

Robust Traffic Sign Recognition Against Camera Failures

MUHAMMAD ATIF¹, ANDREA CECCARELLI¹, TOMMASO ZOPPI¹, MOHAMAD GHARIB²,
AND ANDREA BONDAVALLI¹ (Member, IEEE)

¹Department of Mathematics and Informatics, University of Florence, 50134 Florence, Italy

²Institute of Computer Science, University of Tartu, 51009 Tartu, Estonia

CORRESPONDING AUTHOR: M. ATIF (e-mail: muhammad.atif@unifi.it)

This work was supported in part by the H2020 Programme through the Marie Skłodowska-Curie Grant (ADVANCE) Project under Agreement 823788, and in part by the Regione Toscana through the POR FESR Project under Grant 2014-2020 SPaCe.

ABSTRACT Failures of the vehicle camera may compromise the correct acquisition of frames, that are subsequently used by autonomous driving tasks. A clear understanding of the behavior of the autonomous driving tasks under such failure conditions, together with strategies to avoid safety is jeopardized, are indeed necessary. This study analyses and improve the performance of Traffic Sign Recognition (TSR) systems for road vehicles under the possible occurrence of camera failures. Our experimental assessment relies on three public datasets, which are commonly used for benchmarking TSR systems. We artificially inject 13 different types of camera failures into the three datasets. Then, we exploit three deep neural networks (DNNs) to classify either a single frame of a traffic sign or a sequence (i.e., a sliding window) of frames. We show that sliding windows significantly improves the robustness of the classifier against altered frames. We confirm our observations through explainable AI, which allows understanding why different classifiers have different performance in case of camera failures.

INDEX TERMS Traffic sign recognition, camera failures, deep learning, sliding windows, meta learning, robustness.

I. INTRODUCTION

TRAFFIC Sign Detection and Recognition (TSDR) allows Advanced Driver Assistance Systems (ADAS) [1], [2], [3] to safely operate (semi-) autonomous vehicles on roads. TSDR is generally split into two stages: detection and recognition of traffic signs. Detection [4], [5] means locating traffic sign in an input scene image, while recognition [6], [7] is concerned about identifying the type of traffic sign. The detection and recognition of traffic signs are treated as completely separate tasks [8], [9]. There are many state of the art approaches adopted for traffic sign detection [10], [11], [12]. This study focus is on the later part of the TSDR system, which is recognition of traffic signs. Particularly, Traffic Sign Recognition (TSR) assists the driver by automatically providing information about traffic

signs to improve driving and road safety. Due to distraction, fatigue, or adverse operating conditions, human drivers can miss or misinterpret an important traffic sign [13], [14] potentially leading to dangerous situations.

Automatic TSR systems embed Machine Learning (ML), and especially Deep Neural Network (DNN) classifiers which process image frames captured by cameras installed on the vehicle. Those TSR systems are known to accurately classify traffic signs, even reaching perfect classification performance under nominal operating conditions [15], [16], [17]. Unfortunately, the adverse environmental conditions, or the malfunctions of the camera, may produce low-quality frames that may negatively impact the performance of classifiers. Examples include, but are not limited to: occlusions, shadows, defects of the camera lens, changes in environmental light, raindrops on the camera lens, out-of-focus, flare [18], [19]. Therefore, to guarantee safety of the driving task, it is necessary to study

The review of this article was arranged by Associate Editor Chi-Hua Chen.

the robustness of TSR systems against the aforementioned threats, and develop solutions to tolerate them [20], [21]. This paper accomplishes this tasks, first reviewing the challenges and the state of the art, and then proposing and evaluating alternative solutions.

The paper proceeds as follow. First, we review related works on TSR and we describe our two main strategies to classification: *single-frame* and *sliding window*. The former strategy feeds a single frame into a classifier, which makes its prediction on the type of traffic sign contained in the frame. The latter strategy exploits the natural progression of a vehicle on the road, which gets gradually close to a traffic sign. This way, the classifier is fed with a sliding window of many frames, and it builds its prediction on the traffic sign using more knowledge. This last strategy requires a meta-learner [22] which summarizes the information achieved from the sliding window.

Then, in our study we corrupt frames by applying camera failures that are successively fed to single-frame classifiers and sliding windows classifiers. Frames are taken from three publicly available benchmark datasets: German Traffic Sign Dataset (GTSRB) [23], Dataset of Italian Traffic Signs (DITS) [24], and Belgium Traffic Sign Dataset (BelgiumTSC) [25]. These datasets include sequences of frames and consequently allow applying both classification strategies. We simulated 13 camera failures, each one actionable with different parameters, obtaining a total of 103 failures configuration to be individually injected into each frame of the three datasets.

The produced data allows examining the robustness of three DNNs: AlexNet [26], InceptionV3 [27], and MobileNetV2 [28]. We apply them independently to perform classification, and we also run them in parallel; in this second case, classification is performed by a stacking meta-learner which is fed with the outputs of the classifiers. Further, we apply two different classification strategies: single frame classification and sliding window classification.

Our results indicate that the occurrence of camera failures degrades classification performance, especially for single-frame classifiers. Instead, approaches based on the sliding window are significantly more robust.

Further, we dig into the results using LIME [29], a toolbox for explainable Artificial Intelligence (AI). Explainable AI allows understanding how a classifier uses the input frame to derive its output: we use LIME to explain why the injection of the camera failures alters the behavior of the classifiers, and why certain classifiers are more robust than others against camera failures.

The rest of the paper is organized as follows. Section II provides background on TSR systems and camera failures. Section III expands the TSR strategies we apply in this paper to classify traffic sign, while Section IV provides details on the methodology, selected classifiers, datasets, failure injection strategies, and performance metrics. Section V reports on experimental results achieved using the different TSR strategies, and Section VI verifies the result using explainable

AI. Finally, Section VII concludes the paper with the main findings and our ongoing works, that aim to increase the robustness of TSR through data augmentation (i.e., frames with injected failures are added in the training phase of the model) and by creating a failure detector to alert the TSR system about imperfect frames.

II. BACKGROUND AND RELATED WORKS

This section provides background and related works on ML classifiers for TSR and on the analysis of frames that are corrupted because of camera failures.

A. TRAFFIC SIGN RECOGNITION

In the last decade, researchers and practitioners used non-deep ML classifiers for single-frame TSR [30], [31], [32]. Supervised classifiers such as Support Vector Machines (SVMs, [33]), Decision Trees (and ensembles, [34], [35]) or Nearest Neighbors (KNN, [36]) cannot directly process frames. Therefore, those frames are first pre-processed to extract numerical features according to specific feature descriptors as Histogram of Oriented Gradients (HOG, [37]) or Local Binary Patterns (LBP, [38]). Then, those features are provided to the abovementioned classifiers. Many non-deep ML classifiers are already available for automatic TSR systems. In [39], the authors extract HOG features for the GTSDB dataset to train different classifiers. Similarly, in [40], HOG features are fed to Support Vector Machines (SVMs) to identify the boundary separation of six groups of traffic signs, which are then classified using a DNN. Yang et al. [41] compare different supervised classifiers trained with color fused features; they achieve the highest accuracy with Gentle Adaboost. Authors of [70] proposed a novel feature descriptor that combine multiple features using fusion technique to improve the performance of human action recognition system. In one another study [69], authors present six different fusion models to improve the performance of human action recognition system. Agrawal and Chaurasiya [42] apply principal component analysis (PCA) to reduce the dimensions of HOG feature descriptors for the classification of the GTSRB dataset into 3 subcategories, i.e., danger, mandatory, and denial. Stallkamp et al. [43] extract HOG features to be fed into Random Forest and LDA. In [44], the authors perform TSR using semi-supervised learning to utilize a large amount of unlabeled data and a small amount of labeled data to train a classifier which is more robust to data imbalance. In [23], the authors report on a competition where both humans and ML classifiers targeted frames of the GTSRB dataset: ML algorithms achieve an accuracy of 98.98%, which is comparable to the accuracy of humans on such dataset.

