amazon Mechanical Turk. We include qualification tests and sanity checks aimed to filter out participants misunderstanding the task and spammers [35], and only consider results from participants who pass both tests. As a result, for the ILSVRC'12 image classification task, we use $\{x_i\}$ with 13,000 car images and same number of non-car images, and collect human predictions for 40,000 ($= 5 \times 8 \times 1,000$) transformed images and the same number of the original images, for a total of 80,000 predictions. Note that the effort required for measuring human performance is significantly smaller than the the dataset labeling

effort needed for model training. For example, we collect human predictions for 5,000 transformed images per transformation, with 0.2 s timebox per image, for a total of 1000 s. Training sets are typically over 100,000 images, with at least three ground truth labels assigned independently to each image (for quality control) and each takes multiple seconds; e.g., $100,000 \times 3 \times 2 \text{ s} = 600,000 \text{ s}$. The human experiment results can be found in supplementary material¹.

Estimating tolerated range parameters and instantiating requirements. We propose the following procedure to estimate t_c and t_p from the experimentally-obtained human predictions (step I.a). The key idea is to group and order the image pairs by Δ_v , compute the human performance m_k and human prediction similarity s_k in each group, and use a statistical test to determine the t_c [resp. t_p] value of Δ_v at which m_k [resp. s_k] drops significantly from the human performance m_0 for the original images [resp. the human prediction similarity s_0]. Recall that s_0 is the the human prediction similarity for images transformed with $\Delta_v \leq \epsilon$ vs. originals; we set ϵ to the lower 5th percentile.

More precisely, we determine threshold $t_c^{(l)}$ and $t_p^{(l)}$ for a given transformation $T_{X,l}$ from the image pairs $(x_i, x'_{i,j})$, the human predictions for these images, and their ground truth f^* . First, we compute Δ_v for each pair, and sort the pairs by their Δ_v into r intervals, defined by $r + 1$ equidistant thresholds t_k , with $t_0 = 0$ and $t_r = 1$. We then process the result using smoothing splines [28] to reduce randomness and remove outliers. Then, to estimate $t_c^{(l)}$, for each $k \in [1..r]$ we compute the probability p_k that m_k on the transformed images in the k -th interval $[t_{k-1}, t_k]$ is below m_0 . This probability is obtained using the single-sided binomial test. We then determine the interval with the smallest t_k for which $p_k \geq 0.05$, and return this t_k as $t_c^{(l)}$. Similarly, for $t_p^{(l)}$, we compute the probability p_k that s_k for the original-transformed image pairs in $[t_{k-1}, t_k]$ is below s_0 . Then $t_c^{(l)}$ is the smallest t_k for which $p_k \geq 0.05$.

With the above procedure, we can now estimate the parameters for the task of recognizing cars for each of the eight selected transformations (Fig. 1) using our experimental results. The instantiated parameters are shown in Tbl. 1.

With these parameters, we obtain the instantiated machine-verifiable reliability requirements for each transformation (step I.b). For example, for the transformation brightness, given an MVC that recognizes cars in images, the *correctness-preservation* requirement says that the MVC's recognition accuracy should not decrease if the visual change in the images is within the range $\Delta_v \leq 0.8$; and the *prediction-preservation* requirement says that the percentage of labels humans can preserve after a brightness change should not decrease if the visual change in the images is within the range $\Delta_v \leq 0.86$. To obtain the instantiated requirements for other transformations, only the parameter values need to be replaced with the estimated values in Tbl. 1.

6 CHECKING RELIABILITY REQUIREMENTS

In this section, we describe a method for automatically checking whether MVCs satisfy our machine-verifiable requirements (see steps II.a-c *requirement checking* in Fig. 2). Requirement checking takes as inputs a list of images, a set of transformations, and an MVC under validation. It generates test cases (step II.a) within the specified range of $\Delta_v \leq t_c$ or t_p ; runs the tests on the MVC

Table 1: Estimated parameters for correctness- (t_c) and prediction-preservation (t_p) requirements using human experiment results for the task of recognizing car instances.

	transformation	t_c	t_p	transformation	t_c	t_p
Imagenet	RGB	0.82	0.67	brightness	0.87	0.87
	contrast	0.77	0.28	Gaussian noise	0.91	0.91
	defocus blur	0.98	0.94	color jitter	0.48	0.48
	frost	0.84	0.91	jpeg compression	0.94	0.94
CIFAR-10	brightness	0.78	0.89	contrast	0.63	0.86
	frost	0.61	0.61	jpeg compression	0.60	0.60

(step II.b); and checks whether the MVC satisfies the requirements by estimating the reliability distance (step II.c).

We test the requirements satisfaction by estimating MVC performance or prediction preservation through sampling. This is necessarily, since we do not have direct access to P_X but only its samples. Our test case generation is based on the *bootstrap method* [13], which estimates the metrics m_0 , m_{t_c} , s_0 , and s_{t_p} through sampling the test data with replacement. Since these metrics are defined as expected values or means, by Central Limit Theorem, the values of these metrics computed for sample batches, denoted for each batch i as $\bar{m}_{0,i}$, $\bar{m}_{t_c,i}$, $\bar{s}_{0,i}$, and $\bar{s}_{t_p,i}$, respectively, are normally distributed. Following the bootstrap method, we obtain the population estimates by computing the means \hat{m}_0 , \hat{m}_{t_c} , \hat{s}_0 , and \hat{s}_{t_p} and the standard deviations $\sigma_{\hat{m}_0}$, $\sigma_{\hat{m}_{t_c}}$, $\sigma_{\hat{s}_0}$, and $\sigma_{\hat{s}_{t_p}}$ of the respective batch value sets $\{\bar{m}_{0,i}\}$, $\{\bar{m}_{t_c,i}\}$, $\{\bar{s}_{0,i}\}$, and $\{\bar{s}_{t_p,i}\}$. To do so, for each transformation T_X and a list of original images X , we sample n batches of k images from X and then generate a transformed image by applying T_X to each sampled original image with randomly sampled parameter values while ensuring the required Δ_v range, i.e., $n \times k$ pairs in total. Since sampling is part of the process, n and k should be determined based on $|X|$, and the bigger they are, the more accurate the estimated results would be [13]. Although the lower-bound numbers for n and k are hard to determine, one can check whether the sampling is sufficient as the bootstrap method always converges with enough batches of samples for normal distributions [2]. A choice of n is considered sufficiently large if two separate runs with different random seeds result in similar estimated values. After generating the tests, our method runs them on the MVC under validation, and obtains the MVC predictions for all the original images and for each batch i of transformed images. We then compute the sample batch estimates of the four metrics, i.e., $\{\bar{m}_{0,i}\}$, $\{\bar{m}_{t_c,i}\}$, $\{\bar{s}_{0,i}\}$, and $\{\bar{s}_{t_p,i}\}$, and take mean ($\Delta\hat{m}$, $\Delta\hat{s}$) and standard deviation ($\sigma_{\Delta\hat{m}}$, $\sigma_{\Delta\hat{s}}$) of each set as the population estimates.

