TECHNICAL REPORT

ISO/IEC TR 29119-11

First edition

Software and systems engineering — Software testing —

Part 11: Testing of AI-based systems

PROOF/ÉPREUVE

Reference number
ISO/IEC TR 29119-11:2020(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*.

A list of all parts in the ISO/IEC/IEEE 29119 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

The testing of traditional systems is well-understood, but AI-based systems, which are becoming more prevalent and critical to our daily lives, introduce new challenges. This document has been created to introduce AI-based systems and provide guidelines on how they might be tested.

Annex A provides an introduction to machine learning.

This document is primarily provided for those testers who are new to AI-based systems, but it can also be useful for more experienced testers and other stakeholders working on the development and testing of AI-based systems.

As a Technical Report, this document contains data of a different kind from that normally published as an International Standard or Technical Specification, such as data on the “state of the art”.

Software and systems engineering — Software testing —

Part 11:
Testing of AI-based systems

1 Scope

This document provides an introduction to AI-based systems. These systems are typically complex (e.g. deep neural nets), are sometimes based on big data, can be poorly specified and can be non-deterministic, which creates new challenges and opportunities for testing them.

This document explains those characteristics which are specific to AI-based systems and explains the corresponding difficulties of specifying the acceptance criteria for such systems.

This document presents the challenges of testing AI-based systems, the main challenge being the test oracle problem, whereby testers find it difficult to determine expected results for testing and therefore whether tests have passed or failed. It covers testing of these systems across the life cycle and gives guidelines on how AI-based systems in general can be tested using black-box approaches and introduces white-box testing specifically for neural networks. It describes options for the test environments and test scenarios used for testing AI-based systems.

In this document an AI-based system is a system that includes at least one AI component.

2 Normative references

There are no normative references in this document.

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1.1
A/B testing
split-run testing
statistical testing approach that allows testers to determine which of two systems or components performs better

3.1.2
activation value
<*neural network* (3.1.49)> output of an *activation function* (3.1.3) of a node in a neural network

3.1.3
activation function
transfer function
<*neural network* (3.1.49)> formula associated with a node in a neural network that determines the output of the node (*activation value* (3.1.2)) from the inputs to the neuron

3.1.4
adaptability
ability of a system to react to changes in its environment in order to continue meeting both functional and non-functional requirements

3.1.5
adversarial attack
deliberate use of *adversarial examples* (3.1.6) to cause a *ML model* (3.1.46) to fail

Note 1 to entry: Typically targets ML models in the form of a *neural network* (3.1.49).

3.1.6
adversarial example
input to an *ML model* (3.1.46) created by applying small perturbations to a working example that results in the model outputting an incorrect result with high confidence

Note 1 to entry: Typically applies to ML models in the form of a *neural network* (3.1.49).

3.1.7
adversarial testing
testing approach based on the attempted creation and execution of *adversarial examples* (3.1.6) to identify defects in an *ML model* (3.1.46)

Note 1 to entry: Typically applied to ML models in the form of a *neural network* (3.1.49).

3.1.8
AI-based system
system including one or more components implementing *AI* (3.1.12)

3.1.9
AI effect
situation when a previously labelled *AI* (3.1.12) system is no longer considered to be AI as technology advances

3.1.10
AI quality metamodel
metamodel intended to ensure the quality of *AI-based systems* (3.1.8)

Note 1 to entry: This metamodel is defined in detail in DIN SPEC 92001.

3.1.11
algorithm
ML algorithm
<*machine learning* (3.1.43)> algorithm used to create an *ML model* (3.1.46) from the *training data* (3.1.80)

EXAMPLE ML algorithms include linear *regression* (3.1.62), logistic regression, *decision tree* (3.1.25), SVM, Naive Bayes, kNN, K-means and random forest.

3.1.12
artificial intelligence
AI
capability of an engineered system to acquire, process and apply knowledge and skills

3.1.13
autonomous system
system capable of working without human intervention for sustained periods

3.1.14
autonomy
ability of a system to work for sustained periods without human intervention

3.1.15
back-to-back testing
differential testing
approach to testing whereby an alternative version of the system is used as a *pseudo-oracle* (3.1.59) to generate expected results for comparison from the same test inputs

EXAMPLE The pseudo-oracle may be a system that already exists, a system developed by an independent team or a system implemented using a different programming language.

3.1.16
backward propagation
<*neural network* (3.1.49)> method used in artificial neural networks to determine the weights to be used on the network connections based on the computed error at the output of the network

Note 1 to entry: It is used to train *deep neural networks* (3.1.27).

