

# Black-box Error Diagnosis in Deep Neural Networks for Computer Vision: a Survey of Tools

Piero Fraternali<sup>1†</sup>, Federico Milani<sup>1\*†</sup>, Rocio Nahime Torres<sup>1†</sup> and Niccolò Zangrando<sup>1†</sup>

<sup>1</sup>Department of Electronics, Information and Bioengineering (DEIB), Politecnico di Milano, Via Giuseppe Ponzio, 34, Milan, 20133, MI, Italy.

\*Corresponding author(s). E-mail(s): [federico.milani@polimi.it](mailto:federico.milani@polimi.it);

Contributing authors: [piero.fraternali@polimi.it](mailto:piero.fraternali@polimi.it); [rocionahime.torres@polimi.it](mailto:rocionahime.torres@polimi.it); [niccolo.zangrando@polimi.it](mailto:niccolo.zangrando@polimi.it);

<sup>†</sup>These authors contributed equally to this work.

## Abstract

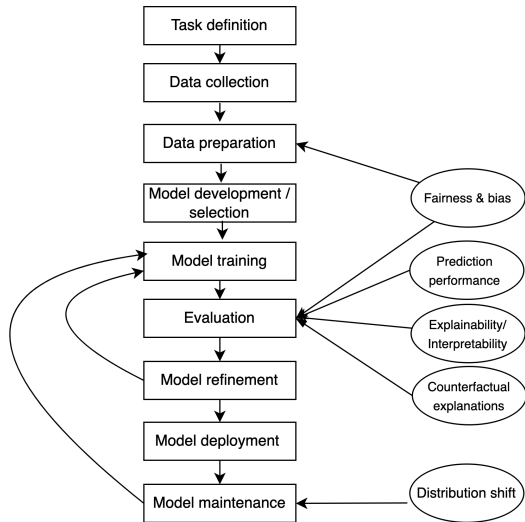
The application of Deep Neural Networks (DNNs) to a broad variety of tasks demands methods for coping with the complex and opaque nature of these architectures. When a gold standard is available, performance assessment treats the DNN as a black box and computes standard metrics based on the comparison of the predictions with the ground truth. A deeper understanding of performances requires going beyond such evaluation metrics to diagnose the model behavior and the prediction errors. This goal can be pursued in two complementary ways. On one side, model interpretation techniques “open the box” and assess the relationship between the input, the inner layers and the output, so as to identify the architecture modules most likely to cause the performance loss. On the other hand, black-box error diagnosis techniques study the correlation between the model response and some properties of the input not used for training, so as to identify the features of the inputs that make the model fail. Both approaches give hints on how to improve the architecture and/or the training process. This paper focuses on the application of DNNs to Computer Vision (CV) tasks and presents a survey of the tools that support the black-box performance diagnosis paradigm. It illustrates the features and gaps of the current proposals, discusses the relevant research directions and provides a brief overview of the diagnosis tools in sectors other than CV.

**Keywords:** black-box, error diagnosis, machine learning, evaluation, metrics

## 1 Introduction

The advances in Machine Learning (ML) led to the development of more complex neural networks nowadays known as Deep Neural Networks (DNNs), which are powerful models able to process complex data of various types [1]. DNNs have achieved outstanding results in many diverse domains and have become the solution of choice for

addressing big data analysis [2]. One of the domains in which DNNs have attained the most impressive results is Computer Vision (CV) where they have outperformed previous methods in a variety of tasks, including image and scene classification, object detection, semantic and instance segmentation, object and activity tracking and pose estimation [3].



**Figure 1** Life cycle of a DNN application

The life cycle of a DNN application differs from the development workflow of a traditional software system, because it relies on predefined algorithms that must be trained for the specific task and data at hand [4].

Figure 1 illustrates the typical development workflow of a DNN-powered application.

The peculiarity of such a workflow is the selection of the predefined algorithm more suited to the task and its fitting to the problem data via training, which introduces a distinction between the preparatory stage and the execution (inference) stage. A further specificity of DNNs is their complex and opaque nature, which makes debugging particularly hard.

The ordinary way of assessing DNNs is the evaluation of their prediction performance with a gold standard. Prediction performance analysis treats the DNN as a black-box and compares its output with the ground truth with metrics such as accuracy, precision, recall, etc. As DNNs get more and more applied to critical tasks, the need arises for a deeper understanding of the way in which predictions are made. This insight can be exploited to justify the output and/or to improve the performances.

The investigation of the behavior of ML models in general, and of DNNs in particular, can be pursued with analysis approaches that focus on distinct aspects:

- Prediction performance: is the assessment of the quality of the predictions. Performance can

be evaluated either qualitatively via manual inspection or quantitatively by comparison with ground truth test data. Quantitative performance analysis exploits standard metrics, such as Accuracy, Precision, Recall, F1-Score, or Average Precision.

- Model interpretability and explainability: is the ability to explain or to present in understandable terms to a human how a model makes a prediction [5, 6]. Intrinsic interpretability refers to those models that are simple and thus interpretable by design, such as short decision trees or sparse linear models. Conversely, post-hoc interpretability requires specific investigation techniques applied to the trained model. In DNNs, post hoc interpretability aims at exposing the relation of the internal representations of deep models to the input and output [7–11]. Techniques such as the Class Activation Maps (CAMs) [12–16] highlight the most influential regions of the feature maps at different network levels and enable the insight into the model prediction process. The difference between interpretability and explainability is subtle: the former emphasizes human intuition whereas the latter stresses the comprehension of the internal logic of the model [17].
- Counterfactual explanations: is the approach that exposes the model behavior by showing how some actions, such as a change to the input, alter the behavior of the system [18, 19].
- Fairness and bias: is the assessment of how an algorithm delivers predictions when applied to inputs belonging to data populations with different characteristics (e.g., to images of people belonging to different demographic groups) [20].
- Distribution shift: is the evaluation of how the model performances evolve when the distribution of data changes with respect to the one of the training and testing data [21].

The support of computer tools to the development life cycle of DNN applications has concentrated mostly on the basic tasks of model development/selection, training and testing. Several software packages provide off-the-shelf functionalities for defining the structure of a DNN, executing the training process and evaluating the most common metrics on the test set [22]. As the application of DNNs matures and becomes a common practice

in the industry, tool support beyond model definition, training and testing needs to be developed. This paper surveys the status of a specific sector of the research in this direction: black-box error diagnosis tools for CV tasks.

## 1.1 Focus of the survey

The survey has a threefold focus: on the application task, on the analysis type, and on the life cycle phase.

The **task focus** concentrates on the use of DNNs for CV. The motivation for this choice is the fact that CV is the sector in which the availability of black-box diagnosis tools is more significant. To complete the overview, we also provide in Section 4.7 a brief appraisal of the status of DNN diagnosis tools in other domains, such as time series analysis, natural language processing and recommender systems.

The **analysis focus** concentrates on prediction performance as the target of black-box diagnosis. Among the analysis approaches mentioned above, explainability and interpretability have already been described in several survey papers [23–26]. Fairness and bias detection is also well documented in several overview works [20, 27], as well as distribution shift [21]. Counterfactual explanations have been long used in statistical learning and have been recently rediscovered as a DNN explainability technique: the works [18, 19] document the status of research in that area.

The **life cycle** focus considers the tools that offer computerized support to the model evaluation and refinement steps.

When appropriate, if a tool supporting primarily the black-box diagnosis approach offers also functionalities for other types of analysis and/or development steps, these will be mentioned too.

## 1.2 Methodology

The target of the research comprises those methods that exploit only knowledge about the input and output to compute and break down performance metrics and to characterize errors. Among such works, we highlight the proposals that provide a tool for DNN evaluation, possibly together with functions for model design and training. This perimeter excludes those contributions that address DNN behavior and performances but pursue different targets such as special-purpose and

domain-dependent metrics, the visualization of DNN internal representations [25], model design for interpretability [5], and human-in-the-loop interpretation [28].

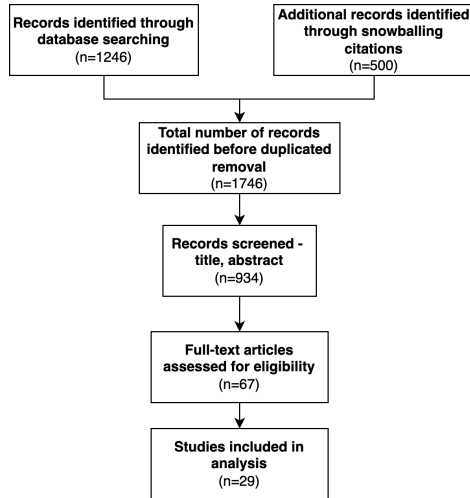
The corpus of the relevant research has been identified by following a simplified PRISMA procedure [29] for systematic reviews. Figure 2 illustrates the adopted workflow.

1. The search has been conducted on the Scopus repository, since previous studies have shown that it supports bibliographic research better than other sources [30]. The search phrases have been composed as follows:

```
<search> :- <task> AND <goal> AND <system>
<task> :- machine learning | deep learning |
computer vision | classification |
image classification |
scene classification |
object detection |
instance segmentation |
semantic segmentation |
object tracking |
pose estimation |
activity detection |
action detection
<goal> :- model diagnosis |
error diagnosis |
performance analysis
<system> :- tool | framework | workbench
```

The output of the search was filtered to retain only contributions in journals, conferences and workshops.

2. The initial corpus has been expanded through snowballing, recursively adding further related studies citing or cited by the works in the initial corpus.
3. The expanded corpus, composed of 1,746 works, has been reduced by removing duplicates. Next, we have identified and eliminated the studies unrelated to black-box techniques, by checking the title, keywords and abstract of each contribution. The reduced corpus contained 67 contributions.
4. A final eligibility filter has been applied by reading the full-text of the articles in the reduced corpus. This final step yielded the 29 works considered in this survey.