Finally, we want to show that the reliability distance for each requirement is zero or negative, i.e., $\Delta m \leq 0$ for correctness-preservation and $\Delta s \leq 0$ for prediction-preservation. Since our estimates from the previous step are normally distributed, their differences are also normally distributed. Thus, the reliability distance estimates have the following means and standard deviations: $\Delta\hat{m} = \hat{m}_0 - \hat{m}_{t_c}$ and $\sigma_{\Delta\hat{m}} = \sqrt{\sigma_{\hat{m}_0}^2 + \sigma_{\hat{m}_{t_c}}^2}$; and $\Delta\hat{s} = \hat{s}_0 - \hat{s}_{t_p}$ and $\sigma_{\Delta\hat{s}} = \sqrt{\sigma_{\hat{s}_0}^2 + \sigma_{\hat{s}_{t_p}}^2}$. To ensure that the reliability distances are zero or negative with a confidence $1 - \alpha = 95\%$, we use the right-handed confidence interval. Thus, $\Delta m \leq 0$ with confidence $1 - \alpha$ iff $\Delta\hat{m} + z_\alpha \sigma_{\Delta\hat{m}} \leq 0$, where z_α is the z-value corresponding to

an area α in the right tail of a standard normal distribution, with $z_{0.05} = 1.645$ for 95% confidence. Similarly, $\Delta s \leq 0$ with confidence $1 - \alpha$ iff $\Delta\hat{s} + z_\alpha \sigma_{\Delta\hat{s}} \leq 0$.

For example, to check whether ResNet50 satisfies our instantiated requirements for the transformation Gaussian noise (see Tbl. 1) for the task of recognizing cars, the testing method first generates tests with the original and the transformed images within the Δ_v range specified in the requirements. The original images are sampled from the ILSVRC'12 validation dataset using bootstrap with $n = 200$, $k = 50$ and the Gaussian noise transformation. Then we run the generated tests on ResNet50; compute the four sets of metrics $\{\bar{m}_{0,i}\}$, $\{\bar{m}_{t_c,i}\}$, $\{\bar{s}_{0,i}\}$, and $\{\bar{s}_{t_p,i}\}$ over the batches; and then compute $\Delta\hat{m} = 0.0045$, $\sigma_{\Delta\hat{m}} = 0.0061$, $\Delta\hat{s} = 0.0011$, and $\sigma_{\Delta\hat{s}} = 0.0045$. We check for correctness-preservation with 95% confidence: $\Delta\hat{m} + z_{0.05} \sigma_{\Delta\hat{m}} > 0$; and prediction-preservation with 95% confidence: $\Delta\hat{s} + z_{0.05} \sigma_{\Delta\hat{s}} > 0$. Therefore ResNet50 does not satisfy either of the requirements. Note that by estimating the reliability distance, we provide engineers with a quantitative measure of how much improvement is needed to meet the requirements in case they are not met.

7 EVALUATION

While our approach is defined for any computer-vision task, in this paper we demonstrate its feasibility on a particular domain: image classification, using parameters instantiated via human performance data collected for this domain as explained in Sec. 6.

First, we evaluate the generality of our instantiated image classification requirements. For a specific transformation, our instantiated requirements contain the tolerated range of changes that do not affect human performance (see Sec. 4), estimated from experiments with human participants. Since such experiments are costly, we aim to minimize the number of experiments that need to be conducted. To achieve this goal, we would like to reuse the collected human performance results for new sets of images from the dataset, different from the ones presented to the humans during the experiments. We expect the images to come from the same dataset to share the underlying data-generating distribution P_X . Crucially, to be reusable, our requirement parameters should not be affected by the choice of the images included in the experiments with human participants. Since our requirements are defined on a particular distribution of images, we aim to answer **(RQ1)**: How reusable are the thresholds t_c and t_p over different samples from the same image distribution?

Second, existing methods for evaluating reliability of image classification MVCs consider either small, imperceptible image changes or an arbitrary range of perceptible changes in images. In this work, our focus is on a meaningful range of changes in images, the one that does not affect human vision, which includes both imperceptible and perceptible changes. Since our goal is to use human performance as a baseline (i.e., "if humans can see it, so should an MVC"), we are interested in understanding how well the existing reliability evaluation approaches already cover the human-tolerated range. We are also interested in comparing the distribution of test cases generated by our method (step II.a in Fig. 2) with those from the other reliability methods, to see whether our method addresses the range better. Therefore, we aim to answer **(RQ2)**: How well do the

existing reliability evaluation methods cover the human-tolerated range of changes?

Finally, we would like to determine whether checking the reliability of image classification MVC models with our method in the human-tolerated range of changes reveals reliability gaps in state-of-the-art image classification models. To do so, we aim to answer **(RQ3)**: How effective is our requirement checking method in identifying reliability gaps compared to existing approaches?

RQ1. To answer RQ1, we compare human-tolerated ranges of transformations (parameters t_c and t_p) estimated using our requirement instantiation method with *different* sets of images from the ILSVRC'12 experiment. We randomly selected two subsets of our results containing 60% of all the images included in the experiment. We compared the similarity of the spline models obtained using these two subsets with all experiment results. As suggested by Koenker et al. [28], two spline models are considered similar if their 83% confidence intervals overlap. Following this, for each of the eight transformation included in our experiment, we compared the confidence intervals of the estimated spline models representing the two subsets and the entire set of experiment results. As a result, for all transformations, we observed that the spline models are unaffected. For example, the spline models obtained for the frost transformation are shown in Fig. 4. Due to page limit, we include the plots and data in supplementary material¹. Since the parameters are derived using the spline models, unaffected spline models suggest that the parameters estimated are also unaffected. To conclude, different subsets of experiment results do not affect the parameters estimated. Therefore, we show evidence that our estimated human-tolerated ranges can be reused for images that are not included in the experiment with human participants, answering RQ1. Note that our requirements are defined on one image distribution, thus the thresholds cannot be reused for different image distributions. We can check this by comparing the values of t_c and t_p estimated using images from CIFAR-10 [30] and ILSVRC'12 [38], shown in Tbl. 1 (Sec. 5).