Additionally, many studies explore the application of DNNs to TSR. Classifiers for TSR exploited DNNs such as AlexNet [26], InceptionV3 [27], MobileNetV2 [28], googLeNet [45]. DNNs do not require any explicit feature extraction: features are extracted through convolutional layers during the training process. In [46], the authors use

InceptionV3 [27] with transfer learning to obtain an accuracy of 99.18% on the BelgiumTSC dataset, while in [47], the VGG-16 DNN model is specialized for TSR by adding Batch Normalization and global average pooling layer, and removing few superfluous convolutional layers. The authors of [15] compare the performance of supervised classifiers including DNNs. In their experiments, three datasets, i.e., GTSRB, BelgiumTSC, and DITS are considered to classify traffic signs into three broad categories, i.e., red triangular, blue circular, and red circular. In [48], the authors develop a real-time DNN-based TSR system and they deploy it on the embedded system. The developed Convolutional Neural Network (CNN) architecture achieves 99.71% accuracy. In [2], a fifteen-layers DNN is utilized to achieve a detection accuracy of 96.5% on the GTSRB dataset. In the study [49], ensembles of Convolutional Neural networks are utilized on GTSRB and on BelgiumTSC to achieve accuracy higher than 99% on the circular traffic signs. Lastly, the authors of [50] compare 5 DNNs i.e., Xception, EfficientNetB0, ResNet50, VGG16, and InceptionV3 trained with transfer learning, achieving the highest accuracy (95.04%) on the GTSRB dataset.

Feature extraction and neural networks can even be combined: Jose et al. [51] combine DNN and Viola-Jones framework for feature extraction. Abizada [52] uses two feature sets, i.e., HOG and features extracted through convolutional layers of a DNN, to classify the GTSRB dataset and achieved a test accuracy of 98.20%. In [53], the authors propose an approach for traffic sign detection and recognition based on two stages: first, they identify the shape using HOG features and linear SVM, then they use a CNN to recognize the specific traffic sign.

The majority of works in the literature processes a frame using a unique classifier, except few studies [17], [54], [55] which process sequences of frames.

B. CAMERA FAILURES

Autonomous vehicles contain visual cameras that observe the environment and capture sequences of frames. Several studies in the literature utilize a visual camera for autonomous driving tasks, such as obstacle detection, pedestrian detection and lane detection, and obviously traffic sign recognition [1], [13], [56], [57].

If the camera has a malfunction, the quality of the captured frames is degraded, and this may result in a critical situation and possible serious consequences. The possible occurrence of camera failures must be considered to guarantee the safe operation of autonomous vehicles [18]. These failures may occur in any component of the camera, and especially they may be malfunctions of the lens, the image sensor, or the ISP (Image Signal Processor): all these components cooperate to create every frame produced by the camera.

Only few works focused specifically on the effect that camera failures may have on the produced frames. In [19], the authors adopted a CNN to deal with different harsh conditions such as out-of-focus, illumination, and missing

information applied on the GTSRB dataset. Authors apply an attention mechanism to build a convolutional pooling for performance improvement. In [18], authors systematically define different failure modes of a vehicular camera, and they discuss the visible effects on the captured frames. Also, they propose a Python library to inject failures into frames. Lastly, Morozov et al. [58], while not strictly considering the image acquisition, simulate hardware failures: they used three CNNs for classification and a Bayesian Network for the analysis of the trustworthiness of results.

In this work, we use the failure categorization of [18] as reference, because we believe it is the most complete representation of camera failures available in the state of the art, and it is supported by a software library that allows reproducing the failure effects on target images.

III. TSR STRATEGIES UNDER CONSIDERATION

This section explains in detail the TSR classification strategies we consider: single-frame classification and sliding windows classification.

A. SINGLE-FRAME CLASSIFICATION

A single frame is usually classified using one classifier. In addition, it is possible to combine multiple classifiers in a two-step process, relying on a stacking meta-learner to decide on the class [22] (Fig. 1). In stacking, first, the frame is fed to multiple base-level classifiers, then the individual predictions from the classifiers are processed by a classifier at a meta-level; in other words, the predictions of parallel classifiers are input meta-features to a meta-level classifier that provides the final prediction.

Fig. 1 exemplifies this process considering as input a frame of a STOP traffic sign and three base-level classifiers: AlexNet, InceptionV3, and MobileNetV2. Each of the three classifiers outputs an array of probabilities $PTS_k = \{pts_{ik}, 1 \leq i \leq n\}$ where n is the number of possible classes, and k is the k -th base-level classifier (in Fig. 1, we use $1 \leq k \leq 3$). Each pts_{ik} describes the probability that the input frame contains a traffic sign of the i -th class, according to the k -th base-level classifier. All $pts_{ik} \in PTS_k$ sum up to 1. The three classifiers process the input frame independently and provide their own probabilities PTS_1 , PTS_2 , and PTS_3 for that input. These probabilities are then aggregated into a unique meta-feature set of $3n$ features and delivered as input to the meta-level classifier, which provides its own PTS_{final} probabilities that are used for TSR. Noticeably, the meta-level classifier plays the role of an adjudicator rather than working as a classifier itself.

In this paper, we consider the following three classification strategies:

- Single-frame classifier on a single frame (**1SFC**, One Single Frame Classifier). In this case, we are using just one classifier to perform classification.
- Two single-frame classifiers on a single frame (**2SFC**, Two Single Frame Classifiers). This approach relies on

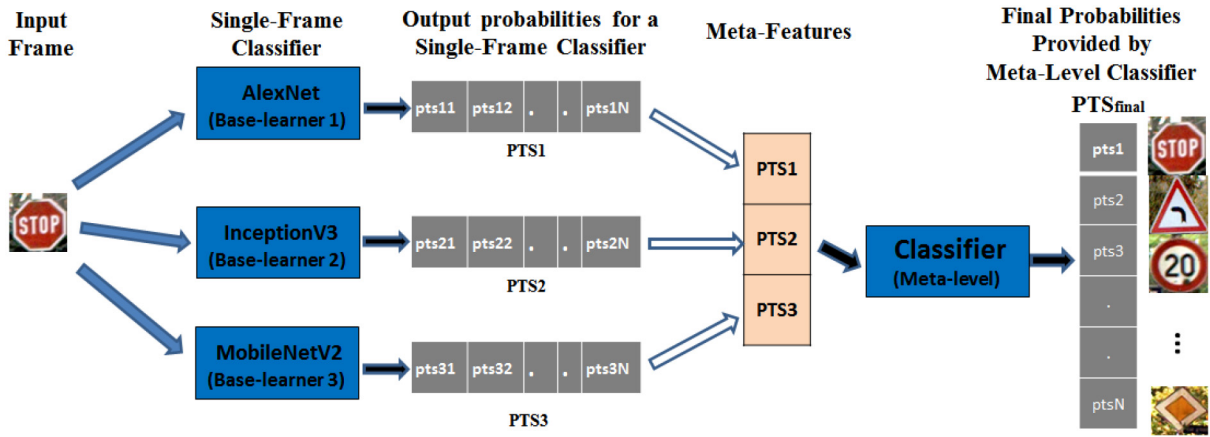


FIGURE 1. A TSR system which uses multiple (three) single-frame classifiers with stacking.