3.1.17
benchmark suite
collection of benchmarks, where a benchmark is a set of tests used to compare the performance of alternatives

3.1.18
bias
<*machine learning* (3.1.43)> measure of the distance between the predicted value provided by the *ML model* (3.1.46) and a desired fair *prediction* (3.1.57)

3.1.19
classification
<*machine learning* (3.1.43)> machine learning function that predicts the output class for a given input

3.1.20
classifier
<*machine learning* (3.1.43)> *ML model* (3.1.46) used for *classification* (3.1.19)

3.1.21
clustering
grouping of a set of objects such that objects in the same group (i.e. a cluster) are more similar to each other than to those in other clusters

3.1.22
combinatorial testing
black-box test design technique in which test cases are designed to execute specific combinations of values of several *parameters* (3.1.54)

EXAMPLE *Pairwise testing* (3.1.53), all combinations testing, each choice testing, base choice testing.

3.1.23
confusion matrix
table used to describe the performance of a classifier (3.1.20) on a set of *test data* (3.1.75) for which the true and false values are known

3.1.24
pre-processing
<*machine learning* (3.1.43)> part of the ML workflow that transforms raw data into a state ready for use by the *ML algorithm* (3.1.11) to create the *ML model* (3.1.46)

Note 1 to entry: Pre-processing can include analysis, normalization, filtering, reformatting, imputation, removal of outliers and duplicates, and ensuring the completeness of the dataset.

3.1.25
decision tree
<*machine learning* (3.1.43)> supervised-learning *model* (3.1.46) for which inference can be represented by traversing one or more tree-like structures

3.1.26
deep learning
approach to creating rich hierarchical representations through the training of *neural networks* (3.1.49) with one or more hidden layers

Note 1 to entry: Deep learning uses multi-layered networks of simple computing units (or "neurons"). In these neural networks each unit combines a set of input values to produce an output value, which in turn is passed on to other neurons downstream.

3.1.27
deep neural net
neural network (3.1.49) with more than two layers

3.1.28
deterministic system
system which, given a particular set of inputs and starting state, will always produce the same set of outputs and final state

3.1.29
distributional shift
dataset shift
<*machine learning* (3.1.43)> distance between the *training data* (3.1.80) distribution and the desired data distribution

Note 1 to entry: The effect of distributional shift often increases as the users' interaction with the system (and so the desired data distribution) changes over time.

3.1.30
drift
degradation
staleness
<*machine learning* (3.1.43)> changes to *ML model* (3.1.46) behaviour that occur over time

Note 1 to entry: These changes typically make *predictions* (3.1.57) less accurate and may require the model to be re-trained with new data.

3.1.31
explainability
<*AI* (3.1.12)> level of understanding how the *AI-based system* (3.1.8) came up with a given result

3.1.32
exploratory testing
experience-based testing in which the tester spontaneously designs and executes tests based on the tester's existing relevant knowledge, prior exploration of the test item (including the results of previous tests), and heuristic "rules of thumb" regarding common software behaviours and types of failure

Note 1 to entry: Exploratory testing hunts for hidden properties (including hidden behaviours) that, while quite possibly benign by themselves, could interfere with other properties of the software under test, and so constitute a risk that the software will fail.

3.1.33
F1-score
<*machine learning* (3.1.43)> performance metric used to evaluate a *classifier* (3.1.20), which provides a balance (the harmonic average) between *recall* (3.1.61) and *precision* (3.1.56)

3.1.34
false negative
incorrect reporting of a failure when in reality it is a pass

Note 1 to entry: This is also known as a Type II error.

EXAMPLE The referee awards an offside when it was a goal and so reports a failure to score a goal when a goal was scored.

3.1.35
false positive
incorrect reporting of a pass when in reality it is a failure

Note 1 to entry: This is also known as a Type I error.

EXAMPLE The referee awards a goal that was offside and so should not have been awarded.

3.1.36
feature engineering
feature selection
<*machine learning* (3.1.43)> activity in which those attributes in the raw data that best represent the underlying relationships that should appear in the *model* (3.1.46) are identified for use in the *training data* (3.1.80)

3.1.37
flexibility
ability of a system to work in contexts outside its initial specification (i.e. change its behaviour according to its actual situation to satisfy its objectives)

3.1.38
fuzz testing
software testing approach in which high volumes of random (or near random) data, called fuzz, are used to generate inputs to the test item

3.1.39
general AI
strong AI
AI (3.1.12) that exhibits intelligent behaviour comparable to a human across the full range of cognitive abilities

3.1.40
graphical processing unit
GPU
application-specific integrated circuit (ASIC) specialized for display functions such as rendering images

Note 1 to entry: GPUs are designed for parallel data processing of images with a single function, but this parallel processing is also useful for executing AI-based software, such as *neural networks* (3.1.49).