RQ2. Existing methods for evaluating the reliability of MVCs with image transformations (imperceptible and perceptible changes) include *metamorphic testing* [12, 32, 52, 56] and *benchmarking* [19]. Metamorphic testing is based on metamorphic relations; thus, instead of finding a range that does not affect human judgment, it consider all possible parameter values for each transformation included [12]. For this RQ, we compare with existing work that considers a broader range of changes: the state-of-the-art image corruption benchmark datasets Imagenet-c and CIFAR-10-c [19]. These benchmark datasets include images transformed with five pre-selected parameter values for 19 arbitrarily chosen transformations. Due to the low resolution of CIFAR-10 images, they look blurry to humans and thus do not share the same characteristics with the ILSVRC'12 dataset [38]. Therefore, to evaluate our method, we conduct an additional experiment with human participant using CIFAR-10 [30] images for four transformations (contrast, brightness, frost, and JPEG compression) and estimated the corresponding human-tolerated ranges, as described in Section 5. We answer RQ2 and RQ3 using six transformations considered by the other works: brightness, contrast, defocus blur, frost, Gaussian noise, and jpeg compression.

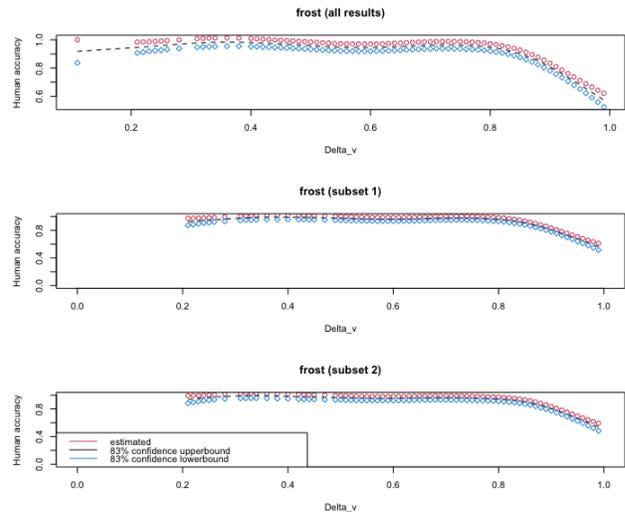


Figure 4: A comparison of different subsets of experimental results for estimating t_c for the frost transformation.

To answer RQ2, we first compare our human-tolerated ranges with the ranges of changes included in the robustness benchmark datasets, to see whether existing methods already cover them. In Fig. 5, we show, for each transformation, the range of changes in images included in Imagenet-c/ CIFAR-10-c [19]² in blue, and our human-tolerated ranges for both requirements in yellow and green. The overlapping of ranges indicates the degree to which our ranges are covered by Imagenet-c/ CIFAR-10-c. The ranges in Imagenet-c and CIFAR-10-c [19] are either larger (e.g., brightness and frost for CIFAR-10-c; brightness for Imagenet-c) or smaller (e.g., contrast and jpeg compression of CIFAR-10-c; Gaussian noise, defocus blur and frost for Imagenet-c) than the human-tolerated range. The images included in Imagenet-c/CIFAR-10-c are transformed by using a pre-selected list of five parameter values per transformation. This result shows that simply generating images this way does not address the full range of realistic changes that do not affect human performance. Secondly, we compare the distribution of the test cases (transformed images) within the human-tolerated range generated from our requirement checking method and from CIFAR-10-c and Imagenet-c. Our requirement checking method for generating test cases samples the parameter space uniformly and then transforms the images. As the number of parameters for a transformation increases, so does the possible number of combinations of parameter values that can lead to the same degree of visual change in the images. Therefore, sampling the parameter space uniformly allows us a better coverage of possible transformed images resulting in a fairer reliability evaluation compared with transformations with pre-selected parameter values, as done in CIFAR-10-c and Imagenet-c [19]. In Fig. 6, we show the distributions of transformed images generated with our requirement checking method and images in CIFAR-10-c and Imagenet-c. As we can observe from the plots, the transformed images from CIFAR-10-c and Imagenet-c either favor certain ranges

²Note that due to the large size of Imagenet-c, the distribution is obtained by uniformly sampling the entire benchmarking dataset.

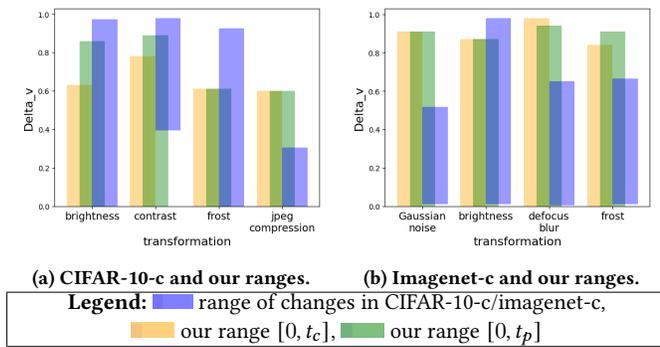


Figure 5: A comparison of our human-tolerated ranges for correctness-preservation and prediction-preservation requirements and the range of changes included in robustness benchmark datasets (Imagenet-c and CIFAR-10-c [19]).

of Δ_D score (Fig. 6b, 6c, 6d) or are discontinuous (Fig. 6a) and therefore biased. This also suggests that the approach of generating transformed images in the benchmark datasets does not guarantee a fair evaluation of reliability within the human-tolerated range of changes because of the biased distribution of tests. Thus, the human-tolerated ranges of changes are not addressed or properly tested by existing methods, answering RQ2.

RQ3. To answer RQ3, we aim to determine whether checking our requirements enables us to discover reliability gaps that were not identified with existing reliability benchmarks. We evaluated the reliability of 13 state-of-the-art image classification models with the vision task of recognizing cars in images using both our requirement checking method and the existing benchmarks CIFAR-10-c and Imagenet-c. Results are shown in Tbl. 2. Note that no models satisfy our requirements with 95% confidence, which is not surprising since they were not trained with data covering the human-tolerated range. However, several models pre-trained on Imagenet (ILSVRC’12) images have negative reliability distance ($\hat{s}_0 - \hat{s}_{t_p}$) for our *prediction-preservation* requirements, which suggests that these models are close to satisfying these requirements.