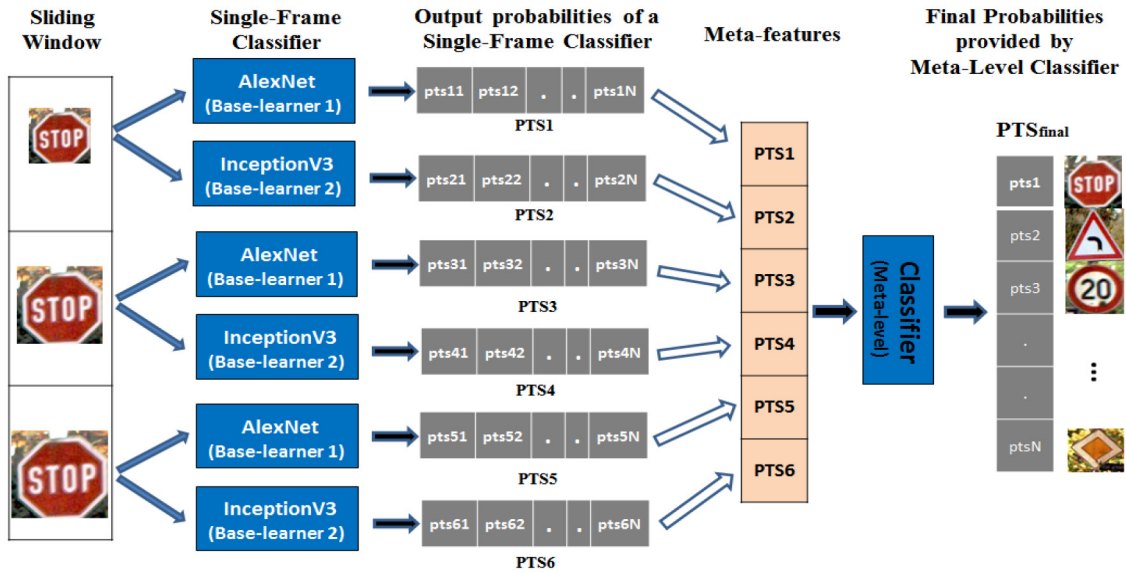


FIGURE 2. A TSR which uses sliding windows of three frames (on the left) with two base-classifiers (AlexNet, InceptionV3) and a meta-level classifier that produces the final classification result PTS_{final} .

stacking the predictions of the two base-level classifiers, relying on a stacking meta-level classifier.

- Three single-frame classifiers on a single frame (3SFC, Three Single Frame Classifiers). This approach corresponds to Fig. 1. It relies on stacking the predictions of the three base-level classifiers, using the stacking meta-level classifier.

B. CLASSIFICATION WITH A SLIDING WINDOW

In addition, we build classifiers that can process multiple frames in a sliding window [17]. The structure of this classifier is described in Fig. 2. Sliding windows contain multiple frames captured at different time instants or from multiple cameras at the same time. Therefore, we need to slightly adapt the classifier from Section III-A to process multiple frames in parallel.

For example, Fig. 2 shows a TSR system that uses a sliding window of three frames and that is composed of two

single-frame classifiers. First, each of the two single-frame classifiers processes each of the three frames individually, building six PTS_k arrays of probabilities $1 \leq k \leq 6$. These arrays constitute the meta-feature set that is provided to the stacking classifier. Finally, the stacking classifier produces the classification result PTS_{final} , which is the output of the TSR system.

In this paper, we consider the following classification strategies based on a sliding window:

- Sliding window is applied using only one classifier (W1C, Sliding Window with One Classifier).
- Sliding window is applied using two classifiers (W2C, Sliding Window with Two Classifiers). This corresponds to Fig. 2. In this case, all frames in the sliding window are individually classified by two base-level classifiers; the predictions of each frame are input meta-features to the meta-level classifier.

- Sliding window is applied using three classifiers (**W3C**, Sliding Window with Three Classifiers). All frames in the sliding window are individually classified by three base-level classifiers; the predictions of each frame are input meta-features to the meta-level classifier.

IV. APPROACH TO ROBUSTNESS EVALUATION

We detail the methodology we used to evaluate the robustness of the TSR strategies previously described, when camera failures occur. We describe the datasets of traffic signs, the failure injection strategies, and the single-frame and sliding-window classifiers that perform TSR on clean and injected frames.

A. METHODOLOGY

First, all classifiers described in Section IV-B, organized in groups 1SFC, 2SFC, 3SFC, W1C, W2C, W3C, are trained using the traffic sign datasets BelgiumTSC, GTSRB, and DITS described in Section IV-D. Then, the stacking meta-learners of Section IV-C are trained as well. This creates all the classification strategies 1SFC, 2SFC, 3SFC, W1C, W2C, W3C trained on the traffic sign datasets.

In the next phase, the camera failures described in Section IV-E are injected into each dataset individually, to create datasets of altered frames. We will call these datasets as injected datasets, opposed to the clean datasets, and the altered frames as injected frames, opposed to the clean frames. This creates a set of injected datasets, each with different failures. Noteworthy, some failures have multiple possible configuration (e.g., the amount of blur that is applied on the frames): we create injected datasets of GTSRB, BelgiumTSC and DITS for each configuration. This leads to a total of 103 configurations, each one repeated on the three datasets. These datasets are provided as test inputs to the 1SFC, 2SFC, 3SFC, W1C, W2C, W3C.

Finally, we measure the robustness of different TSR approaches against each camera failure. The robustness is described using the metrics discussed in Section IV-F.

B. BASE-LEVEL CLASSIFIERS

This study utilizes the following three DNN classifiers. We adapt them for TSR using transfer learning from models trained on ImageNet.

MobileNetV2 (MN2) [28] is a 53 layers DNN which consists of two main blocks, namely: downsizing, and residual blocks. Compared to other DNNs, MN2 is an efficient and lightweight network with fewer parameters that need fine-tuning.

AlexNet (AN) [26] is a convolutional neural network that consists of five convolutional layers and 3 fully connected layers, trained on ImageNet [59].

InceptionV3 (IC3) [27] is a 48 layers DNN trained using the ImageNet dataset [59]. IC3 consists of three main blocks:

the convolutional block, the inception module, and a classifier. IC3 uses a convolutional kernel to decrease the number of feature channels and speed up the training process.

Combination of classifiers (2SFC, 3SFC, W2C, W3C) includes all the possible pairs and triples from the three classifiers above.

C. META-LEVEL CLASSIFIERS

This section presents six supervised classifiers that suit the role of meta-level classifiers for stacking.

k-Nearest Neighbors (KNN) [36] assigns a label to a new data point based on how its “nearest neighbors” are labeled. Neighbors of a data point are calculated according to Euclidean distance; they contribute to derive the label for the new data point based on the majority. We used many different odd values of $k = \{1, 3, 5, 7, 11, 13, 15, 17, 19, 21\}$, which avoid ties when performing 8-class classification (for BelgiumTSC and GTSRB) and 9-class classification for DITS.

Support Vector Machines (SVM) [33] breaks down a multi class task into many smaller binary classification problems. SVMs learn hyper-planes that are then used to separate input space: the shape of those hyper-planes is defined by different kernels. In this study, we consider the simplest available kernel, building a Linear SVM.