**Figure 2** PRISMA flow diagram of the systematic review

### 1.3 Contributions

The contributions of this paper can be summarized as follows:

- 29 black-box DNN performance diagnosis tools are identified from a initial corpus of 1,746 papers resulting from keyword and cross-references search.
- The tools are described and compared based on different dimensions (year, task, data type, metrics, performance and error break down functions, openness and extensibility)
- A list of open issues and relevant research directions are identified and discussed.
- A brief overview of the status of the DNN diagnosis tools in sectors other than CV is provided.

The rest of the paper is organized as follows: Section 2 describes the dimensions used to categorize the surveyed tools, Section 3 and 4 describe and compare the different tools based on the identified dimensions; Section 5 highlights the open issues and discusses the relevant research directions; finally, Section 6 draws the conclusions.

## 2 Classification of the tools

The relevant proposals are described and compared along six dimensions: task, media types, metrics, break down functions, custom properties, and openness and extensibility.

**Task:** An error diagnosis tool is typically designed for performance analysis of a specific task, which in turn may apply to a specific media type or to a range of media types. The surveyed diagnosis tools span the following tasks:

- **Classification (CL):** assignment of class labels to input samples. The samples used in CV applications belong to visual media; however, a tool may apply the same classification performance diagnosis functions to other data types, such as text, aural content, and record data. Therefore we consider classification in general and not only image classification.
- **Object Detection (OD):** localization of objects of a certain class through bounding boxes in images and videos.
- **Semantic Segmentation (SS):** assignment of a class label to each pixel in images or videos.
- **Instance Segmentation (IS):** similar to semantic segmentation but multiple objects of the same class are treated as separate instances.
- **Object Tracking (OT):** similar to object detection but each unique object is tracked as it moves across the frames of a video.
- **Pose Estimation (PE):** recognition of single or multiple body poses through key points in image and video.
- **Action Detection (AD):** assignment of an action label to a video.
- **Video Relation Detection (VRD):** spatio-temporal localization of object and subject pairs in videos and assignment of a label that describes their interaction.

**Media types:** Depending on the task, several media types can be relevant. The media types processed by the surveyed tools comprise image and video. The “generic” media type is used to refer to values of arbitrary record type.

**Metrics:** The quantitative analysis relies on metrics that may vary based on the targeted task. Near 40 metrics are mentioned in the surveyed tools. The definition of the non standard metrics can be found in the publications cited in the comparison tables.

**Break down functions:** In addition to the computation of the performance metrics, some tools implement functions that improve the characterization of the input and of the predictions, by breaking down input data sets, metrics and errors based on several criteria.

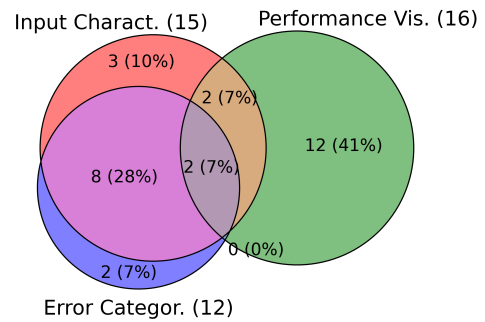
- Overall / Per-class / Per-property performance analysis: the tool supports the computation of the metrics for the entire data set and/or for individual classes or properties of the input.
- Overall / Per-class / Per-property reporting: the tool supports the construction of summary reports for the different levels of granularity used to compute the performance metrics.
- Categorization of errors: the tool supports the attribution of errors to specific categories, e.g., confusion with similar/dissimilar classes, poor localization, occlusion, etc.
- Error contribution isolation: the tool supports error impact analysis by highlighting the effect of all errors of a certain type on the performance metrics.
- Properties / Class distribution: the tool visualizes the distribution of properties and classes of the input data set.

**Custom property editing:** Some tools integrate a framework for adding custom properties to the input samples, with the following features:

- Manual annotation creation: a graphical interface allows the user to add annotations to the input samples.
- Annotation purpose selection: annotation can be distinguished into those for training (e.g., class labels) and those for diagnosis (e.g., domain dependent properties).
- Automatic annotation extraction: the tool enables the execution of algorithms for extracting meta-data and associating them to the input samples as custom properties.
- Selective visualization: the user can display the input data set and its annotations with multiple criteria (e.g., all samples of a certain class or with a specific property).

**Openness and extensibility:** Openness and extensibility are fundamental properties to support adoption especially when novel metrics or diagnosis approaches are proposed. The surveyed tools have been assessed based on their open source status and on the effort required for their extension. This qualitative dimension is characterized by means of the following values:

- Open source: the code is public and freely available.



**Figure 3** Venn diagram of the distribution of the surveyed tools in three macro-categories: Input characterization oriented, Error categorization oriented and Performance visualization oriented. Each circle represents the tools in one specific macro-category, while the intersections between circles show tools assigned to multiple categories (e.g., the 8 tools in the violet area pertain to both the Input characterization and Error categorization categories)

- User-defined metrics: the tool can be extended with custom metrics without modifying the framework.
- Data set independence: the tool can be applied to multiple data sets.
- User-defined properties: the tool enables the plug-in of modules that implement custom analysis types not present in the original proposal.

### 3 Tool characterization and description

Table 1 lists the 29 identified tools with the name of the tool or of its authors, the publication year, the targeted tasks, the media types, the ability to work with different data sets, and the link to the source code (only for open-source tools)<sup>1</sup>.

The tools can be grouped into three categories, shown in Figure 3, based on their prevalent approach to prediction performance diagnosis.

- *Input characterization oriented:* tools in this class stress the annotation of the input with custom properties to distinguish which aspects of the input affect the output and cause model failure and performance loss. Figure 4 shows an example of the interface for annotating images with custom properties and Figure 5 illustrates

<sup>1</sup>The link to the code repository is navigable in the online version of the paper.

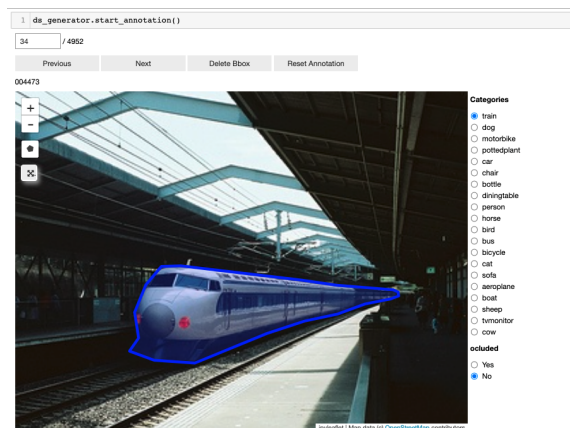
**Table 1** The surveyed tools listed by ascending year of publication. In the *Code* column, “-” indicates that the code is not available and “link” contains a reference to the code repository

Reference	Year	Task	Media	Data set independence	Code
Dollar et al. [31]	2009	OD	image	no	-
Hoiem et al. [32]	2012	OD	image	yes	link
Russakovsky et al. [33]	2013	OD	image	no	-
COCO API [34]	2014	OD, IS, PE	image	yes	link
Hariharan et al. [35]	2014	IS	image	no	link
Zhu et al. [36]	2015	OD	image	no	-
ModelTracker [37]	2015	CL	generic	yes	-
Redondo et al. [38]	2016	PE, OD+PE	image	no	link
Prospector [39]	2016	CL	generic	yes	-
Zhang et al. [40]	2016	OD	image	no	-
Ronchi et al. [41]	2017	PE	image	yes	link
Explanation Explorer [42]	2017	CL	generic	yes	link
Squares [43]	2017	CL	generic	yes	-
Sigurdsson et al. [44]	2017	AD	video	no	link
DETAD [45]	2018	AD	video	yes	link
Nekrasov et al. [46]	2018	SS	image	no	-
Manifold [47]	2018	CL	generic	yes	link
What If Tool [48]	2019	CL, BD	generic	yes	link
TIDE [49]	2020	OD, IS	image	yes	link
ODIN [50, 51]	2020	OD, IS, CL	image/generic	yes	link
Padilla et al. [52]	2020	OD	image	yes	link
TF-GraF [53]	2020	OD	image	yes	link
Boxer [54]	2020	CL	generic	yes	-
OpenVINO [55]	2020	CL, OD, SS, IS	images	yes	-
Padilla et al. [56]	2021	OD, OT	image	yes	link
TracKlinic [57]	2021	OT	video	yes	-
Chen et al. [58]	2021	VRD	video	yes	link
AIDeveloper [59]	2021	CL	image	yes	link
DETOXER [60]	2022	CL	video	yes	link

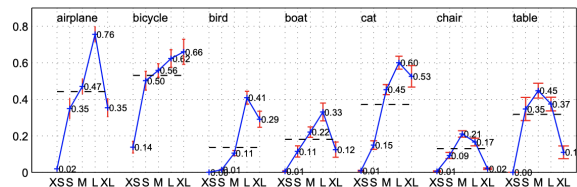
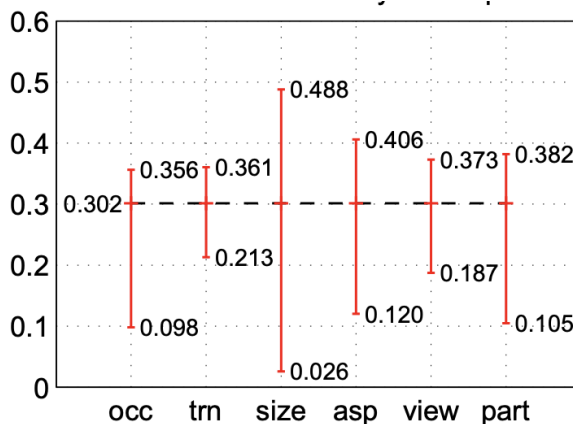
an example of the use of custom properties to break down the Average Precision (AP) metrics.

- *Error categorization oriented*: tools in this class stress the distinction among different types of errors to quantify the contribution of an error type to the performance loss. Figure 6 illustrates an example of error characterization diagram.
- *Performance visualization oriented*: tools in this class resort to advanced visualization and interaction to support the human judgement of

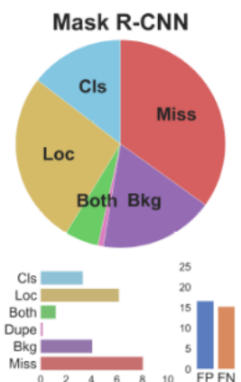
the performance problems. Figure 7 shows an example of the functionality of a performance visualization tool.