For each transformation, the models in Tbl. 2 are ranked based on the evaluation results (accuracy) of benchmark images. A higher ranking means that the model is more reliable. We compare the reliability ranking of these models using our reliability distance for both of our requirements (see Sec. 6) with the benchmark ranking, indicating the differences in blue. The tests included in the CIFAR-10-c and Imagenet-c benchmarks [19] are biased toward images with small transformation magnitudes, resulting in significant differences with our ranking for brightness (Fig. 6b), frost (Fig. 6c and 6h), jpeg compression (Fig. 6d) and defocus blur (Fig. 6g) transformations. Therefore, if a model is lower on the ranking of our reliability distance than on the benchmark ranking, it is less reliable than predicted by the benchmark, meaning that our method discovered a new reliability gap. Below we summarize the main reliability gaps identified by our method. (i) RLAT is ranked by CIFAR-10-c within the three last models for the transformations contrast, brightness, and frost, but the first for jpeg compression. However, our method ranks RLAT at the bottom for all the transformations including jpeg compression, indicating that the tests generated by our

method are able to detect the reliability gap missed by the benchmark. (ii) For jpeg compression, RLATAugMixNoJSD is ranked second both by the CIFAR-10-c benchmark and by our correctness-preservation reliability distance. However, RLATAugMixNoJSD is ranked last by our prediction-preservation reliability distance. Similarly, for Gaussian noise, resnext101_a+d is ranked first by both our *correction-preservation* and Imagenet-c benchmark, but it is ranked second last by our *prediction-preservation* reliability distance. This shows that both RLATAugMixNoJSD and resnext101_a+d have a high accuracy for transformed images but their predictions are not consistent. Therefore, checking our prediction-preservation requirement enabled us to identify new reliability gaps that could not be detected by only checking accuracy on transformed images, answering RQ3.

Summary. Through answering RQ1, we show that parameters of the requirements estimated with our requirement instantiation method are *reusable* for different images sharing the same class and images distribution. Through answering RQ2, we show that existing work does not adequately cover the range of changes that do not affect humans. Finally, through answering RQ3, we show that our requirement checking method is useful, since it can discover reliability gaps that are missed by the existing methods. Also, notice that our *prediction-preservation* is close to being satisfied by several models pre-trained on Imagenet (ILSVRC’12) images; this indicates that our requirements are satisfiable. Thus, our evaluation suggests that the proposed requirements are useful and reusable for checking reliability of MVCs.

Threats to validity. [Construct] For the correctness-preservation requirement, the human performance may seem too hard for MVCs to match. However, following guidelines provided by Firestone [14], we choose to keep the requirements for a fair comparison between a human and an ML performance. Further, training with data augmentation that covers the range of visual changes for each transformation as per our requirements might enable an MVC to meet them. Checking this hypothesis is future work. [Internal] We assumed that the parameter values for any transformation should be uniformly distributed. This may be different depending on the application of the MVC, e.g., heavy snow may be less relevant for autonomous cars deployed in tropical regions than other regions. [External] Due to budget considerations, we included a limited set of transformations and image classes in our experiments with humans. Experiments for other visual tasks are also future work.

8 RELATED WORK

In this section, we first review the software engineering (SE) approaches defining reliability of MVCs, then the SE and the computer vision (CV) approaches for evaluating reliability of MVCs and, finally, those comparing human performance against MVCs.

Specifying reliability of MVCs. The inductive data-driven nature of machine-learning creates several challenges for requirements specification and verification in MVCs. Yet, multiple recent studies explored this area [25, 36, 46]. While they agree on the necessity of requirements elicitation in MVCs, they fail to provide a systematic approach for inferring the requirements. Several authors attempted to specify the expected behaviour of MVCs indirectly through specifying a set of quality characteristics for training

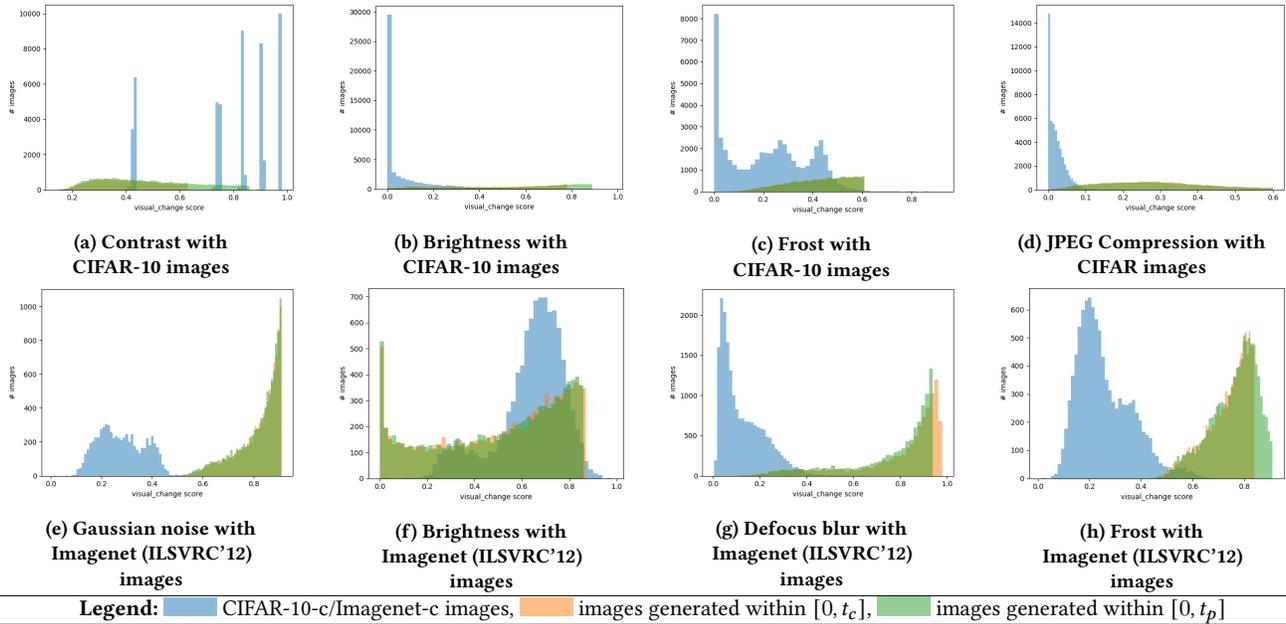


Figure 6: A comparison of the range and distribution of Δ_v scores in test images of robustness benchmark datasets (CIFAR-10-c and Imagenet-c [19]) and test images generated with our requirement checking method. The x-axis is the Δ_v score. The y-axis is the number of images.