Decision Tree (DT) builds a tree-like structure [60]. In the tree, rules applied to features are used to create new branches, down to leaf nodes, which contain the final output. A final pruning step usually helps to limit overfitting to the training set. In our experiments, we used the default tree depth of MATLAB, which assigns $MaxNumSplits = n - 1$, where n represents the size of the training sample, and no limit on the depth of the decision tree [61].

Linear Discriminant Analysis (LDA) relies on a generalization of Fisher’s linear discriminant. Such discriminant derives a vector that can be used to separate classes by building a linear combination between features [62]. As usual, we train LDA with a pseudo-linear discriminant.

AdaBoostM2 (ABM2) [34] combines multiple weak learners into a unique strong learner to improve classification performance through boosting meta-learning. Weak learners, often called decision stumps, are usually decision trees with very limited depth; in our experiments, we exercised ABM2 as an ensemble of 100 (weak) decision trees, with $MaxNumSplits = training\ sample\ size - 1$, with no depth limits on decision trees.

Random Forests (RF) [35] embed multiple decision trees using Bagging: therefore, each decision tree is trained on a subset of the training set which is obtained by random sampling with replacement. This study builds random forests composed of 100 decision trees.

D. TRAFFIC SIGN DATASETS

This study builds upon three datasets containing sequences of traffic signs: i) the BelgiumTSC dataset [25], ii) the GTSRB dataset [23], and iii) the Dataset of Italian Traffic



FIGURE 3. Injection of 13 different visual camera failures to a sample traffic sign. (best viewed in color).

TABLE 1. Details of the traffic signs datasets.

Dataset	Training Images	Testing Images	Sequence Length	Time Ordered
GTSRB	39210	12570	30	✓
BelgiumTSC	4581	2505	3	
DITS	7500	1159	2-15	✓

Signs (DITS) [24]. GTSRB and DITS contain time-ordered sequences of frames related to the same traffic sign, whereas BelgiumTSC contains three frames for each traffic sign captured at the same time from independent cameras from different viewpoints. Table 1 highlights the characteristics of the datasets used in this study: GTSRB contains more than 50,000 frames, structured in 1726 sequences of 30 traffic signs. DITS has a similar organization with sequences of 2 to 15 frames, whereas traffic signs in BelgiumTSC are organized in sequences of three frames.

Each dataset has its own categorization of traffic signs, which we homogenize into 9 classes that overlap across all three datasets, reported in Table 2. DITS contains frames for all 9 categories of traffic signs, GTSRB dataset misses category 8, i.e., Blue Rectangular traffic signs, while BelgiumTSC does not contain circular traffic signs, i.e., category 9.

E. STRATEGIES TO INJECT CAMERA FAILURES

This section provides a brief overview of the potential failures of a camera installed on vehicles. We considered a total of 13 failures, described in what follows. Failures may have multiple configurations; each failure and its configurations are applied on the three datasets. In total, this leads to 103 copies of each dataset, where all frames are modified according to the specified configuration. The source

code to reproduce the injection of failures is based on the library available at [63], which was customized where needed.

Banding. A frame with banding has many horizontal and/or vertical lines in the background. Fig. 3a shows the clean frame, while Fig. 3b shows the effect of applying banding failures on the frame.

Blurred. Blurred occurs when the captured frame is not properly focused by the camera lens (see Fig. 3c). Overall, we used 11 different configurations to produce blurred frames with different amounts of blurriness.










Brightness. This failure alters the brightness of the produced frame (Fig. 3d), ranging from no brightness (black frame) to full brightness (white frame) depending on malfunctions of the lens shutter, diaphragm, or iris. The same visual effect on the output frame can happen when the light enters the camera with a narrow angle (e.g., in case the sun is in front). We simulated 8 levels of brightness from a very dark frame to an almost white one.

No Chromatic Aberration Correction. This type of failure occurs when the ISP fails: halos appear on the edges and corners of the frame. The resulting frame (Fig. 3e) shows the blurred effect on the outer edges.

Condensation. Condensation on the lens happens when humidity levels are high, or because of rapid temperature changes. To inject a condensation failure, we overlapped condensation images of [64] to each of the frames in all datasets, as in Fig. 3f. We utilized 3 different overlaying images to produce three different effects of condensation failures.

No Bayer Filter. Should the Bayer filter [65] do not work properly, the acquired frame will result in a colorless frame as in Fig. 3g. In our experiments, we replicated this effect by changing the RGB (Red, Green, Blue) channels to convert a frame to grayscale.

TABLE 2. Categorization of traffic signs based on their shape, content, and color.

Category	1	2	3	4	5	6	7	8	9
Traffic Signs									
	Stop Sign	Red Triangular	Inverted Triangular	Speed Limit	Red Circular	Blue Circular	Diamond	Blue Rectangular	White Circular
GTSRB	✓	✓	✓	✓	✓	✓	✓		✓
BelgiumTSC	✓	✓	✓	✓	✓	✓	✓	✓	
DITS	✓	✓	✓	✓	✓	✓	✓	✓	✓

Dirt. This camera failure occurs when there is dirt on the internal or external lens. Fig. 3h shows a dirty frame generated after overlapping the dirt images of [64] to a traffic sign frame. This study uses 36 different dirt images.

Ice. This type of failure occurs when the temperature of the environment drops below freezing, affecting frame quality. Fig. 3i shows the ice effect on an acquired frame by using one out of the four different ice images of [64] that we overlapped to traffic sign frames.

No Demosaicing. The raw frame may be not processed by ISP for demosaicing: this creates the output frames by interpolating the mosaic of RGB colors produced by the image sensors. As a result, the frame remains pixelated (i.e., each pixel contains either red, green, or blue color channel values) as shown in Fig. 3j.

No Noise Reduction. When frames are captured by a camera, the ISP is responsible for removing noise. However, this process may fail: to simulate this event, we used 10 different configurations of speckle noise: an example is shown in Fig. 3k.

Rain. During rainy weather, the external lens of the camera may have small drops of water that degrade the quality of the acquired frame. We simulate this event by overlapping 5 different rain images from [64] to traffic sign frames; an example is in Fig. 3l.

Dead Pixels. Certain failures of the image sensor may produce an output frame with dark spots; visually, the effect is analogous to visualizing dead pixels. In this study we simulate different configurations, namely i) an array of vertical pixels is set to black, ii) horizontal and vertical arrays are set to black, or iii) sets of [50, 200, 500, 1000] pixels, randomly selected, are turned black. Fig. 3m shows different failure effects when they are applied on the clean frame.

Broken Lens. This failure occurs when one or more lenses of the camera break because of many reasons, e.g., debris that crashes against the lens. This may make scratches visible in the produced frame. We simulated a broken lens with 15 different overlaying images from [64]. An example is in Fig. 3n.

F. PERFORMANCE METRICS

To evaluate results, we rely on accuracy [66], [67], which consider each misclassification as equally harmful. Further,

we elaborate on accuracy by defining a metrics that measures the accuracy reduction. We define *accuracy drop* the difference between the accuracy obtained on the injected datasets and on the clean datasets. The accuracy drop is computed for each failure. Some of the failures have multiple configurations: in this case, for such failures we compute the minimum accuracy drop and the maximum accuracy drop.

V. EXPERIMENTAL RESULTS

We organize results as follows. Section V-A discusses the results achieved using the clean datasets GTSRB, BelgiumTSC, and DITS for single frame classifiers (1SFC, 2SFC, and 3SFC) and sliding window classifiers (W1C, W2C, and W3C). The successive sections instead consider the injected datasets: Section V-B describes the results of 1SFC, Section V-C of 2SFC and 3SFC, Section V-D of W1C, and Section V-E of W2C and W3C.