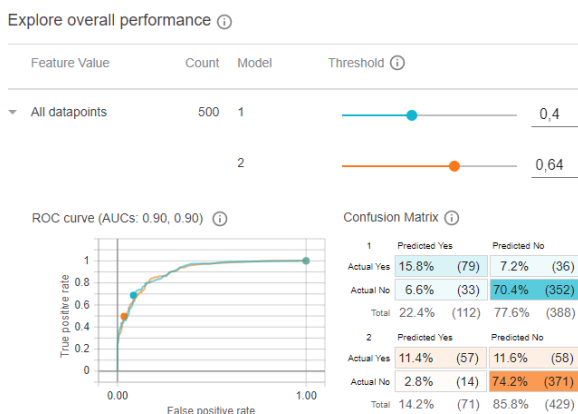
**Figure 4** ODIN tool [51]: the GUI lets the user annotate the input samples denoting either the class or a custom property not used for training



**Figure 5** [32] Use of custom properties (occlusion, truncation, bbox size, bbox aspect ratio, view point, part visibility) of the images for performance break down. Property sensitivity and impact analysis diagram (top): it shows the variation of AP for the different properties. AP break down diagram (bottom): it breaks down the AP metrics by bbox size value

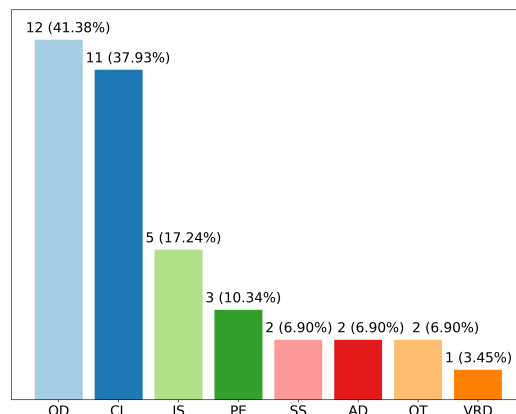


**Figure 6** Error categorization in the TIDE tool [49]: errors are distinguished in the types: categorization, localization, categorization+localization, duplicate detection, background and missed ground truth

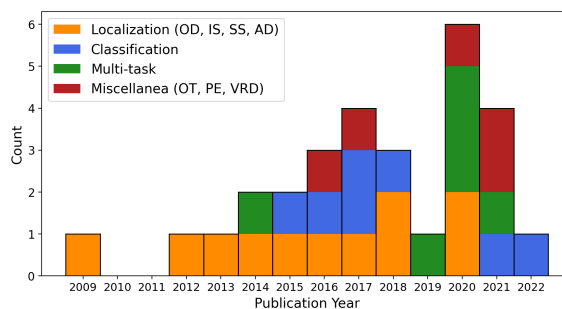


**Figure 7** Interactive performance visualization in the What If Tool [48]: the user can interact with the slider to modify the classification threshold and view the corresponding point on the curves, the values of the confusion matrix and the errors

Figure 8 shows the distribution of the tools across the CV tasks. Objection Detection and Classification are the most represented ones. The relevance of OD is not surprising because the pioneering works on black-box diagnosis [31] and [32] originated from the performance analysis of that task. Most tools work with images or with generic inputs, provide open-source code and have been released in the last five years. Figure 9 illustrates the distribution over time of the surveyed works. If one does not consider the early 2009 work [31], the timeline shows that the interest began in 2012, the same year in which the research on Deep Learning started its escalation. The idea originated in the CV field for such tasks as object detection and



**Figure 8** Distribution of the surveyed tools per task



**Figure 9** Distribution of the surveyed tools per year and per type of task

image segmentation and then propagated to other ML applications.

### 3.1 Descriptions of the tools

In the rest of this section we provide a brief description of the surveyed tools in ascending chronological order.

**Dollar et al.:** The work [31] presented a data set for pedestrian detection consisting of an annotated video with challenging low resolution images and occluded people. The data set was used to evaluate several detectors and the authors advocated the use of ad-hoc features of the input to help diagnose errors and support model refinement. To this end, performances were broken down based on specific properties of the input, such as the scale, the aspect ratio of the ground truth bounding boxes and the presence of occlusions of the pedestrians. Although the code was

not released, this early work is the first example of a black-box diagnosis tool and inspired the posterior proposals.

**Hoiem et al.:** The work in [32] pioneered the systematic black-box approach to error analysis in OD tasks and showed the utility of adding extra annotations to the input besides the labels used for training. The framework exploits a fixed set of diagnosis-oriented object meta-data that can affect the model accuracy, such as: size, parts visibility, aspect ratio, shape and occlusion (Figure 5). The authors demonstrate how breaking down standard metrics into sub-metrics linked to a metadata value aids in understanding model faults and in focusing redesign where the margin for improvement is maximal.

**Russakovsky et al.:** The work in [33] follows the black-box diagnosis method of [32] and assesses the performance of several object detectors applied to the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) data set. A first type of analysis breaks down the performance indicators using *per-image* properties, such as the number of instances per image and the chance performance of localization (CPL). A second type of analysis shows how the performances are affected by *per-class* properties, such as the distinctiveness of color and of shape, the instance deformability and the amount of texture.

**COCO API:** A step towards the popularization of custom input properties as an aid for error diagnosis is found in the development framework of the MS COCO data set [34]. The computation of mean Average Precision (mAP) is differentiated based on the object size:  $\text{mAP}_{\text{small}}$ ,  $\text{mAP}_{\text{medium}}$  and  $\text{mAP}_{\text{big}}$ . This distinction assists the diagnosis of the issues and enables the design of strategies to improve the localization, such as multi-scale object detection [61]. In addition to the the break down of the metrics, the API also allows developers to load any data set that respects the MS COCO format and to visualize both the images and the annotations.

**Hariharan et al:** The work in [35] introduces simultaneous detection and segmentation (SDS) as a novel computer vision task. The authors provide the DNN architecture and a tool for its diagnosis. Besides the assessment of standard metrics, error diagnosis is supported by introducing three error classes (localization, confusion with

similar classes, and confusion with background) and by computing the impact of each error type on the performance indicators.

**Zhu et.al.:** The authors of [36] follow the methodology of [32] and evaluate object detectors using custom properties. They contrast different methods for creating object proposals on the PASCAL VOC data set. The comparison uses object characteristics such as size, aspect ratio, iconic view, color contrast, shape regularity and texture. The authors also discuss how to exploit objects properties to investigate model limitations and show the sensitivity of the model to the characteristics of the objects.

**ModelTracker:** The work in [37] investigates the performances of a classifier with a black-box approach that combines metrics summaries and interactive visualizations. Binary predictions are color-coded and arranged by classification score. The analysis of results is facilitated by tagging input samples with custom properties and by highlighting samples similarity and outliers.

**Redondo et al.:** The authors of [38] propose a diagnostic tool tailored to the study of pose estimation errors. The tool examines the effects of custom properties (aspect ratio, size, visibility of parts) on the detection and pose estimation performances and highlights the impact of different types of pose-related False Positives. The authors analyze four state-of-the-art object detection and posture estimation models to uncover flaws and recommend improvements.

**Prospector:** The work in [39] describes a web-based tool that implements counterfactual analysis. The tool implements a partial dependence technique for determining the impact of each input feature on the DNN results. The developer can apply changes to the input data and measure the impact on the output. The system suggests the shift in the value of each input feature that would lead to the greatest performance improvement. The diagnostic utility of the approach is demonstrated in a diabetes prediction task.

**Zhang et al.:** In [40] the authors apply error diagnosis to the state-of-the-art pedestrian detection algorithms. They enhance the annotations of the Caltech [31] data set and study both False Positives (FPs) and False Negatives (FNs) with different error categories. FP errors are distinguished into localization, background, and annotation. FN errors are categorized into scale,



viewpoint, occlusion, and other types. The authors also analyze the impact of FPs on performances.

**Ronchi et al.:** The work in [41] applies the approach of [32] to the Multi-Instance Pose Estimation task. Three error types are defined (localization, scoring and background) and the impact of three challenging factors is studied (occlusion, crowding and size). Their tool visualizes the distribution of errors for each key point and highlights the improvement in the Precision-Recall curve obtainable by correcting specific types of errors.

**Explanation Explorer:** In [42], the authors of Prospector [39] describe a novel tool for the assessment and interpretability of binary classifiers. Their approach comprises three steps: 1) the display of classical performance metrics and analyses; 2) the explanation generation, which computes the features of the input samples that impact the outcome most significantly; 3) the interactive visualization of the explanations. The visualization is organized in three stages: 1) outcome-level, focusing on the overall accuracy; 2) feature-level, presenting the explanations along with the corresponding features; and 3) instance-level, which allows the user to analyze each instance and derive hypotheses about the classifier failures. The authors advocate that visual analytics should play a major role in error diagnosis but also highlight that not all failures can be rectified by training a stronger model because some errors require bias mitigation in the original data set.

**Squares:** Squares [43] is a tool for the interactive performance analysis of multi-class single-label classifiers. The classes and the corresponding instances are displayed on the same row with a distinctive color. The observations are ordered by their prediction confidence score and grouped. The first group represents the FNs whereas the second cluster comprises both the True Positives (TPs) (highlighted with the color of their class) and the FPs (highlighted with the color of their true class). When an observation is selected from one class, its representations in the other classes are emphasized visually, thus allowing a comparison among the different predictions of the same sample.

**Sigurdsson et al.:** The work in [44] surveys the state-of-the-art in the action detection task and compares the existing methods. The evaluation is based on the categorization of errors in four classes: boundary, other class with the same object, other class with the same verb, other

class with neither and no class. A further analysis evaluates the models w.r.t. the complexity of the objects/verbs that characterize a category. Finally, the authors examine the impact of two specific features of the input: the temporal extent of the action and the presence of people.