Table 2: A comparison of reliability evaluation of MVCs using our method and using state-of-the-art benchmarks (CIFAR-10-c and Imagenet-c). For each transformation and the visual task of car recognition, the MVC models are ranked w.r.t. their accuracy on all benchmark images. \hat{m}_0 and \hat{s}_0 are, resp., the required accuracy and percentage of labels preserved in our requirements. \hat{m}_{t_c} and \hat{s}_{t_p} are, resp., the resulting accuracy and perception preservation percentage through checking the models against our requirements. The differences between the benchmark ranking and our ranking using the reliability distance are highlighted in blue.

dataset	model name	accuracy on all benchmark images	Checking our Correctness-preservation required and estimated accuracy $\hat{m}_0 \hat{m}_{t_c}$			Checking our Prediction-preservation required and estimated percentage $\hat{s}_0 \hat{s}_{t_p}$			model name	accuracy on all benchmark images	Checking our Correctness-preservation required and estimated accuracy $\hat{m}_0 \hat{m}_{t_c}$			Checking our Prediction-preservation required and estimated percentage $\hat{s}_0 \hat{s}_{t_p}$		
			reliability distance $\hat{m}_0 - \hat{m}_{t_c}$ (rank)			reliability distance $\hat{s}_0 - \hat{s}_{t_p}$ (rank)					reliability distance $\hat{m}_0 - \hat{m}_{t_c}$ (rank)			reliability distance $\hat{s}_0 - \hat{s}_{t_p}$ (rank)		
cifar-10	contrast															
	Augmix_ResNeXt [20]	0.9920	0.9961 0.9871	(2) 0.009	0.996 0.9761	(2) 0.0199	Augmix_ResNeXt [20]	0.9952	0.9963 0.9776	(3) 0.0187	0.999 0.9576	(3) 0.0414				
	Augmix_WRN [20]	0.9909	0.9952 0.9868	(1) 0.0084	0.995 0.9756	(1) 0.0194	AugMixNoJSD [27]	0.9945	0.9961 0.9782	(1) 0.0179	0.996 0.9573	(1) 0.0387				
	AugMixNoJSD [27]	0.9901	0.9952 0.9851	(3) 0.0101	0.997 0.9674	(3) 0.0296	Augmix_WRN [20]	0.9943	0.9953 0.9768	(2) 0.0185	0.994 0.9540	(2) 0.04				
	Standard [54]	0.9862	0.9952 0.9809	(4) 0.0143	0.994 0.9570	(4) 0.037	RLATAugMixNoJSD [27]	0.9942	0.9953 0.9730	(4) 0.0223	0.998 0.9488	(4) 0.0492				
	RLATAugMixNoJSD [27]	0.9788	0.9936 0.9710	(5) 0.0226	0.993 0.9416	(5) 0.0514	Standard [54]	0.9933	0.9947 0.9690	(5) 0.0257	0.998 0.9451	(5) 0.0529				
	Gauss50percent [27]	0.9577	0.9925 0.9261	(6) 0.0664	0.987 0.8994	(6) 0.0876	RLAT [27]	0.9928	0.9930 0.9557	(7) 0.0373	0.993 0.9307	(6) 0.0623				
	RLAT [27]	0.9550	0.9936 0.9133	(7) 0.0803	0.991 0.8880	(7) 0.103	Gauss50percent [27]	0.9904	0.9925 0.9555	(6) 0.037	0.995 0.9305	(7) 0.0645				
	frost															
	Augmix_ResNeXt [20]	0.9912	0.9969 0.9771	(2) 0.0198	0.995 0.9776	(1) 0.0174	RLAT [27]	0.9910	0.9927 0.9443	(5) 0.0484	0.999 0.9773	(1) 0.0217				
	RLATAugMixNoJSD [27]	0.9910	0.9958 0.9737	(4) 0.0221	0.994 0.9738	(2) 0.0202	RLATAugMixNoJSD [27]	0.9899	0.9942 0.9659	(2) 0.0283	0.999 0.9365	(7) 0.0625				
	Augmix_WRN [20]	0.9899	0.9955 0.9765	(1) 0.019	0.998 0.9758	(4) 0.0222	Gauss50percent [27]	0.9897	0.9915 0.9701	(1) 0.0214	0.999 0.9735	(2) 0.0255				
	AugMixNoJSD [27]	0.9890	0.9965 0.9754	(3) 0.0211	0.997 0.9754	(3) 0.0216	Augmix_ResNeXt [20]	0.9894	0.9949 0.9516	(4) 0.0433	0.999 0.9529	(4) 0.0461				
	RLAT [27]	0.9875	0.9948 0.9414	(7) 0.0534	0.986 0.9430	(7) 0.043	Augmix_WRN [20]	0.9886	0.9942 0.9511	(3) 0.0431	0.999 0.9547	(3) 0.0443				
Gauss50percent [27]	0.9867	0.9933 0.9506	(6) 0.0427	0.99 0.9524	(5) 0.0376	AugMixNoJSD [27]	0.9868	0.9953 0.9443	(6) 0.051	0.998 0.9471	(5) 0.0509					
Standard [54]	0.9752	0.9956 0.9567	(5) 0.0389	0.997 0.9564	(6) 0.0406	Standard [54]	0.9734	0.9952 0.9359	(7) 0.0593	0.996 0.9365	(6) 0.0595					
imagenet	Gaussian noise															
	resnext101_a+d [18]	0.9962	0.997 0.9959	(1) 0.0011	0.998 0.997	(5) 0.001	resnext101_a+d [18]	0.9958	0.9974 0.9954	(1) 0.002	0.996 0.9962	(1) -0.0002				
	aug+deep [18]	0.9956	0.9958 0.9942	(2) 0.0016	0.996 0.9961	(1) -0.0001	aug+deep [18]	0.9952	0.9967 0.9943	(2) 0.0024	0.996 0.9942	(5) 0.0018				
	deepaugmt [18]	0.9955	0.9963 0.9937	(4) 0.0026	0.996 0.9959	(3) 0.0001	ANT3x3_SIN [37]	0.9944	0.996 0.9935	(3) 0.0025	0.992 0.9924	(1) -0.0004				
	ANT_SIN [37]	0.9946	0.9953 0.9935	(3) 0.0018	0.996 0.9962	(1) -0.0002	ANT_SIN [37]	0.9941	0.9962 0.9927	(4) 0.0035	0.99 0.9927	(1) -0.0027				
	Speckle_Model [37]	0.9934	0.9958 0.9916	(5) 0.0042	0.996 0.9952	(4) 0.0008	deepaugmt [18]	0.9936	0.9966 0.993	(5) 0.0036	0.994 0.992	(6) 0.002				
	resnet50 [17]	0.9924	0.9953 0.9908	(6) 0.0045	0.996 0.9949	(6) 0.0011	resnet50 [17]	0.9921	0.9957 0.9907	(6) 0.005	0.992 0.9911	(4) 0.0009				
	brightness															
	resnext101_a+d [18]	0.9972	0.9967 0.9953	(2) 0.0014	1 0.9972	(4) 0.0028	resnext101_a+d [18]	0.9949	0.9977 0.995	(1) 0.0027	0.994 0.9957	(1) -0.0017				
	aug+deep [18]	0.9966	0.9959 0.9947	(1) 0.0012	1 0.9964	(5) 0.0036	aug+deep [18]	0.9946	0.9972 0.9937	(2) 0.0035	0.996 0.9943	(2) 0.0017				
	deepaugmt [18]	0.9959	0.9956 0.9937	(3) 0.0019	0.996 0.9949	(2) 0.0011	deepaugmt [18]	0.9924	0.9965 0.9914	(5) 0.005	0.998 0.9929	(5) 0.0051				
	ANT3x3_SIN [37]	0.9957	0.9954 0.9926	(5) 0.0028	0.996 0.994	(3) 0.002	ANT_SIN [37]	0.9920	0.997 0.9917	(6) 0.053	0.998 0.9929	(5) 0.0051				
	ANT_SIN [37]	0.9956	0.9954 0.993	(4) 0.0024	0.993 0.998	(1) -0.005	ANT3x3_SIN [37]	0.9919	0.9963 0.9924	(3) 0.0036	0.998 0.9931	(4) 0.0049				
	resnet50 [17]	0.995	0.9956 0.9917	(6) 0.0039	1 0.9937	(6) 0.0063	resnet50 [17]	0.9909	0.9961 0.9921	(4) 0.0040	0.996 0.9922	(3) 0.0038				
defocus blur																
resnext101_a+d [18]	0.9962	0.997 0.9959	(1) 0.0011	0.998 0.997	(5) 0.001	resnext101_a+d [18]	0.9958	0.9974 0.9954	(1) 0.002	0.996 0.9962	(1) -0.0002					
aug+deep [18]	0.9956	0.9958 0.9942	(2) 0.0016	0.996 0.9961	(1) -0.0001	aug+deep [18]	0.9952	0.9967 0.9943	(2) 0.0024	0.996 0.9942	(5) 0.0018					
deepaugmt [18]	0.9955	0.9963 0.9937	(4) 0.0026	0.996 0.9959	(3) 0.0001	ANT3x3_SIN [37]	0.9944	0.996 0.9935	(3) 0.0025	0.992 0.9924	(1) -0.0004					
ANT_SIN [37]	0.9946	0.9953 0.9935	(3) 0.0018	0.996 0.9962	(1) -0.0002	ANT_SIN [37]	0.9941	0.9962 0.9927	(4) 0.0035	0.99 0.9927	(1) -0.0027					
Speckle_Model [37]	0.9934	0.9958 0.9916	(5) 0.0042	0.996 0.9952	(4) 0.0008	deepaugmt [18]	0.9936	0.9966 0.993	(5) 0.0036	0.994 0.992	(6) 0.002					
resnet50 [17]	0.9924	0.9953 0.9908	(6) 0.0045	0.996 0.9949	(6) 0.0011	resnet50 [17]	0.9921	0.9957 0.9907	(6) 0.005	0.992 0.9911	(4) 0.0009					