A. RESULTS ON THE CLEAN DATASETS

Fig. 4 highlights the accuracy achieved on three clean datasets, using the different classification strategies. With 1SFC, we reach 99.35%, 99.72%, and 96.03% accuracy on GTSRB, BelgiumTSC, and DITS, respectively. The usage of 2SFC improved the classification accuracy to 100%, 99.88%, and 99.48% on GTSRB, BelgiumTSC, and DITS, respectively. Classification of traffic signs based on sliding windows approach achieves an accuracy of 100% across GTSRB and DITS for W1C, W2C, and W3C, while 100% accuracy is achieved on BelgiumTSC using W1C and W2C.

W1C, W2C, or W3C achieves 100% accuracy on the GTSRB dataset. On DITS, strategies W1C, W2C and W3C achieves 100% accuracy, with the exception of W1C with AN. On BelgiumTSC, we get 100% classification accuracy using AN (W1C) and AN + MN2 classifiers (W2C). Fig. 4 shows that 2SFC and 3SFC significantly reduce the misclassification across all datasets with respect to 1SFC, while W1C, W2C, or W3C achieves 100% classification accuracy across all three datasets. This proves and quantifies the efficacy of the sliding window approach for classification in nominal operating conditions (clean frames).

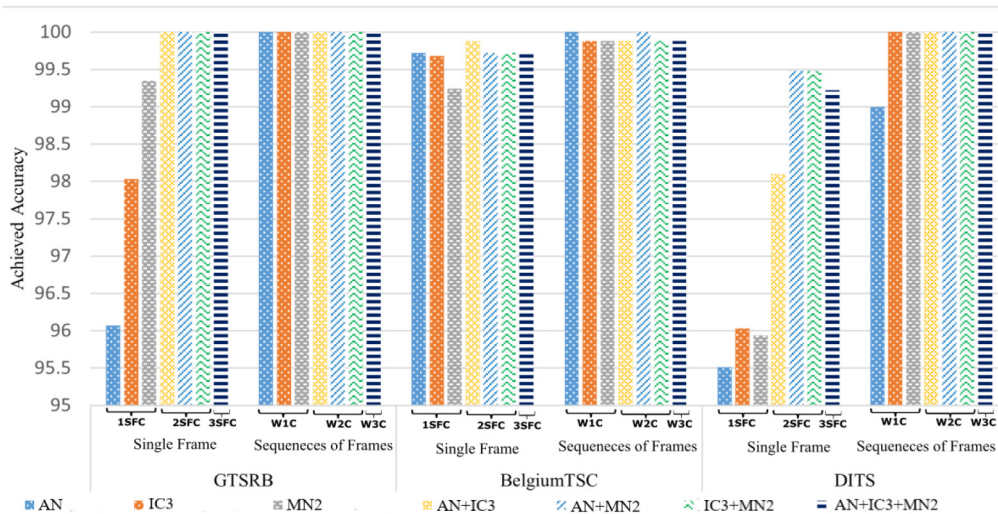


FIGURE 4. Highest accuracy achieved on the clean datasets GTSRB, BelgiumTSC, DITS.

TABLE 3. Minimum and maximum accuracy drop of 1SFC when it is exercised on the injected datasets and the clean datasets.

Original Dataset (Max Achieved Acc.)		GTSRB		DITS		BelgiumTSC	
		99.35%		96.03%		99.72%	
Fault Type	Config.	Accuracy drop	Base Classifier	Accuracy drop	Base Classifier	Accuracy drop	Base Classifier
Banding	3	[-3.09, -4.69]	AN	[-0.86, -5.87]	IC3	[-0.12, -0.60]	AN
Blurred	11	[-1.00, -4.76]	MN2	[-12.34, -18.03]	IC3	[-2.75, -13.49]	AN
Brightness	8	[-3.41, -40.15]	AN	[-1.90, -60.14]	AN	[-0.16, -67.30]	AN
Broken Lens	15	[-15.72, -52.48]	AN	[-10.78, -35.37]	AN	[-1.60, -53.17]	AN
Chromatic Aberration	1	-0.02	MN2	-1.12	IC3	-0.08	IC3
Condensation	3	[-0.06, -1.91]	MN2	[-0.52, -2.85]	IC3	[-0.20, -0.92]	AN
Dead Pixels	6	[-0.01, -11.82]	MN2	[-0.78, -11.13]	MN2	[-0.20, -6.03]	AN
Dirt	36	[0.05, -2.31]	MN2	[-0.09, -11.56]	IC3	[0.00, -1.92]	IC3
No Bayer Filter	1	-13.22	MN2	-14.84	IC3	-39.92	MN2
Ice	4	[-3.32, -26.45]	AN	[-2.24, -59.88]	AN	[-0.88, -39.40]	AN
Demosaicing	1	-3.08	AN	-18.64	IC3	-2.67	IC3
Noise	9	[-3.42, -57.40]	AN	[-5.69, -64.36]	AN	[-0.20, -76.97]	AN
Rain	5	[-3.53, 12.98]	AN	[-2.16, -17.00]	AN	[-0.56, -13.33]	AN

B. 1SFC ON THE INJECTED DATASETS

We analyze the accuracy drop obtained by 1SFC on the clean datasets and the injected datasets (Table 3). The 1SFC that obtains the best accuracy is reported: for such classifier, we report the maximum and minimum difference in accuracy (accuracy drop) with respect to the case on the clean datasets.

Experimental results show that there is a significant rise in misclassifications. For example, with banding, under different configurations the maximum and minimum accuracy drop on GTSRB is [-3.09, -4.69], where AN performs better than MN2 and IC3.

Experimental results indicate that 1SFC MN2 on GTSRB, and IC3 on DITS and BelgiumTSC, are overall robust against chromatic aberration failure; all the other failures have a more relevant impact on the accuracy of 1SFC. The accuracy drop is more relevant for some failures, which are highlighted in bold in Table 3; these are broken lens, ice, brightness, and noise. In particular, there are specific cases

of brightness failure that lead to a significant accuracy drop. For example, when brightness is very low (frames are very dark) or very high (frames are almost white), even humans may not recognize traffic signs. Another important observation is that for brightness, broken lens, ice, noise, or rain failures, AN always achieves the best accuracy. IC3 performs worse compared to the other 1SFC, i.e., AN and MN2 on GTSRB. Overall, MN2 and AN perform better on GTSRB, while AN and IC3 performance are better on the other two datasets (but still, the performance of 1SFC is far from perfect).

C. 2SFC AND 3SFC ON THE INJECTED DATASETS

Table 4 discusses the accuracy drop achieved by 2SFC and 3SFC on the clean dataset and the injected datasets. Experimental results show that 2SFC and 3SFC reduce misclassifications against each failure except few cases, where accuracy is less than in the 1SFC cases. These last

TABLE 4. Minimum and maximum accuracy drop of 2SFC and 3SFC on the injected datasets with respect to the clean datasets.