**DETAD:** The focus of [45] is on the identification of temporal actions in videos. The diagnosis tool enables the analysis of FPs and FNs and the estimation of the sensitivity of mAP-based metrics to six action characteristics: length, context distance, agreement, coverage, context size and number of instances.

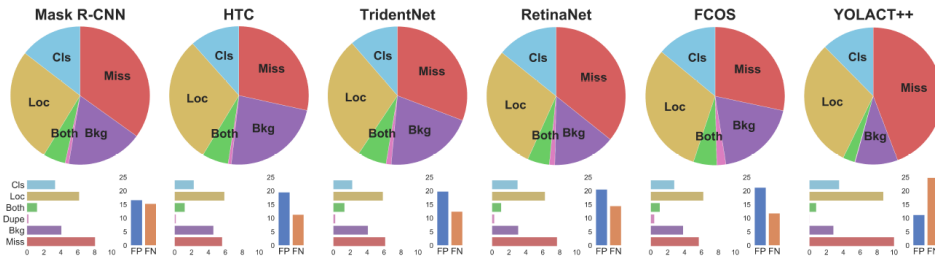
**Nekrasov et al.:** In [46] the authors apply the black-box diagnostic strategy of [32] to the semantic segmentation task.

**Manifold:** The authors of [47] discuss an interactive framework for the evaluation and debugging of ML models. An agreement analysis function enables the comparison of model pairs by highlighting the similarities and differences of their predictions. A feature distribution function permits the selection of a subset of the samples and measures the intra-group similarity based on the occurrence frequency of each feature.

**What If Tool:** As the name suggests, the tool described in [48] allows researchers to analyze the performances of ML systems in hypothetical situations by visualizing the effect of several features on different models and on different subsets of the input data. Among all the surveyed tools, this is the only one providing also a fairness analysis that highlights bias in the input data set. Other functionalities support data point editing, counterfactual reasoning, performance measures for classification and regression (Figure 7) and data set fairness optimization.

**TIDE:** The tool illustrated in [49] supports error diagnosis in object detection and instance segmentation. The tool is applicable to multiple data sets and output formats. Similarly to [32] it categorizes detection and segmentation errors and provides compact error summaries and impact reports. For example, it includes the direct comparison of results by different models (Figure 10). Unlike other tools such as [32, 50], TIDE purposely avoids resorting to properties of the input other than those used for training.

**ODIN:** The ODIN framework [50, 51] aims at generalizing and integrating into a unique solution the previous approaches to DNN black-box



**Figure 10** Error categorization and model comparison in TIDE [49]

diagnosis for classification, object detection and instance segmentation. ODIN allows the addition of custom properties to the input, supports the plug-in of user-defined performance indicators and implements a wide range of metrics and analysis reports off-the-shelf (Figure 4). It can be used to study both model performance and data set bias. It combines error impact sensitivity and confidence calibration. The latter analysis evaluates the similarity of the distribution of predictions to the real probability distribution of the input data. The tool also comprises a Graphical User Interface (GUI) for annotating the input with custom properties.

**Padilla et al.:** The authors of [52] discuss the most commonly used metrics for object detection and provide the implementation code in a toolkit. The work in [56] describes the most recent release, which makes the toolkit independent of the input formats, adds more bounding box formats, and includes novel spatio-temporal metrics for object detection in video.

**TF-GraF:** The goal of the work in [53] is to create an easy-to-use Tensorflow object detection environment, simplifying the installation and set-up and the coding and execution of the workflows. The tool supports pre-processing, training and evaluation with the MS COCO metrics. It incorporates the best known object detection and instance segmentation architectures (Faster RCNN, SSD, Mask RCNN) and provides the visualization of the training and test data sets. Non-experts can configure, train, and assess DNNs with no programming.

**Boxer:** The work in [54] presents a tool called Boxer for comparing the performances of different classifiers. The system supports the selection of metrics, the grouping of training and test data into “boxes” based on selected features and the comparative visualization of the outputs. To evaluate

data quality and bias, a novel method based on set algebra is presented to link views and analyze multiple data subsets. The authors demonstrate the tool utility in a variety of use cases and discuss strategies for dealing with very large data sets.

**OpenVINO DL Workbench:** The work in [55] focuses on model training, analysis, optimization, evaluation and deployment, covering the entire model development workflow with a hybrid black-box and white-box approach. The workbench includes an Accuracy Checker that implements a black-box analysis for classification, regression, and object detection tasks by computing the most common metrics for the whole data set and per class. It also supports the open-box analysis, for example to evaluate and improve model performance in terms of execution time and memory consumption.

**TracKlinic:** The work in [57] studies the factors that challenge object tracking in videos. Custom properties can be manually associated with the video frames to specify seven common error-inducing factors: occlusion, rotation, out-of-view, background clutter, illumination variation, shape variation, and motion blur. The proposed diagnosis tool exploits the Intersection over Union (IoU) base metrics to analyze the failure rates and the success scores of ten state-of-the-art architectures applied to three benchmark data sets manually annotated with the above mentioned factors. Per-frame proposals of alternative models can be visualized together and compared with the ground truth annotations. The diagnosis results show that most models fail when complex situations occur, such as out-of-view transitions and shape variations.

**Chen et al.:** The work in [58] focuses on the relation detection task in videos and assesses the state-of-the-art detectors over two benchmarks (ImageNet-VidVRD [62] and VidOR [63]). The

authors categorize the FPs and compute their impact over the Average Precision, analyze the FN distribution across different input characteristics (e.g., the video length, the number of subject/predicate/object instances, the subject/object pixel scale) and compute the performance gain achievable by removing each error type.

**AIDeveloper:** The work in [59] presents an open-source software supporting the entire development process of an image classification model: dataset definition and visualization, model training and optimization and performance evaluation. The tool includes an easy-to-use GUI that allows researchers to develop DL-powered applications without coding.

**DETOXER:** The work in [60] is an interactive visual analytics tool for debugging Temporal Multi-Label Classification models in multiple videos. The authors designed it in order to provide three different levels of granularity for explanations and evaluation: frame-level analysis offering a compact visualization of all the categories; video-level explanation providing an overview of the errors (false positives and false negatives) for each video; and, global-level summary revealing error trends across all the analyzed videos.

## 4 Comparison of diagnostic tools

For each tool described in Section 3 the supported metrics and types of analysis were extracted, resulting in more than 70 options. For ease of comparison four tables are introduced, one for each family of homogeneous metrics/analyses: generic multi-task (Table 2), classification (Table 3), localization (Table 4), and a miscellaneous category grouping the functions found in the less frequent tools for object tracking and pose estimation (Table 5). The rows specify the metrics/analysis and the columns the tools that support them, sorted in chronological order. For space reasons, the tool or authors' names are omitted but they can be recovered from the corresponding reference. Each cell specifies if the option is offered by the specified tool. The symbol “-” means that the row is not relevant for the specific tool. For example, a metrics specific for pose estimation is not relevant for tools focused on object tracking. Note that when some base metrics is used to compute a

derived metrics (e.g., IoU and AP in Table 4) the row of the base metrics contains the  $\checkmark$  value only when the tool exposes the base metrics explicitly.

### 4.1 Multi-task metrics and analyses

Table 2 lists 16 general metrics that apply to all the considered tasks and the 23 tools that implement them. Only Accuracy is well represented in the surveyed classification tools. Other standard ML metrics (e.g., precision, recall, F1-score and AUCs) are present only in 4 or 5 tools, less than one may expect. A similar consideration also holds for the Precision-Recall, F1 and ROC curves, usually employed to visualize and compare performances and optimize the hyperparameters. Only 7 out of 23 tools offer such features. The most represented types of analysis are those related to FPs and FNs, which are the most common starting points for error diagnosis.

### 4.2 Classification metrics and analyses

Table 3 lists 8 classification-specific metrics implemented in 8 frameworks. The Confusion Matrix is implemented by 7 out of 8 tools, whereas Error Rate, Mean Error (ME), Mean Absolute Error (MAE), Mean Squared Error (MSE), Odds Ratio and the True Negative (TN) analysis are implemented by only one proposal.

### 4.3 Localization metrics and analyses

Table 4 presents the 6 localization-specific metrics and the 19 tools that implement them. The surveyed frameworks support a variety of localization tasks: OD, IS, SS, AD and PE. The review shows that there is little consensus among localization-oriented frameworks about which metrics are essential and should be provided off-the-shelf. The implementation by tools concentrates only on the metrics commonly required by the most popular CV benchmarks: Average Precision (IoU) for OD and IS; other useful metrics, both general and localization-specific, such as Miss Rate, Average Recall (IoU) or F1 Score are implemented rather infrequently by the object localization tools.

**Table 2** Multi-task metrics and analyses.

Metric / Analysis	[32]	[33]	[34]	[35]	[36]	[38]	[39]	[40]	[41]	[42]	[44]	[45]	[46]	[48]	[49]	[50]	[52]	[54]	[55]	[57]	[58]	[59]	[60]
Accuracy	-	✓	-	-	-	-	✓	-	-	✓	-	-	✓	✓	-	✓	-	✓	✓	-	-	✓	✓
Precision																							
Recall					✓																		
ROC Curve	-	-	-	-	-	-	-	-	✓	-	-		✓	-	✓	-					-	-	✓
ROC AUC	-	-	-	-	-	-	-	-	✓	-	-		✓	-	✓	-					-	-	✓
PR curve				✓					✓					✓	✓						-	-	✓
PR AUC							✓							✓	✓						-	-	✓
F1 Score														✓	✓						-	-	✓
F1 Curve																✓							
F1 AUC																✓							
# FP														✓	✓								✓
# FN														✓	✓								✓
FN Analysis			✓					✓	✓						✓	✓							✓
FP Analysis	✓		✓	✓		✓		✓	✓			✓	✓		✓	✓							✓
TP Analysis																✓							✓
Reliability Analysis																✓					✓		

**Table 3** Classification-specific metrics and analyses.

Metric / Analysis	[42]	[43]	[47]	[48]	[50]	[54]	[55]	[59]
Error Rate					✓			
Confusion Matrix		✓	✓	✓	✓	✓		✓
Mean Error				✓				
Mean Absolute Error				✓				
Mean Squared Error				✓				
Odds Ratio	✓							
Matthews Correlation Coefficient							✓	✓
TN Analysis							✓	

## 4.4 Object Tracking and Pose Estimation metrics and analyses

Table 5 lists task-specific metrics found in tools designed for OT and PE. Also in this case a lack of consensus about a common set of relevant metrics can be observed. For example both [41] and [38] focus on PE but do not share any of the task-specific metrics.