Note: Accuracy is calculated with (true positive + true negative) / all images; all the accuracy values are closed to 1 because of the binary classification task. All numbers are rounded.

datasets [29], specifying additional ML-related requirements for each phase of software development processes [39], specifying higher-level requirements [15], or specifying how MVCs address the target applications [41]. Yet these approaches cannot be used to check reliability of MVCs automatically, whereas our reliability requirements are machine-verifiable.

Checking reliability. Metrics for testing MVCs robustness against image transformations have been defined using metamorphic testing [12, 32, 52, 56]. In contrast, we focus on a different set of transformations, the ones that do degrade the image quality while preserving the human opinion in the image rather than transformations that can be covered with equivariant relations. Existing works also use testing to generate corner cases [51], or corner case tests to increase MVCs robustness [31, 47]. In contrast, our approach does not focus on corner-cases, but rather on typical cases that can be found in real-world deployments while preserving the human opinion about the content.

Several works evaluated safety of MVCs through assessing their robustness against adversarial examples, either by providing a testing approach to generate adversarial examples [33, 40, 50] in the SE area, providing robustness benchmarks [10, 11, 34] or verifying the presence of adversarial examples in a given range of image modifications [23, 44] in the CV area. In contrast, our focus is on defining boundaries of image modifications using human performance within which the MVCs are expected to maintain their robustness. Also, we do not consider an arbitrary range of image modifications; our approach estimates the range of transformation levels that does not affect human performance. Previously, we presented the idea of defining adversarial examples using IQA models [22], focusing only on non-visible changes. In contrast, our current approach considers both visible and non-visible changes in a broader range of real world scenarios.

Comparing human against machines. Prior studies also referred to human performance as the benchmark for the evaluation of their proposed methods [45], to better study the existing differences between human and neural networks [14], to study invariant transformations [26], to compare recognition accuracy [21], or to compare robustness [16]. In contrast, our focus is not on comparing humans performance with MVCs, but rather on the ranges of transformation magnitudes that do not affect human performance.