Original Dataset (Max Achieved Acc.)	GTSRB			DITS			BelgiumTSC		
	100%			100%			100%		
Fault Type (Config.)	Accuracy drop	Base Classifiers	Meta Classifier	Accuracy drop	Base Classifiers	Meta Classifier	Accuracy drop	Base Classifiers	Meta Classifier
Banding (3)	[-0.23, -2.85]	AN+MN2	KNN, DT	[-1.29, -4.31]	AN+IC3+MN2	KNN	[-0.04, -0.56]	AN+IC3	KNN
Blurred (11)	[-1.52, -5.92]	AN+IC3+MN2	LDA	[-7.76, -28.47]	MN2+IC3	RF	[-1.60, -14.57]	AN+IC3	KNN
Brightness (8)	[-0.10, -35.69]	AN+MN2	ABM2, RF	[-1.64, -63.33]	AN+IC3+MN2	KNN	[-0.28, -66.67]	AN+IC3+MN2	RF, KNN, ABM2
Broken Lens (15)	[-2.20, -46.17]	AN+IC3+MN2	RF, KNN	[-4.31, -28.04]	AN+IC3+MN2	KNN	[-2.24, -51.34]	AN+IC3	KNN, RF
Chromatic Aberration (1)	-0.08	AN+IC3+MN2	ABM2	-0.69	AN+IC3+MN2	KNN	-0.08	AN+IC3	KNN
Condensation (3)	[-0.10, -0.36]	AN+IC3+MN2	ABM2	[-0.86, -3.10]	AN+IC3+MN2	KNN	[-0.12, -1.44]	AN+IC3	KNN
Dead Pixels (6)	[-0.07, -1.99]	AN+MN2	ABM2, RF	[-2.41, -10.52]	AN+MN2	RF, KNN	[-0.36, -5.91]	AN+MN2	ABM2, KNN
Dirt (36)	[-0.02, -0.73]	AN+IC3+MN2	ABM2, RF	[-1.46, -9.75]	AN+IC3	KNN	[-0.04, -2.43]	AN+IC3	KNN, RF
No Bayer Filter (1)	-7.34	AN+IC3+MN2	RF	-11.13	MN2+IC3	KNN	-38.76	AN+MN2	KNN
Ice (4)	[-0.21, -22.43]	AN+MN2	ABM2, KNN	[-0.69, -59.88]	AN+IC3+MN2	KNN	[-0.68, -44.55]	AN+IC3	KNN
Demosaicing (1)	-1.01	AN+IC3+MN2	KNN	-18.03	MN2+IC3	KNN	-1.36	MN2+IC3	KNN
Noise (9)	[-0.96, -52.30]	AN+IC3+MN2	RF, DT	[-7.33, -67.64]	AN+IC3	KNN, RF	[-0.60, -75.45]	AN+MN2	KNN
Rain (5)	[-0.10, -4.80]	AN+MN2	ABM2, RF	[-0.43, -15.10]	AN+IC3+MN2	KNN	[-0.20, -20.96]	AN+IC3	KNN

TABLE 5. Minimum and maximum accuracy drop of W1C on the injected datasets with respect to the original datasets.

Original Dataset (Max Achieved Acc.)	GTSRB			DITS			BelgiumTSC		
	100%			100%			100%		
Fault Type (Config.)	Accuracy drop	Base Classifier	Meta Classifier	Accuracy drop	Base Classifier	Meta Classifier	Accuracy drop	Base Classifier	Meta Classifier
Banding (3)	[-0.24, -0.72]	AN	KNN, RF	-1.00	IC3	KNN, SVM, RF, DT, ABM2	[0.00, -0.36]	AN	KNN
Blurred (11)	0.00	MN2	KNN, LDA, SVM, ABM2	[-8.00, -21.00]	AN	KNN, DT, LDA	[-1.56, -12.10]	AN	KNN
Brightness (8)	[0.00, -6.93]	AN	KNN, DT, ABM2, RF	[-1.00, -51.00]	AN	SVM, LDA	[0.00, -62.04]	AN	KNN
Broken Lens (15)	[-1.20, -11.22]	AN	DT, LDA	[-8.00, -21.96]	AN	SVM, LDA, ABM2	[-0.72, -46.95]	AN	KNN
Chromatic Aberration (1)	0.00	AN	KNN, DT, ABM2, RF	0.00	IC3	KNN	0.00	AN	RF
Condensation (3)	0.00	AN	KNN, DT, ABM2, RF	[0.00, -2.00]	IC3	KNN, ABM2	[-0.12, -0.60]	AN	KNN
Dead Pixels (6)	0.00	AN	KNN, DT, ABM2, RF	[0.00, -7.00]	MN2	KNN	[0.00, -3.24]	AN	KNN
Dirt (36)	0.00	AN	KNN, DT, ABM2, RF	[0.00, -5.00]	IC3	KNN, SVM, ABM2	[0.00, -0.96]	IC3	KNN, RF
No Bayer Filter (1)	-0.72	MN2	KNN, ABM2, DT	-3.00	IC3	KNN	-35.33	MN2	RF
Ice (4)	[0.00, -2.39]	AN	KNN, DT, ABM2, RF	[-1.00, -40.66]	AN	KNN, SVM, RF, DT, ABM2	[-0.36, -37.49]	AN	KNN
Demosaicing (1)	0.00	AN	LDA	-3.00	IC3	KNN	-1.80	IC3	KNN
Noise (9)	[0.00, -37.71]	AN	KNN, ABM2, RF, DT	[-4.00, -60.00]	AN	KNN, SVM, DT	[-0.12, -77.25]	AN	KNN
Rain (5)	[0.00, -0.24]	AN	KNN, ABM2, RF, DT	[-1.00, -12.00]	AN	SVM, ABM2	[-0.24, -11.38]	AN	KNN

ones are in bold in Table 4, and they are brightness, broken lens, noise, and ice failures.

With respect to 1SFC and 2SFC, on GTSRB and DITS, 3SFC achieved better accuracy on the majority of failures. Instead, on BelgiumTSC, the 2SFC AN + IC3 is the most robust against the majority of failures. Overall, utilizing 2SFC and 3SFC improves the accuracy. The meta-level classifier KNN performs better on all three datasets.

However, the effect of failures is still not fully mitigated with the 2SFC and 3SFC: there is no case that achieves 100% accuracy against any failure, and in some cases the

combination of classifiers is worse than using just a single classifier.

D. W1C ON THE INJECTED DATASETS

Table 5 reports the accuracy drop achieved by W1C on the clean datasets versus the injected datasets. Classification performed on subsequent frames within a sliding window of size 3 improves the classification performance with respect to 1SFC. For example, W1C achieved 100% accuracy for all three datasets under the chromatic aberration failure, where the accuracy difference is zero; this is marked in bold in Table 5. Furthermore, on the GTSRB

TABLE 6. Minimum and maximum accuracy drop of W2C and W3C on the injected datasets with respect to the clean datasets.