## 4.5 Error categorization metrics and analyses

Table 6 lists 12 tools that implement error categorization and define 14 types of errors. Error categorization is provided only by OD, IS or PE tools and is not yet common for other tasks. As discussed in the surveyed works, error categorization unveils specific factors associated with model failure that are difficult to extract from aggregated

metrics alone. The error categorization tools exploit this feature for impact analysis and highlight the performance gain obtainable by removing or mitigating a specific type of error.

## 4.6 Additional features

Table 7 presents 14 features not related to any specific metrics. Some functions let the user inspect the predictions at different levels of granularity and display the distribution of classes and properties in the input data set. Other functions refer to the availability and extensibility of custom properties and metrics. A last group contains functions for model comparison and data set annotation and visualization. The multi-level analysis and the comparison of models are present in most tools. Few frameworks accept custom evaluation criteria. Despite the fact that many frameworks include

**Table 4** Localization-specific metrics and analyses.

Metric / Analysis	[31]	[32]	[34]	[35]	[36]	[38]	[40]	[41]	[44]	[45]	[46]	[49]	[50]	[52]	[53]	[55]	[56]	[57]	[58]
Mean IoU				✓		-	-				✓					✓			
Average Precision (IoU)		✓	✓		✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Average Recall (IoU)			✓												✓	✓	✓	✓	✓
Miss Rate		✓						✓	-	-						✓			
Localization Latency				✓									✓	✓					
IoU Analysis				✓	✓	✓							✓	✓					✓

**Table 5** Object Tracking and Pose Estimation metrics and analyses.

Metric / Analysis	[38]	[41]	[56]	[57]
Success Score (OT)	-	-		✓
Failure Rate (OT)	-	-		✓
Consistency Analysis (OT)	-	-		✓
Spatio-Temporal Tube Average Precision (STT-AP) (OT)	-	-	✓	
Pose Estimation Average Precision (PE)	✓		-	-
Average Viewpoint Precision(PE)	✓		-	-
Average Orientation Similarity(PE)	✓		-	-
Mean Angle Error (PE)	✓		-	-
Median Angle Error (PE)	✓		-	-
Object Keypoint Similarity (PE)		✓	-	-

**Table 6** Error categorization metrics and analysis.

Metric / Analysis	[32]	[35]	[38]	[40]	[41]	[44]	[45]	[46]	[49]	[50]	[57]	[58]
Errors Categorization (ET)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ET: Classification	✓								✓			✓
ET: Localization	✓	✓		✓	✓		✓	✓	✓	✓		✓
ET: Classification + Localization									✓			✓
ET: Duplicated	✓						✓		✓			✓
ET: Missed Ground Truth		✓				✓			✓			✓
ET: Confusion with background	✓	✓	-	✓	✓		✓	✓	✓	✓	✓	✓
ET: Confusion with similar class	✓	✓	-					✓		✓		
ET: Confusion withn similar class	✓		-					✓		✓		
ET: Other	✓	✓	✓	✓	✓	✓				✓	✓	✓
ET: Opposite (PE)	-		✓	-	-	-	-	-	-	-	-	-
ET: Nearby (PE)	-		✓	-	-	-	-	-	-	-	-	-
ET: Confusion with other class with same object	-	-	-	-	-	✓	-	-	-	-	-	-
ET: Confusion with other class with same verb	-	-	-	-	-	✓	-	-	-	-	-	-
ET: Boundary	-	-	-	-	-	✓	-	-	-	-	-	-
Error Contribution Analysis	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓

analysis processes that depend on custom properties, only 2 tools include an annotator GUI. Only 5 tools offer an interface for visualizing and filtering the Ground Truth (GT) data and the model predictions (e.g., for filtering predictions by a specific error type). A final remark on the reporting capabilities: less than 30% of the surveyed tools offer a way to build a comprehensive report with all the relevant metrics. Even less tools provide a summary organized by class or by custom property.

## 4.7 Beyond Computer Vision

The focus of this survey is on the tools that assist the CV tasks. The overview has been extended also to the frameworks that support the classification of other types of data, because they share most metrics and types of analysis with the surveyed image classification tools. However, the black-box diagnosis approach is relevant also in other scenarios in which DNNs are applied to such tasks as time series analysis (TS), natural language processing (NLP), and recommender systems (RS). In this section we provide some essential references to the research and survey works

that address the black-box diagnosis for tasks other than the CV ones.

A notable example is the application of DNNs to temporal data series for such applications as anomaly detection [64, 65] and predictive maintenance [66]. The time series data sets for such applications are characterized by many properties (e.g., the sampling frequency, the stationarity and periodicity of the series, the type and physical characteristics of the signal and of the corresponding acquisition sensor). Such a richness of significant input properties could be exploited to enable the break down of performance indicators and the attribution of errors to specific features of the input. Recent works have started to implement off-the-shelf black-box error diagnosis and performance break down functionalities [67].

Tools such as [68–70] extend the support beyond the use of basic metrics in the evaluation phase and cover also the training and refinement of the model. The work [71] proposes novel temporal evaluation metrics and provides their implementation as command line scripts. Several research and commercial frameworks assist the workflow of anomaly detection and predictive maintenance applications but do not support error attribution and metrics break down. An example is

**Table 7** Performance break down, input distribution, custom properties and metrics, model comparison, data editing and visualization functions.

Feature	[31]	[32]	[33]	[34]	[35]	[36]	[37]	[38]	[39]	[40]	[41]	[42]	[43]	[44]	[45]	[46]	[47]	[48]	[49]	[50]	[52]	[53]	[54]	[55]	[56]	[57]	[58]	[59]	[60]
<b>Performance break down and input distribution functions</b>																													
Overall Analysis	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Per class Analysis	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Per property Analysis	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Overall report				✓	✓															✓	✓	✓	✓					✓	
Per class report					✓															✓	✓	✓	✓					✓	
Per property report				✓																✓	✓	✓	✓					✓	
Property distribution																					✓	✓	✓	✓		✓	✓	✓	
Class distribution																					✓	✓	✓	✓		✓	✓	✓	
<b>Custom properties and metrics</b>																													
Builtin input properties	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
User-defined properties																													
User-defined metrics																													
<b>Editing, visualization and comparison functions</b>																													
Model comparison	✓	✓																											
Annotator																					✓	✓	✓	✓					
Visualizer																					✓	✓	✓	✓					✓

RELOAD [68], which aids the ingestion of data, the selection of the most informative features, the execution of multiple anomaly detection algorithms, the evaluation of alternative anomaly identification strategies, the evaluation of multiple metrics and the visualization of results in a GUI. RELOAD implements multiple metrics and algorithms off-the-shelf and has an extensible architecture. However it does not support yet the break down of performance metrics and the attribution of errors based on the features of the input. A black-box error diagnosis tool for time series is ODIN TS [72], an extension of the ODIN tool [51] for time series analysis. ODIN TS implements the basic time series metrics and diagrams (accuracy, precision, recall, F1 score, miss alarm rate, false alarm rate, NAB score, MAE, MSE, Root Mean Squared Error, Mean Absolute Percentage Error, Precision-Recall and ROC curves), introduces new types of analysis for anomaly detection, such as FP error categorization, enables the annotation of the time series and the visualization of the data set and of the predictions.

In the NLP field, the BlackBox workshops series<sup>2</sup> is dedicated to interpretability and diagnosis issues in the application of DNNs to NLP problems, with a multidisciplinary approach spanning not only machine learning, but also psychology, linguistics, and neuroscience. As an example of this line of research, the work [73] describes the GEval tool, a framework for detecting anomalies in NLP test sets, supporting data preparation, detecting problems in the model and comparing the output of alternative models. LIT (Language Interpretability Tool) [74], by Google Research, is a framework focused on the interpretability of NLP

models, which also supports multiple evaluation metrics, counterfactual analysis and the visualization of the data set and of the model outputs. EXPATS (Explainable Automated Text Scoring) [75] builds upon LIT and offers life cycle support for the specific NLP task of text scoring. It implements multiple feature extractors and metrics and can work with predefined or user-supplied models. The tool is open source and designed with an open architecture, to enable the plugin of custom analysis components.

The proliferation of algorithms in the recommender system sector has spawned the interest for tools supporting the life cycle of system development, with the aim of standardizing the evaluation and easing the reproduction of results. The recent contributions include comprehensive libraries implementing the most popular RS algorithms, such as the RecBole library [76], and tools supporting the model development and assessment. A recent example of the available toolkits is ELLIOT [77] an open-source recommendation system development framework assisting pre-processing operations, enabling alternative hyperparameters optimization strategies and implementing multiple evaluation metrics. ELLIOT also covers bias and fairness analysis, supported by statistical significance tests, a functionality particularly relevant in recommendation scenarios.

## 5 Issues and research directions

Black-box error diagnosis is a viable complement to interpretability techniques for achieving a deeper understanding of the performances of DNNs. However, the panorama of the tools and

<sup>2</sup><https://blackboxnlp.github.io/>

frameworks that support error diagnosis shows that there are still margins for improvement and research before full maturity is attained.

## 5.1 Open issues

The analysis of the surveyed tools reveals several open issues.