9 CONCLUSION

In this paper, we defined reliability of machine vision components (MVC) as ‘if a human can see it, so should the MVC’. More precisely, we specified two classes of reliability requirements: correctness-preservation and prediction-preservation. Our requirements specify that an MVC should be reliably unaffected by safety-related image transformations, at least within the range of changes that does not affect humans. We showed, through an evaluation with 13 state-of-the-art pre-trained image classification models, that our approach captures reliability gaps that state-of-the-art reliability methods are unable to detect. Therefore, we conclude that checking this human tolerated range is important to help software engineers ensure quality and reliability of MVCs. While not discussed in the paper, our requirements can be used for other SE tasks such as checking refinement from higher-level system requirements, and

checking consistency and compatibility with requirements of other connected components.

In the future, we aim to improve of our requirement-checking process by providing reliability diagnosis that would help software engineers understand the reliability gaps in their MVCs. We also aim to validate, through additional experiments, our assumption that our approach can be applicable beyond image classification models, e.g., to handle object detection. Finally, we aim to use our reliability requirements for MVCs to provide evidence for building safety assurance cases for the overall system.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their feedback and insightful comments. We also thank all the MTurkers for participating in our experiments; Dr. Dimitrios Papadopoulos for providing an MTurk experiment implementation that formed the basis of our experiments; Professor Radu Craiu for his comments on the statistical methods; Dr. Nikita Dvornik, Nick Feng, Dr. Mona Rahimi, Valentina Manferrari, Dr. Ramy Shahin, Alexander Tough, and Dr. Shurui Zhou for helping improve this manuscript; and Valentina Manferrari for her assistance during an earlier version of the experiments.

REFERENCES

- [1] IEEE Standard Glossary of Software Engineering Terminology. *IEEE Std 610-12-1990*, pages 1–84, 1990.
- [2] K. B. Athreya. Bootstrap of the Mean in the Infinite Variance Case. *The Annals of Statistics*, 15(2):724–731, 1987.
- [3] Osbert Bastani, Yani A Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya V. Nori, and Antonio Criminisi. Measuring Neural Net Robustness with Constraints. In *NeurIPS’16*, 2016.
- [4] N Boudette. ‘It Happened So Fast’: Inside a Fatal Tesla Autopilot Accident. <https://www.nytimes.com/2021/08/17/business/tesla-autopilot-accident.html>, August 2021.
- [5] N Boudette and N. Chokshi. U.S. Will Investigate Tesla’s Autopilot System Over Crashes With Emergency Vehicles. <https://www.nytimes.com/2021/08/16/business/tesla-autopilot-nhtsa.html>, August 2021.
- [6] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albuminations: Fast and Flexible Image Augmentations. *Information*, 11(2), 2020.
- [7] D. M. Chandler and S. S. Hemami. VSNR: A Wavelet-Based Visual Signal-to-Noise Ratio for Natural Images. *IEEE Trans. on Image Processing*, 16(9):2284–2298, 2007.
- [8] Chiara Picardi and Colin Paterson and Richard Hawkins and Radu Calinescu and Ibrahim Habli. Assurance Argument Patterns and Processes for Machine Learning in Safety-Related Systems. In *SafeAI@AAAI*, volume 2560 of *CEUR Workshop Proceedings*, pages 23–30. CEUR-WS.org, 2020.
- [9] Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified Adversarial Robustness via Randomized Smoothing. *ArXiv*, abs/1902.02918, 2019.
- [10] Francesco Croce, Maksym Andriushchenko, Vikash Sehwal, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. RobustBench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.
- [11] Yinpeng Dong, Qi-An Fu, Xiao Yang, Tianyu Pang, Hang Su, Zihao Xiao, and Jun Zhu. Benchmarking Adversarial Robustness on Image Classification. In *Proc. of CVPR’20*, pages 321–331, 2020.
- [12] Anurag Dwarakanath, Manish Ahuja, Samarth Sikand, Raghotham M. Rao, R. P. Jagadeesh Chandra Bose, Neville Dubash, and Sanjay Podder. Identifying implementation bugs in machine learning based image classifiers using metamorphic testing. *ISSTA 2018*, page 118–128, New York, NY, USA, 2018. Association for Computing Machinery.
- [13] Bradley Efron and Robert J Tibshirani. *An Introduction to the Bootstrap*. CRC press, 1994.
- [14] Chaz Firestone. Performance vs. Competence in Human–Machine Comparisons. volume 117, pages 26562–26571. National Academy of Sciences, 2020.
- [15] Lydia Gauerhof, Richard Hawkins, Chiara Picardi, Colin Paterson, Yuki Hagiwara, and Ibrahim Habli. Assuring the Safety of Machine Learning for Pedestrian Detection at Crossings. In *Proc. of SAFECOMP’20*, pages 197–212, 2020.