Original Dataset (Max Achieved Acc.)	GTSRB			DITS			BelgiumTSC		
	100%			100%			100%		
Fault Type (Config.)	Accuracy drop	Base Classifiers	Meta Classifier	Accuracy drop	Base Classifier	Meta Classifier	Accuracy drop	Base Classifier	Meta Classifier
Banding (3)	[0.00, -0.72]	AN+IC3	KNN, RF, DT	[0.00, -2.00]	AN+IC3+MN2	KNN, SVM, ABM2	[-0.48, -0.44]	AN+MN2	DT
Blurred (11)				[-8.14, 20.00]	AN+IC3+MN2	KNN, DT	[-0.72, -11.02]	AN+IC3	KNN
Brightness (8)	[0.00, -6.45]	AN+MN2	KNN, LDA, SVM	[0.00, -50.00]	AN+MN2	RF, KNN	[0.00, -65.15]	AN+IC3	KNN
Broken Lens (15)	[0.00, -14.32]	AN+IC3+MN2	LDA, KNN	[-4.00, -21.00]	AN+IC3	KNN, ABM2	[-0.84, -49.47]	AN+IC3	KNN, RF
Chromatic Aberration (1)									
Condensation (3)				0.00	MN2+IC3	KNN, ABM2	[0.00, -0.36]	AN+IC3	KNN
Dead Pixels (6)				[-1.00, -7.00]	AN+MN2	KNN, LDA, ABM2	[-0.24, -3.72]	AN+MN2	KNN
Dirt (36)				[-1.00, -4.00]	AN+IC3	KNN	[0.00, -1.80]	AN+IC3+MN2	KNN, SVM
No Bayer Filter (1)	-0.48	AN+IC3+MN2	ABM2	-4.00	MN+IC3	KNN	-33.78	AN+IC3+MN2	ABM2
Ice (4)	[0.00, -2.15]	AN+IC3	KNN, SVM, DT, RF, ABM2	[0.00, -37.00]	AN+MN2	KNN	[-0.12, -40.96]	AN+IC3	KNN
No Demosaicing (1)				-2.00	AN+IC3	KNN	-0.72	MN2+IC3	KNN
Noise (9)	[0.00, -27.21]	AN+IC3	KNN, SVM, ABM2, DT	[-3.00, -57.00]	AN+IC3	KNN, SVM	[-0.36, -74.02]	AN+MN2	KNN, DT
Rain (5)	[0.00, -0.72]	AN+IC3+MN2	KNN, SVM, DT, RF, ABM2	[0.00, -12.00]	AN+MN2	KNN, ABM2	[-0.12, -13.18]	AN+MN2	KNN, DT

dataset we achieved 100% accuracy against blurred, chromatic aberration, condensation, dead pixels, dirt and no demosaicing.

Another important observation is that, for the majority of failures, W1C (with AN as base classifier and KNN meta-level classifier) achieves the highest accuracy. W1C improves accuracy with respect to 1SFC, 2SFC, and 3SFC, with few exceptions, such as broken lens failure on DITS and no demosaicing failure on BelgiumTSC. The reason behind the inferior performance of W1C in this case is most likely due to a frame which is misclassified by the single classifier: 2SFC and 3SFC may classify the frame correctly, because there is more than one base classifier executing in parallel.

Similarly, to the previous cases, there is no significant improvement against some failures such as broken lens, brightness, ice, and noise failures.

E. W2C AND W3C ON THE INJECTED DATASETS

Table 6 highlights the degradation intervals of accuracy achieved by W2C and W3C on the clean dataset versus the injected datasets.

Accuracy against failures blurred, chromatic aberration, condensation, dead pixels, dirt, and no demosaicing is 100% as already achieved for W1C, and it is not further reported in Table 6. The bold values in Table 6 represent the failures where the performance of W2C and W3C improves with respect to W1C. For the remaining failures, W1C performs better. The possible reason is that a frame is classified correctly by a base classifier, but when the same frame is classified by multiple base classifiers, there are chances that some misclassify the frame. This way, when the various meta-features for W2C and W3C are provided to meta-level classifiers, this may result in a misclassification.

Overall, few failures such as ice, broken lens, brightness, and noise are still very hard to be classified accurately. Overall, the meta-level classifier KNN is the one that performs better in the majority of cases.

VI. EXPLANATION OF DNNS ROBUSTNESS

The LIME [29] tool aims to explain the predictions of a DNN. It shows how different models select the frame's influential features that contribute the most to the final classification. We explore some frames with the LIME toolbox to investigate the reason behind the prediction of the models, especially to understand the difference identified in our experiments between AN, IC3, and MN2.

Fig. 5 shows the LIME output for various frames that are classified by the three DNNs. Coloring shows which areas of the frame contribute the most to the classification. The LIME output shows that AN selects stronger features compared to the other classifiers: this is particularly evident in the second and fourth rows. The LIME output of IC3 shows that its strongest features are mostly congested in an area. Instead, AN decides using features that are scattered through the frame. This is most likely the reason IC3 is the less robust in our experiments: when a failure is injected in a frame, as for example a scratch of the broken lens failure, it may overlap the areas that are relevant for feature classification. As shown in Fig. 5, for the blue circular traffic sign, AN highlight many features, and the strongest features are located in various parts of the traffic sign; on the other hand, IC3 and MN2 have very few strong features and mostly in adjacent positions.

Summarizing, the main hurdle for classifiers robustness against different camera failures may depend on the DNNs: if the frame alteration due to failure is on the frame portion which plays the key role in classification, that failure

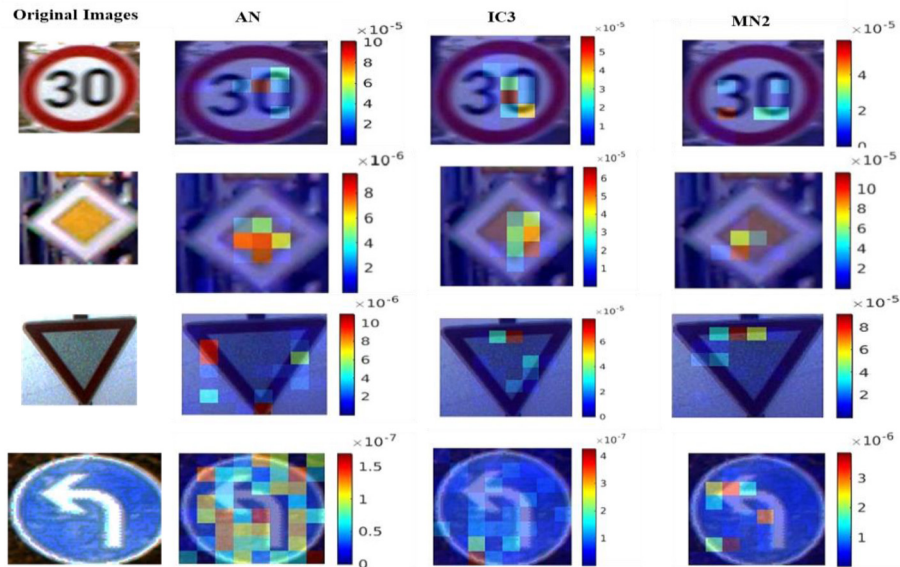


FIGURE 5. Explanation of AN, IC3, and MN2 models through LIME, highlighting the influential features of selected traffic sign frames. (best viewed in color).

will misguide the TSR to output an incorrect label for the traffic sign.

VII. CONCLUDING REMARKS

To conclude this paper, we summarize in this section the findings and lessons learned, and we discuss on increasing the robustness of TSR and future directions.

A. LESSONS LEARNED

This study considers different strategies for TSR, i.e., single-frame and sliding window, to understand their robustness against camera failures. As reference, we use the datasets GTSRB, DITS, and BelgiumTSC, where no single-frame classifier achieves 100% accuracy. Instead, stacking meta-level classifiers that work on sliding windows (W1C, W2C, W3C) significantly improve the accuracy of TSR: they achieve 100% accuracy on all three datasets.

Concerning instead the presence of camera failures, we observed that the implemented failures generally lead to a drop in accuracy. For example, Fig. 6 highlights the highest accuracy achieved against each camera failure on the three datasets separately, i.e., GTSRB, DITS, and BelgiumTSC, considering the failures configurations which has the highest accuracy drop.