- **Consensus:** Average Precision and Accuracy are implemented off-the-shelf by most CV and classification tools. But besides such basic metrics, there is little agreement among the tools addressing the same task about the core set of metrics and analyses that are most beneficial to performance and error diagnosis. Table 8 shows task by task the percentages of tools that implement each metrics. A common effort to define a core set of metrics per task must go beyond the few metrics popularized by the benchmarks focused on end-to-end evaluation. The definition of a consensus set of metrics and analyses would promote the design of methodological guidelines for black-box error diagnosis based on fundamental performance indicators and diagnosis reports available off-the-shelf. It would also help model and framework developers save time in the implementation of diagnosis metrics and reports and concentrate on more advanced functionalities of their architectures and tools. The starting point for the definition of a consensus set of indicators is the abundant literature on the performance metrics most suitable for each task [52, 56, 78–80].
- **Workflow coverage:** all the analyzed tools focus on prediction performance evaluation with few considering also the other life cycle phases of Figure 1. Thus, a complete development workflow would require the use of multiple tools, each one with its own input/output formats, configuration, metrics, and visualizations/reports. Integrating in one solution the support to all the life cycle phases would accelerate the model development, evaluation and refinement loop. This requirements is especially relevant to those tools that offer the break down of metrics based on custom properties, which should be added manually to the input data set, or extracted automatically from it, during the preparation phase.
- **Visual analytics and support for qualitative analysis:** the surveyed tools include almost only quantitative metrics or analyses that are useful to measure model performance or diagnose errors. In particular, when dealing with visual data, these measures are not enough to fully understand the model behavior. Qualitative analysis is fundamental to visualize both errors and correct predictions and, associated with interpretability techniques, may help discover unwanted correlations. The interfaces of error diagnosis tools could be extended to support the automatic visualization of data samples relevant for qualitative analysis, e.g., those falling into given ranges of one or more output metrics or those displaying a certain type of error.
- **Automatic extraction of properties:** custom properties associated to the input data have been shown to be beneficial in diagnosing and categorizing errors. Such properties in part can be automatically derived from the data (e.g., image color space, bounding box size and aspect ratio, difficulty and quality level, number of objects, etc.). Some of the analyzed tools compute only elementary properties (e.g., bounding box size or image color scheme) and none integrate an approach to extract non trivial diagnosis-oriented properties automatically.
- **Data quality assessment:** errors that are attributed to the model may be due to the presence of noise in the ground truth annotations. Only few tools integrate data quality and error diagnosis and provide head to head comparison of the same model on different data sets. A particular case of data quality analysis is bias detection in the input data set, which is the subject of a completely distinct family of tools [20] but is supported only by one black-box error diagnosis framework [48].
- **Scalability:** the authors of several tools openly admit that their framework cannot be applied to very large data sets, featuring many samples, many classes, or many custom properties. The difficulty stems from both the computational effort required (e.g., metrics may be computationally heavy) and from the visualization of results (traditional plots, interactive views, and summaries get cluttered and difficult to read).
- **Architecture and API standardization:** Albeit most tools publish their implementation code in open source repositories, the lack of a common architecture and of standard APIs

**Table 8** The tools that implement each metrics in percentage. Each column represents a task or a family of related tasks and shows the number of tools that support it.

Metrics-Analysis/Task	CL (11)	OD/IS/SS (14)	PE (3)	AD (2)	OT (2)	VRD (1)
Accuracy	<b>73% (8)</b>	14% (2)	0% (0)	-	-	-
Error Rate	9% (1)	-	-	-	-	-
Precision	36% (4)	14% (2)	0% (0)	0% (0)	0% (0)	0% (0)
Recall	36% (4)	21% (3)	0% (0)	0% (0)	0% (0)	0% (0)
F1 score	36% (4)	14% (2)	0% (0)	0% (0)	0% (0)	0% (0)
Average Precision	18% (2)	<b>64% (9)</b>	<b>100% (3)</b>	<b>100% (2)</b>	<b>50% (1)</b>	<b>100% (1)</b>
Average Recall	9% (1)	29% (4)	33% (1)	0% (0)	<b>50% (1)</b>	0% (0)
ROC AUC	45% (5)	-	-	-	-	-
Precision-Recall AUC	36% (4)	7% (1)	0% (0)	0% (0)	0% (0)	0% (0)
F1 AUC	9% (1)	7% (1)	0% (0)	0% (0)	0% (0)	0% (0)
Mean Error	9% (1)	-	-	-	-	-
Mean Absolute Error	9% (1)	-	-	-	-	-
Mean Squared Error	9% (1)	-	-	-	-	-
Odds Ratio	9% (1)	-	-	-	-	-
Matthews Correlation Coefficient	18% (2)	-	-	-	-	-
Mean Intersection Over Union (Mean IoU)	-	21% (3)	0% (0)	0% (0)	<b>100% (2)</b>	0% (0)
Miss Rate	-	21% (3)	-	-	0% (0)	0% (0)
Localization Latency	-	7% (1)	-	0% (0)	0% (0)	0% (0)
Success Score	-	-	-	-	<b>50% (1)</b>	-
Failure Rate	-	-	-	-	<b>50% (1)</b>	-
Spatio-Temporal Tube Average Precision (STT-AP)	-	-	-	-	<b>50% (1)</b>	-
Average Viewpoint Precision	-	-	33% (1)	-	-	-
Average Orientation Similarity	-	-	33% (1)	-	-	-
Mean Angle Error	-	-	33% (1)	-	-	-
Median Angle Error	-	-	33% (1)	-	-	-
Object Keypoint Similarity	-	-	33% (1)	-	-	-
ROC curve	36% (4)	-	-	-	-	-
Precision-Recall curve	36% (4)	14% (2)	33% (1)	0% (0)	0% (0)	0% (0)
F1 curve	10% (1)	7% (1)	0% (0)	0% (0)	0% (0)	0% (0)
Confusion Matrix	<b>73% (8)</b>	-	-	-	-	-
# False Positives (FP)	27% (3)	14% (2)	0% (0)	0% (0)	0% (0)	0% (0)
# False Negatives (FN)	27% (3)	14% (2)	0% (0)	0% (0)	0% (0)	0% (0)
True Positive Analysis	9% (1)	7% (1)	0% (0)	0% (0)	0% (0)	0% (0)
False Positive Analysis	9% (1)	50% (7)	<b>100% (3)</b>	<b>100% (2)</b>	50% (1)	<b>100% (1)</b>
False Negative Analysis	9% (1)	29% (4)	66% (2)	50% (1)	0% (0)	<b>100% (1)</b>
True Negative Analysis	9% (1)	-	-	-	-	-
Error Categorization	9% (1)	43% (6)	<b>100% (3)</b>	<b>100% (2)</b>	50% (1)	<b>100% (1)</b>
False Positive Error Categorization	9% (1)	43% (6)	100% (3)	100% (2)	50% (1)	<b>100% (1)</b>
False Negative Error Categorization	9% (1)	14% (2)	0% (0)	0% (0)	0% (0)	<b>100% (1)</b>
Error Contribution Analysis	9% (1)	50% (7)	66% (2)	50% (1)	0% (0)	<b>100% (1)</b>
Intersection Over Union Analysis	-	36% (5)	33% (1)	0% (0)	<b>50% (1)</b>	0% (0)
Reliability Analysis	9% (1)	7% (1)	0% (0)	0% (0)	<b>50% (1)</b>	0% (0)
Temporal Reasoning	-	-	-	<b>50% (1)</b>	0% (0)	-
Person-based Reasoning	-	-	-	<b>50% (1)</b>	0% (0)	-
Qualitative Analysis (Visualizer)	27% (3)	21% (3)	0% (0)	-	0% (0)	0% (0)



makes the implementation of metrics, property extractors and visualizations/reports non portable and requires the re-implementation or the wrapping of even the most basic performance indicators. The definition of a plug-in architecture and of standard module interfaces is typical in more mature fields such as software development, as witnessed e.g., by the popular Eclipse<sup>3</sup> and JetBrains<sup>4</sup> frameworks. A similar approach applied to black-box diagnosis tools and more generally to DNN workflow management tools would promote the development of a community-managed library of reusable metrics, property extractors and visualization-s/reports which could be installed rather than re-implemented in any given framework.

## 5.2 Research directions

The review of DNNs black-box diagnosis frameworks uncovers many research challenges that are still to be pursued.

- **Integration of black-box diagnosis and interpretability techniques:** interpretability in machine learning is defined as “the ability to explain or to present in understandable terms to a human” [5, 6] how a model makes a prediction. The interpretability of DNNs has attracted increasing attention by researchers [8] due to the impact that this class of algorithms has in critical domains such as medicine, economy, safety and social sciences. Interpretability techniques offer a view of the system behavior alternative but closely related to that afforded by black-box error diagnosis. They seek to unveil *interpretability factors* i.e., human-understandable concepts and processes that are at the base of the model prediction. Black-box error diagnosis and performance break down can aid the discovery of interpretability factors: if a model consistently fails or succeeds when the input exhibits certain human-defined properties, this is a hint that such properties play a role in the interpretability. For example, if a system for iconography classification in art images [81] consistently classifies representations of a given subject (e.g., images of Saint Jerome in Christian art paintings) with more accuracy and

confidence when the input contains distinctive symbols (e.g., a lion couched at the saint’s feet or the cardinal’s galero) this finding could attest the interpretative value of such symbols in the classification of that character. The relationship between interpretability and black-box analysis is exploited in methods, such as [82], that aim at discovering interpretability factors by modifying the input data and then measuring the changes in the output (e.g., by suppressing some features or masking part of an image), and [83, 84], which allow the user to understand and interpret the models outputs based on the confusion matrices.