3.1.41
hyperparameter
<*neural network* (3.1.49)> variable used to define the structure of a neural network and how it is trained

Note 1 to entry: Typically, hyperparameters are set by the developer of the *model* (3.1.46) and may also be referred to as a tuning *parameter* (3.1.54).

3.1.42
interpretability
<*AI* (3.1.12)> level of understanding how the underlying (AI) technology works

3.1.43
machine learning
ML
process using computational techniques to enable systems to learn from data or experience

3.1.44
metamorphic relation
description of how a change in the test inputs from the source test case to the follow-up test case affects a change (or not) in the expected outputs from the source test case to the follow-up test case

3.1.45
metamorphic testing
testing where the expected results are not based on the specification but are instead extrapolated from previous actual results

3.1.46
model
ML model
<*machine learning* (3.1.43)> output of a *ML algorithm* (3.1.11) trained with a training dataset that generates *predictions* (3.1.57) using patterns in the input data

3.1.47
accuracy
<*machine learning* (3.1.43)> performance metric used to evaluate a *classifier* (3.1.20), which measures the proportion of *classifications* (3.1.19) *predictions* (3.1.57) that were correct

3.1.48
narrow AI
weak AI
AI (3.1.12) focused on a single well-defined task to address a specific problem

3.1.49
neural network
artificial neural network
network of primitive processing elements connected by weighted links with adjustable weights, in which each element produces a value by applying a nonlinear function to its input values, and transmits it to other elements or presents it as an output value

Note 1 to entry: Whereas some neural networks are intended to simulate the functioning of neurons in the nervous system, most neural networks are used in *artificial intelligence* (3.1.12) as realizations of the connectionist *model* (3.1.46).

Note 2 to entry: Examples of nonlinear functions are a threshold function, a sigmoid function, and a polynomial function.

[SOURCE: ISO/IEC 2382:2015, 2120625, modified — The admitted term "neural net" has been removed; notes 3 to 5 to entry have been removed.]

3.1.50
neuron coverage
proportion of activated neurons divided by the total number of neurons in the *neural network* (3.1.49) (normally expressed as a percentage) for a set of tests

Note 1 to entry: A neuron is considered to be activated if its *activation value* (3.1.2) exceeds zero.

3.1.51
non-deterministic system
system which, given a particular set of inputs and starting state, will not always produce the same set of outputs and final state

3.1.52
overfitting
<*machine learning* (3.1.43)> generation of a *ML model* (3.1.46) that corresponds too closely to the *training data* (3.1.80), resulting in a model that finds it difficult to generalize to new data

3.1.53
pairwise testing
black-box test design technique in which test cases are designed to execute all possible discrete combinations of each pair of input *parameters* (3.1.54)

Note 1 to entry: Pairwise testing is the most popular form of *combinatorial testing* (3.1.22).

3.1.54
parameter
<*machine learning* (3.1.43)> parts of the *model* (3.1.46) that are learnt from applying the *training data* (3.1.80) to the *algorithm* (3.1.11)

EXAMPLE Learnt weights in a neural net.

Note 1 to entry: Typically, parameters are not set by the developer of the model.

3.1.55
performance metrics
<*machine learning* (3.1.43)> metrics used to evaluate *ML models* (3.1.46) that are used for *classification* (3.1.19)

EXAMPLE Typical metrics include *accuracy* (3.1.47), *precision* (3.1.56), *recall* (3.1.61) and *F1-score* (3.1.33).

3.1.56
precision
<*machine learning* (3.1.43)> performance metric used to evaluate a *classifier* (3.1.20), which measures the proportion of predicted positives that were correct

3.1.57
prediction
<*machine learning* (3.1.43)> machine learning function that results in a predicted target value for a given input

EXAMPLE Includes *classification* (3.1.19) and *regression* (3.1.62) functions.

3.1.58
probabilistic system
system whose behaviour is described in terms of probabilities, such that its outputs cannot be perfectly predicted

3.1.59
pseudo-oracle
derived test oracle
independently derived variant of the test item used to generate results, which are compared with the results of the original test item based on the same test inputs

Note 1 to entry: Pseudo-oracles are a useful alternative when traditional *test oracles* (3.1.76) are not available.

3.1.60
reasoning technique
<*AI* (3.1.12)> form of AI that generates conclusions from available information using logical techniques, such as deduction and induction