- [16] R Geirhos, CR Medina Temme, J Rauber, HH Schütt, M Bethge, and FA Wichmann. Generalisation in Humans and Deep Neural Networks. In *Proc. of NeurIPS'18*, pages 7549–7561. Curran, 2019.
- [17] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [18] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Lixuan Zhu, Samyak Parajuli, Michael Guo, D. Song, J. Steinhardt, and J. Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ArXiv*, abs/2006.16241, 2020.
- [19] Dan Hendrycks and Thomas Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. *Proc. of ICLR'19*, 2019.
- [20] Dan Hendrycks, Norman Mu, E. D. Cubuk, Barret Zoph, J. Gilmer, and Balaji Lakshminarayanan. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty. *ArXiv*, abs/1912.02781, 2020.
- [21] T. Ho-Phuoc. CIFAR10 to Compare Visual Recognition Performance between Deep Neural Networks and Humans. *ArXiv*, abs/1811.07270, 2018.
- [22] B. C. Hu, R. Salay, K. Czarnecki, M. Rahimi, G. Selim, and M. Chechik. Towards Requirements Specification for Machine-learned Perception Based on Human Performance. In *AIRE'20*, pages 48–51, 2020.
- [23] Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety Verification of Deep Neural Networks. In *CAV'17*, pages 3–29, 2017.
- [24] Fuyuki Ishikawa and Nobukazu Yoshioka. How Do Engineers Perceive Difficulties in Engineering of Machine-learning Systems?: Questionnaire Survey. In Marcus Ciolkowski, Dusia Marijan, Matthias Galster, Weiyi Shang, Andreas Jedlitschka, Rakesh Shukla, and Kanchara Padmanabhan, editors, *CESSER-IP@ICSE 2019*, pages 2–9. IEEE / ACM, 2019.
- [25] H. Kaindl and J. Ferdigg. Towards an Extended Requirements Problem Formulation for Superintelligence Safety. In *Proc. of AIRE'20*, pages 33–38, 2020.
- [26] Saeed Reza Kheradpisheh, Masoud Ghodrati, Mohammad Ganjtash, and Timothée Masquelier. Deep Networks Can Resemble Human Feed-forward Vision in Invariant Object Recognition. *Scientific reports*, 6(1):1–24, 2016.
- [27] Klim Kireev, Maksym Andriushchenko, and Nicolas Flammarion. On the Effectiveness of Adversarial Training Against Common Corruptions. *ArXiv*, abs/2103.02325, 2021.
- [28] Roger Koehn, Pin Ng, and Stephen Portnoy. Quantile smoothing splines. *Biometrika*, 81(4):673–680, 1994.
- [29] Marc Kohli, Ronald Summers, and Jr Geis. Medical Image Data and Datasets in the Era of Machine Learning. *JDI*, 30(4):392–399, 2017.
- [30] Alex Krizhevsky, Geoffrey Hinton, et al. Learning Multiple Layers of Features from Tiny Images. 2009.
- [31] Seokhyun Lee, Sooyoung Cha, Dain Lee, and Hakjoo Oh. Effective White-Box Testing of Deep Neural Networks with Adaptive Neuron-Selection Strategy. *ISSTA 2020*, page 165–176, New York, NY, USA, 2020. Association for Computing Machinery.
- [32] Rohan Reddy Mekala, Gudjon Einar Magnusson, Adam Porter, Mikael Lindvall, and Madeline Diep. Metamorphic detection of adversarial examples in deep learning models with affine transformations. In *Proceedings of the 4th International Workshop on Metamorphic Testing (MET@ICSE'19)*, Montreal, QC, Canada, pages 55–62. IEEE / ACM, 2019.
- [33] Marco Melis, Ambra Demontis, Battista Biggio, Gavin Brown, Giorgio Fumera, and Fabio Roli. Is Deep Learning Safe for Robot Vision? Adversarial Examples against the Icube Humanoid. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 751–759, 2017.
- [34] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S. Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking Robustness in Object Detection: Autonomous Driving when Winter is Coming. *arXiv preprint arXiv:1907.07484*, 2019.
- [35] Dim P. Papadopoulos, Jasper R. R. Uijlings, Frank Keller, and Vittorio Ferrari. Training Object Class Detectors with Click Supervision. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017*, pages 180–189. IEEE Computer Society, 2017.
- [36] Mona Rahimi, Jin L. C. Guo, Sahar Kokaly, and Marsha Chechik. Toward Requirements Specification for Machine-Learned Components. In *AIRE'19*, pages 241–244, 2019.
- [37] E. Rusak, Lukas Schott, Roland S. Zimmermann, Julian Bitterwolf, O. Bringmann, M. Bethge, and Wieland Brendel. Increasing the robustness of dnns against image corruptions by playing the game of noise. *ArXiv*, abs/2001.06057, 2020.
- [38] Olga Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Zhiheng Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115:211–252, 2015.
- [39] Rick Salay and Krzysztof Czarnecki. Using Machine Learning Safely in Automotive Software: An Assessment and Adaption of Software Process Requirements in ISO 26262. *ArXiv*, abs/1808.01614, 2018.
- [40] Alex Serban, Erik Poll, and Joost Visser. Adversarial Examples on Object Recognition: A Comprehensive Survey. *ACM Computing Surveys (CSUR)*, 53(3):1–38, 2020.
- [41] Sanjit A. Seshia and Dorsa Sadigh. Towards Verified Artificial Intelligence. *ArXiv*, abs/1606.08514, 2016.
- [42] H. R. Sheikh and A. C. Bovik. Image information and visual quality. *IEEE Transactions on Image Processing*, 15(2):430–444, 2006.
- [43] H. R. Sheikh, M. F. Sabir, and A. C. Bovik. A Statistical Evaluation of Recent Full Reference Image Quality Assessment Algorithms. *IEEE Transactions on Image Processing*, 15(11):3440–3451, 2006.
- [44] Arvind Kumar Shekar, Liang Gou, Liu Ren, and Axel Wendt. Label-Free Robustness Estimation of Object Detection CNNs for Autonomous Driving Applications. *International Journal of Computer Vision*, 129(4):1185–1201, 2021.
- [45] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323 – 332, 2012. Selected Papers from IJCNN 2011.
- [46] Andreas Vogelsang and Markus Borg. Requirements Engineering for Machine Learning: Perspectives from Data Scientists. In *AIRE'19*, pages 245–251. IEEE, 2019.
- [47] Jingyi Wang, Jialuo Chen, Youcheng Sun, Xingjun Ma, Dongxia Wang, Jun Sun, and Peng Cheng. ROBOT: Robustness-Oriented Testing for Deep Learning Systems. *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 300–311, 2021.
- [48] Zhou Wang and Alan C. Bovik. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE Signal Processing Magazine*, 26(1):98–117, 2009.
- [49] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. on Image Processing*, 13(4):600–612, 2004.
- [50] Matthew Wicker, Xiaowei Huang, and Marta Kwiatkowska. Feature-Guided Black-Box Safety Testing of Deep Neural Networks. In *Proc. of TACAS'18*, pages 408–426. Springer, 2018.
- [51] Weibin Wu, Hui Xu, Sanqiang Zhong, Michael R. Lyu, and Irwin King. Deep Validation: Toward Detecting Real-World Corner Cases for Deep Neural Networks. In *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 125–137, 2019.
- [52] Xiaoyuan Xie, Joshua Wing Kei Ho, Christian Murphy, Gail E. Kaiser, Baowen Xu, and Tsong Yueh Chen. Testing and Validating Machine Learning Classifiers by Metamorphic Testing. *Journal of Systems and Software*, 84(4):544–558, 2011.
- [53] S. Xu, J. Wang, W. Shou, T. Ngo, A. Sadick, and X. Wang. Computer Vision Techniques in Construction: a Critical Review. *Archives of Computational Methods in Engineering*, pages 1–15, 2020.
- [54] Sergey Zagoruyko and N. Komodakis. Wide residual networks. *ArXiv*, abs/1605.07146, 2016.
- [55] Oliver Zendel, Markus Murschitz, Martin Humenberger, and Wolfgang Herzner. CV-HAZOP: Introducing Test Data Validation for Computer Vision. In *ICCV'15*, pages 2066–2074, 2015.
- [56] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. 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)*, pages 132–142. IEEE, 2018.