- **Integration of performance break down and impact analysis with saliency/attention maps:** a specific case of integration of interpretability techniques and black-box analysis occurs in CV: DNNs interpretability approaches compute saliency or attention maps which highlight the pixels that are more important for the classification of the image. The saliency/attention maps could help the designer discover some properties of the input that affect prediction performance. Such properties could then be added as annotations to the input data set and used by the performance break down functionality to quantify their impact on the prediction performances. A step in this direction is the SECA system [28], which uses crowdsourced conceptual labels associated to saliency maps to enable explanation queries about the inference made by DNNs for image classification. The interpretability factors derived in this way could be exploited for error diagnosis and performance break down bridging the gap between performance-oriented and interpretation-oriented model evaluation.
- **Integration of error diagnosis and runtime performance analysis:** at the beginning of their diffusion DL models were designed to obtain the best possible performance on GPU-accelerated systems. With the emergence of mobile and edge AI applications the portability of DL models to constrained hardware (e.g., mobile phones and embedded devices) has become a major research topic [85]. The profiling of models (e.g., with metrics based on throughput, occupied memory, latency, or operation level) helps optimize architectures and support math-limited or memory-limited devices. Some

<sup>3</sup><https://www.eclipse.org>

<sup>4</sup><https://www.jetbrains.com/>

techniques have been proposed, such as model pruning and quantization, but their automatic application and the parameters optimization search are still ongoing research. Only one tool [55] among the surveyed ones integrates runtime performance analysis and some elementary diagnosis functions such as accuracy checking and model comparison. Given that in hardware-constrained scenarios the trade off between accuracy and runtime performances is a prominent concern, integrating the two perspectives would produce a constraint-aware tool extending error diagnosis and performance break down to scenarios with hardware limitations.

- **Model design guidance:** deep neural models are not mere data driven tools. They embody a lot of prior knowledge expressed implicitly in the structure of the architecture, in the selection of the operators and in the definition of the training strategy and of the data set. A recent research field, the so called automated deep learning (AutoDL), addresses the *combined selection and hyperparameter optimization of classification algorithms* [86, 87]. AutoDL research investigates the design patterns that can be applied to support or even minimize the human effort in the definition of optimal DL models for a variety of tasks [88, 89]. Current methods mostly rely on neural architecture search, which applies a rather brute force space exploration approach to the distillation of the best model for a given task. An extremely interesting evolution of the future error diagnosis frameworks would be to turn them into tools capable not only of diagnosing problems, but also to recommend model improvements. This would require a mapping between the model weaknesses identified by error diagnosis and a portfolio of model refactoring and improvement operators distilling the current wisdom of manual and automated DL design. Such a progress would somehow reconcile the practices of software development and data driven model design, which are now regarded as completely secluded and move the AutoDL field beyond the mere architecture search.
- **New task and data types:** As shown in Table 1 and in the brief overview of Section 4.7, the surveyed frameworks focus mostly on classification and CV localization tasks. The black-box

error diagnosis approach and tools could benefit also other domains especially those featuring complex data and non trivial performance indicators.

## 6 Conclusions

This survey presented the existing tools for the black-box diagnosis of errors in DL models for CV tasks. Major properties that would guide the user choice have been discussed and analyzed: supported tasks and media types, implemented metrics and analyses, performance break down functionalities, customization capabilities and openness. Novelties, advantages and disadvantages of the surveyed works have been described to provide the reader with a clear and up to date view of the field. Metrics, analyses and functionalities have been collected from each work and grouped by task to ease the comparison between the tools with a common focus. Several issues emerged from the survey have been discussed and the most promising research directions have been highlighted to help improve the present state of the art.

**Acknowledgments.** This work is partially supported by the project “PRECEPT - A novel decentralized edge-enabled PREsCriptivE and ProacTive framework for increased energy efficiency and well-being in residential buildings” funded by the EU H2020 Programme, grant agreement no. 958284.

## Declarations

**Conflict of interest.** The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

**Data availability.** Data sharing not applicable to this article as no datasets were generated or analysed during the current study

## List of Abbreviations

<b>AD</b>	Action Detection
<b>AI</b>	Artificial Intelligence
<b>AP</b>	Average Precision
<b>AUC</b>	Area Under the Curve
<b>CAM</b>	Class Activation Map
<b>CL</b>	Classification
<b>CV</b>	Computer Vision
<b>DNN</b>	Deep Neural Network
<b>ET</b>	Error Type
<b>FN</b>	False Negative
<b>FP</b>	False Positive
<b>GT</b>	Ground Truth
<b>IoU</b>	Intersection over Union
<b>IS</b>	Instance Segmenttion
<b>MAE</b>	Mean Absolute Error
<b>mAP</b>	mean Average Precision
<b>ME</b>	Mean Error
<b>ML</b>	Machine Learning
<b>MSE</b>	Mean Squared Error
<b>NAB</b>	Numenta Anomaly Benchmark
<b>NLP</b>	Natural Language Processing
<b>OD</b>	Object Detection
<b>OT</b>	Object Tracking
<b>PE</b>	Pose Estimation
<b>PR</b>	Precision-Recall
<b>RMSE</b>	Root Mean Squared Error
<b>ROC</b>	Receiver Operating Characteristic
<b>RS</b>	Recommender Systems
<b>SS</b>	Semantic Segmentation
<b>TN</b>	True Negative
<b>TP</b>	True Positive
<b>TS</b>	Time Series
<b>VRD</b>	Video Relation Detection

## References

- [1] Liu W, Wang Z, Liu X, et al (2017) A survey of deep neural network architectures and their applications. *Neurocomputing* 234:11–26
- [2] Chiroma H, Abdullahi UA, Alarood AA, et al (2018) Progress on artificial neural networks for big data analytics: a survey. *IEEE Access* 7:70,535–70,551
- [3] Voulodimos A, Doulamis N, Doulamis A, et al (2018) Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience* 2018
- [4] Gharibi G, Walunj V, Nekadi R, et al (2021) Automated end-to-end management of the modeling lifecycle in deep learning. *Empirical Software Engineering* 26(2):1–33
- [5] Doshi-Velez F, Kim B (2017) Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:170208608
- [6] Guidotti R, Monreale A, Ruggieri S, et al (2019) A survey of methods for explaining black box models. *ACM Comput Surv* 51(5):93:1–93:42
- [7] Zhang Qs, Zhu SC (2018) Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering* 19(1):27–39
- [8] Montavon G, Samek W, Müller KR (2018) Methods for interpreting and understanding deep neural networks. *Digital Signal Processing* 73:1–15
- [9] Carvalho DV, Pereira EM, Cardoso JS (2019) Machine learning interpretability: A survey on methods and metrics. *Electronics* 8(8)
- [10] Tjoa E, Guan C (2021) A survey on explainable artificial intelligence (XAI): toward medical XAI. *IEEE Trans Neural Networks Learn Syst* 32(11):4793–4813
- [11] Barredo Arrieta A, Gil-Lopez S, Laña I, et al (2021) On the post-hoc explainability of deep echo state networks for time series forecasting, image and video classification. *Neural Computing and Applications* pp 1–21
- [12] Zhou B, Khosla A, A. L, et al (2016) Learning Deep Features for Discriminative Localization. *CVPR*
- [13] Selvaraju RR, Cogswell M, Das A, et al (2017) Grad-cam: Visual explanations from deep networks via gradient-based localization. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp 618–626

- [14] Chattopadhyay A, Sarkar A, Howlader P, et al (2018) Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)
- [15] Sun KH, Huh H, Tama BA, et al (2020) Vision-based fault diagnostics using explainable deep learning with class activation maps. *IEEE Access* 8:129,169–129,179
- [16] Bae W, Noh J, Kim G (2020) Rethinking class activation mapping for weakly supervised object localization. In: Vedaldi A, Bischof H, Brox T, et al (eds) *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV, Lecture Notes in Computer Science*, vol 12360. Springer, pp 618–634
- [17] Linardatos P, Papastefanopoulos V, Kotsiantis S (2020) Explainable ai: A review of machine learning interpretability methods. *Entropy* 23(1):18
- [18] Verma S, Dickerson J, Hines K (2020) Counterfactual explanations for machine learning: A review. arXiv preprint arXiv:201010596
- [19] Stepin I, Alonso JM, Catala A, et al (2021) A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence. *IEEE Access* 9:11,974–12,001
- [20] Mehrabi N, Morstatter F, Saxena N, et al (2021) A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)* 54(6):1–35
- [21] Wu X, Hu Z, Pei K, et al (2021) Methods for deep learning model failure detection and model adaption: A survey. In: 2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), IEEE, pp 218–223
- [22] Wang Z, Liu K, Li J, et al (2019) Various frameworks and libraries of machine learning and deep learning: a survey. *Archives of computational methods in engineering* pp 1–24
- [23] Gilpin LH, Bau D, Yuan BZ, et al (2018) Explaining explanations: An overview of interpretability of machine learning. In: 2018 IEEE 5th International Conference on data science and advanced analytics (DSAA), IEEE, pp 80–89
- [24] Choo J, Liu S (2018) Visual analytics for explainable deep learning. *IEEE computer graphics and applications* 38(4):84–92
- [25] Roscher R, Bohn B, Duarte MF, et al (2020) Explainable machine learning for scientific insights and discoveries. *IEEE Access* 8:42,200–42,216
- [26] Molnar C (2022) *Interpretable Machine Learning*, 2nd edn. Independent publisher
- [27] Pessach D, Shmueli E (2022) A review on fairness in machine learning. *ACM Computing Surveys (CSUR)* 55(3):1–44
- [28] Balayn A, Soilis P, Lofi C, et al (2021) What do you mean? interpreting image classification with crowdsourced concept extraction and analysis. In: Leskovec J, Grobelnik M, Najork M, et al (eds) *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19–23, 2021. ACM / IW3C2*, pp 1937–1948
- [29] Page MJ, McKenzie JE, Bossuyt PM, et al (2021) The prisma 2020 statement: an updated guideline for reporting systematic reviews. *International Journal of Surgery* 88:105,906
- [30] Falagas ME, Pitsouni EI, Malietzis GA, et al (2008) Comparison of pubmed, scopus, web of science, and google scholar: strengths and weaknesses. *The FASEB journal* 22(2):338–342
- [31] Dollár P, Wojek C, Schiele B, et al (2009) Pedestrian detection: A benchmark. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, pp 304–311

- [32] Hoiem D, Chodpathumwan Y, Dai Q (2012) Diagnosing error in object detectors. In: European conference on computer vision, Springer, pp 340–353
- [33] Russakovsky O, Deng J, Huang Z, et al (2013) Detecting avocados to zucchinis: what have we done, and where are we going? In: Proceedings of the IEEE International Conference on Computer Vision, pp 2064–2071
- [34] Lin TY, Maire M, Belongie S, et al (2014) Microsoft coco: Common objects in context. In: Fleet D, Pajdla T, Schiele B, et al (eds) Computer Vision – ECCV 2014. Springer International Publishing, Cham, pp 740–755
- [35] Hariharan B, Arbeláez P, Girshick R, et al (2014) Simultaneous detection and segmentation. In: European conference on computer vision, Springer, pp 297–312
- [36] Zhu H, Lu S, Cai J, et al (2015) Diagnosing state-of-the-art object proposal methods. arXiv preprint arXiv:150704512
- [37] Amershi S, Chickering M, Drucker SM, et al (2015) Modeltracker: Redesigning performance analysis tools for machine learning. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, pp 337–346
- [38] Redondo-Cabrera C, López-Sastre RJ, Xiang Y, et al (2016) Pose estimation errors, the ultimate diagnosis. In: European Conference on Computer Vision, Springer, pp 118–134
- [39] Krause J, Perer A, Ng K (2016) Interacting with predictions: Visual inspection of black-box machine learning models. In: Proceedings of the 2016 CHI conference on human factors in computing systems, pp 5686–5697
- [40] Zhang S, Benenson R, Omran M, et al (2016) How far are we from solving pedestrian detection? In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1259–1267
- [41] Ruggero Ronchi M, Perona P (2017) Benchmarking and error diagnosis in multi-instance pose estimation. In: Proceedings of the IEEE international conference on computer vision, pp 369–378
- [42] Krause J, Dasgupta A, Swartz J, et al (2017) A workflow for visual diagnostics of binary classifiers using instance-level explanations. In: 2017 IEEE Conference on Visual Analytics Science and Technology (VAST), IEEE, pp 162–172
- [43] Ren D, Amershi S, Lee B, et al (2016) Squares: Supporting interactive performance analysis for multiclass classifiers. IEEE transactions on visualization and computer graphics 23(1):61–70
- [44] Sigurdsson GA, Russakovsky O, Gupta A (2017) What actions are needed for understanding human actions in videos? In: Proceedings of the IEEE international conference on computer vision, pp 2137–2146
- [45] Alwassel H, Heilbron FC, Escorcia V, et al (2018) Diagnosing error in temporal action detectors. In: Proceedings of the European Conference on Computer Vision (ECCV), pp 256–272
- [46] Nekrasov V, Shen C, Reid I (2018) Diagnostics in semantic segmentation. arXiv preprint arXiv:180910328
- [47] Zhang J, Wang Y, Molino P, et al (2018) Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. IEEE transactions on visualization and computer graphics 25(1):364–373
- [48] Wexler J, Pushkarna M, Bolukbasi T, et al (2019) The what-if tool: Interactive probing of machine learning models. IEEE transactions on visualization and computer graphics 26(1):56–65
- [49] Bolya D, Foley S, Hays J, et al (2020) Tide: A general toolbox for identifying object detection errors. In: European Conference on Computer Vision, Springer, pp 558–573

- [50] Torres RN, Fraternali P, Romero J (2020) Odin: An object detection and instance segmentation diagnosis framework. In: European Conference on Computer Vision, Springer, pp 19–31
- [51] Torres RN, Milani F, Fraternali P (2021) Odin: Pluggable meta-annotations and metrics for the diagnosis of classification and localization. In: International Conference on Machine Learning, Optimization, and Data Science, Springer, pp 383–398
- [52] Padilla R, Netto SL, da Silva EA (2020) A survey on performance metrics for object-detection algorithms. In: 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), IEEE, pp 237–242
- [53] Yoon H, Lee SH, Park M (2020) Tensorflow with user friendly graphical framework for object detection api. arXiv preprint arXiv:200606385
- [54] Gleicher M, Barve A, Yu X, et al (2020) Boxer: Interactive comparison of classifier results. In: Computer Graphics Forum, Wiley Online Library, pp 181–193
- [55] Demidovskij A, Tugaryov A, Kashchikhin A, et al (2021) Openvino deep learning workbench: Towards analytical platform for neural networks inference optimization. In: Journal of Physics: Conference Series, IOP Publishing, p 012012
- [56] Padilla R, Passos WL, Dias TL, et al (2021) A comparative analysis of object detection metrics with a companion open-source toolkit. *Electronics* 10(3):279
- [57] Fan H, Yang F, Chu P, et al (2021) Tracklinic: Diagnosis of challenge factors in visual tracking. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp 970–979
- [58] Chen S, Pascal M, Snoek CG (2021) Diagnosing errors in video relation detectors. In: BMVC
- [59] Kräter M, Abuhattum S, Soteriou D, et al (2021) Aideveloper: deep learning image classification in life science and beyond. *Advanced science* 8(11):2003,743
- [60] Nourani M, Roy C, Honeycutt DR, et al (2022) Detoxer: A visual debugging tool with multi-scope explanations for temporal multi-label classification. *IEEE Computer Graphics and Applications*
- [61] Deng Z, Sun H, Zhou S, et al (2018) Multi-scale object detection in remote sensing imagery with convolutional neural networks. *ISPRS Journal of Photogrammetry and Remote Sensing* 145:3–22. *Deep Learning RS Data*
- [62] Shang X, Ren T, Guo J, et al (2017) Video visual relation detection. In: Proceedings of the 25th ACM international conference on Multimedia, pp 1300–1308
- [63] Shang X, Di D, Xiao J, et al (2019) Annotating objects and relations in user-generated videos. In: Proceedings of the 2019 International Conference on Multimedia Retrieval, pp 279–287
- [64] Pang G, Shen C, Cao L, et al (2021) Deep learning for anomaly detection: A review. *ACM Comput Surv* 54(2)
- [65] Chalapathy R, Chawla S (2019) Deep learning for anomaly detection: A survey. arXiv preprint arXiv:190103407
- [66] Zhang W, Yang D, Wang H (2019) Data-driven methods for predictive maintenance of industrial equipment: A survey. *IEEE Systems Journal* 13(3):2213–2227
- [67] Vollert S, Atzmueller M, Theissler A (2021) Interpretable machine learning: A brief survey from the predictive maintenance perspective. In: 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), pp 01–08
- [68] Zoppi T, Ceccarelli A, Bondavalli A (2019) Evaluation of anomaly detection algorithms made easy with reload. In: 2019 IEEE 30th

- International Symposium on Software Reliability Engineering (ISSRE), IEEE, pp 446–455
- [69] Herzen J, Lässig F, Piazzetta SG, et al (2021) Darts: User-friendly modern machine learning for time series. arXiv preprint arXiv:211003224
- [70] Carrasco J, López D, Aguilera-Martos I, et al (2021) Anomaly detection in predictive maintenance: A new evaluation framework for temporal unsupervised anomaly detection algorithms. *Neurocomputing* 462:440–452
- [71] Krokotsch T, Knaak M, Gühmann C (2020) A novel evaluation framework for unsupervised domain adaption on remaining useful lifetime estimation. In: 2020 IEEE International Conference on Prognostics and Health Management (ICPHM), IEEE, pp 1–8
- [72] Zangrando N, Torres RN, Milani F, et al (2022) Odin ts: a tool for the black-box evaluation of time series analytics. In: Conference proceedings ITISE, Springer
- [73] Gralinski F, Wróblewska A, Stanisławek T, et al (2019) Geval: Tool for debugging nlp datasets and models. In: Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pp 254–262
- [74] Tenney I, Wexler J, Bastings J, et al (2020) The language interpretability tool: Extensible, interactive visualizations and analysis for nlp models. arXiv preprint arXiv:200805122
- [75] Manabe H, Hagiwara M (2021) Expats: A toolkit for explainable automated text scoring. arXiv preprint arXiv:210403364
- [76] Zhao WX, Mu S, Hou Y, et al (2021) Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp 4653–4664
- [77] Anelli VW, Bellogín A, Ferrara A, et al (2021) Elliot: a comprehensive and rigorous framework for reproducible recommender systems evaluation. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp 2405–2414
- [78] Monteiro FC, Campilho AC (2006) Performance evaluation of image segmentation. In: International Conference Image Analysis and Recognition, Springer, pp 248–259
- [79] Hossin M, Sulaiman M (2015) A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process* 5(2):1
- [80] Novaković JD, Veljović A, Ilić SS, et al (2017) Evaluation of classification models in machine learning. *Theory and Applications of Mathematics & Computer Science* 7(1):39–46
- [81] Milani F, Fraternali P (2021) A dataset and a convolutional model for iconography classification in paintings. *Journal on Computing and Cultural Heritage (JOCCH)* 14(4):1–18
- [82] Petsiuk V, Jain R, Manjunatha V, et al (2020) Black-box explanation of object detectors via saliency maps. arXiv preprint arXiv:200603204
- [83] Theissler A, Thomas M, Burch M, et al (2022) Confusionvis: Comparative evaluation and selection of multi-class classifiers based on confusion matrices. *Knowledge-Based Systems* 247:108,651
- [84] Theissler A, Vollert S, Benz P, et al (2020) MI-modeexplorer: An explorative model-agnostic approach to evaluate and compare multi-class classifiers. In: International Cross-Domain Conference for Machine Learning and Knowledge Extraction, Springer, pp 281–300
- [85] Chen Y, Zheng B, Zhang Z, et al (2020) Deep learning on mobile and embedded devices: State-of-the-art, challenges, and future directions. *ACM Comput Surv* 53(4):84:1–84:37

- [86] Talbi EG (2021) Automated design of deep neural networks: A survey and unified taxonomy. *ACM Comput Surv* 54(2)
- [87] Thornton C, Hutter F, Hoos HH, et al (2013) Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp 847–855
- [88] Liu Z, Xu Z, Rajaa S, et al (2020) Towards automated deep learning: Analysis of the autodl challenge series 2019. In: *NeurIPS 2019 Competition and Demonstration Track*, PMLR, pp 242–252
- [89] Dong X, Kedziora DJ, Musial K, et al (2021) Automated deep learning: Neural architecture search is not the end. *arXiv preprint arXiv:211209245*