However, we also noticed that some TSR strategies are significantly more robust than others and are able to tolerate the majority of failures. While there is no individual TSR strategy that performs better than the other either against each failure or on all three datasets, the takeaway message of this study is that meta-level classifiers in conjunction with sliding windows of subsequent frames improve the TSR performance and robustness. More concretely, we report the following considerations on robustness:

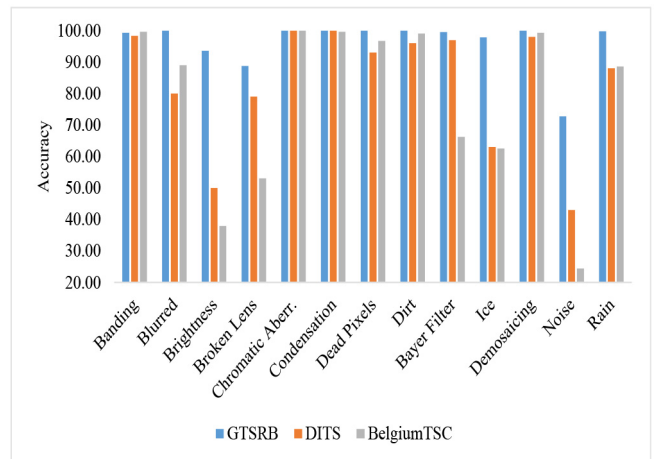


FIGURE 6. Accuracy achieved for each failure on the three datasets, when we consider the failure configurations which lead to the highest accuracy drop.

- The 1SFC strategy proved insufficiently robust against camera failures, with a severe degradation of accuracy with respect to the clean datasets.
- Instead, 2SFC and 3SFC strategies are more robust, and present a lower performance degradation; the most critical cases are highlighted in bold in Table 4.
- One important observation is that IC3 never achieves the highest accuracy on GTSRB in presence of camera failures. This is motivated through Explainable AI in Section VI.
- Sliding windows strategies W1C, W2C, and W3C improve the robustness of TSR with respect to 1SFC, 2SFC, and 3SFC as shown in Table 5. Especially, we achieve 100% accuracy against 6 failures on the GTSRB dataset, while for DITS and BelgiumTSC 100%

accuracy is achieved for chromatic aberration failure only. DITS achieved 100% accuracy for condensation failure using W2C (MN2 + IC3).

- Further, some failures such as noise, ice, brightness, broken lens, and no bayer filter have a bad impact on the robustness of the TSR.

B. INCREASING THE ROBUSTNESS OF TSR

Our experimental results indicate that camera failures degrade the TSR performance. Despite the different classification strategies, we are unable to minimize the drop in accuracy under all conditions and datasets, exceptions made for chromatic aberration. This brings the necessity of possible strategies to overcome the effect of camera failures on the performance of a TSR classifier. This constitute our ongoing work, and it can be organized in two different strategies.

First, to increase the robustness of classifiers against different camera failures, we can use data augmentation [68] in which the failed frames are added to the training dataset to make the classifier more robust.

Second, a detector can analyze each frame to understand if the frame is of acceptable quality or is degraded. If the frame contains defects, an alarm can be raised, and the accuracy of the prediction can be suspected at system level. The detector can be created by training deep classifiers on the injected datasets.

C. CONCLUSION

The objective of this study is to analyze the effect of camera failures on the robustness of different Traffic Sign Recognition (TSR) strategies. This study performs extensive experiments on three traffic sign datasets that are perturbed with 13 camera failures. Different classifiers are proposed, that are composed either to process a single frame or a sliding window of frames. We observe that approaches based on sliding windows perform better compared to approaches based on single frames, both on the clean datasets, and when failures are considered.

Furthermore, some failures impacted the TSR performance severely so that several misclassifications are measured. Overall, there is no TSR strategy that is more robust than the others on all datasets and camera failures.

REFERENCES

- [1] K. Bayouh, F. Hamdaoui, and A. Mtibaa, "Transfer learning based hybrid 2D–3D CNN for traffic sign recognition and semantic road detection applied in advanced driver assistance systems," *Appl. Intell.*, vol. 51, no. 1, pp. 124–142, Jan. 2021, doi: [10.1007/s10489-020-01801-5](https://doi.org/10.1007/s10489-020-01801-5).
- [2] W. Farag, "Traffic signs classification by deep learning for advanced driving assistance systems," *Intell. Decis. Technol.*, vol. 13, no. 3, pp. 305–314, Jan. 2019, doi: [10.3233/IDT-180064](https://doi.org/10.3233/IDT-180064).
- [3] A. Gudigar, S. Chokkadi, and U. Raghavendra, "A review on automatic detection and recognition of traffic sign," *Multimedia Tools Appl.*, vol. 75, no. 1, pp. 333–364, Jan. 2016, doi: [10.1007/s11042-014-2293-7](https://doi.org/10.1007/s11042-014-2293-7).
- [4] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German traffic sign detection benchmark," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Aug. 2013, pp. 1–8, doi: [10.1109/IJCNN.2013.6706807](https://doi.org/10.1109/IJCNN.2013.6706807).
- [5] Á. Arcos-García, J. A. Álvarez-García, and L. M. Soria-Morillo, "Evaluation of deep neural networks for traffic sign detection systems," *Neurocomputing*, vol. 316, pp. 332–344, Nov. 2018, doi: [10.1016/j.neucom.2018.08.009](https://doi.org/10.1016/j.neucom.2018.08.009).
- [6] M. Mathias, R. Timofte, R. Benenson, and L. Van Gool, "Traffic sign recognition—How far are we from the solution?" in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Aug. 2013, pp. 1–8, doi: [10.1109/IJCNN.2013.6707049](https://doi.org/10.1109/IJCNN.2013.6707049).
- [7] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2011, pp. 2809–2813, doi: [10.1109/IJCNN.2011.6033589](https://doi.org/10.1109/IJCNN.2011.6033589).
- [8] A. Mogelmoose, M. M. Trivedi, and T. B. Moeslund, "Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1484–1497, Dec. 2012, doi: [10.1109/TITS.2012.2209421](https://doi.org/10.1109/TITS.2012.2209421).
- [9] S. Maldonado-Bascon, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno, and F. Lopez-Ferreras, "Road-sign detection and recognition based on support vector machines," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 264–278, Jun. 2007, doi: [10.1109/TITS.2007.895311](https://doi.org/10.1109/TITS.2007.895311).
- [10] J. Zhang, Z. Xie, J. Sun, X. Zou, and J. Wang, "A cascaded R-CNN with multiscale attention and imbalanced samples for traffic sign detection," *IEEE Access*, vol. 8, pp. 29742–29754, 2020, doi: [10.1109/ACCESS.2020.2972338](https://doi.org/10.1109/ACCESS.2020.2972338).
- [11] H. S. Lee and K. Kim, "Simultaneous traffic sign detection and boundary estimation using convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 5, pp. 1652–1663, May 2018, doi: [10.1109/TITS.2018.2801560](https://doi.org/10.1109/TITS.2018.2801560).
- [12] Y. Tian, J. Gelernter, X. Wang, J. Li, and Y. Yu, "Traffic sign detection using a multi-scale recurrent attention network," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 12, pp. 4466–4475, Dec. 2019, doi: [10.1109/TITS.2018.2886283](https://doi.org/10.1109/TITS.2018.2886283).
- [13] J. Levinson et al., "Towards fully autonomous driving: Systems and algorithms," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Jun. 2011, pp. 163–168, doi: [10.1109/IVS.2011.5940562](https://doi.org/10.1109/IVS.2011.5940562).
- [14] S. Hurtado and S. Chiasson, "An eye-tracking evaluation of driver distraction and unfamiliar road signs," in *Proc. 8th Int. Conf. Autom. User Interfaces Interact. Veh. Appl.*, New York, NY, USA, Oct. 2016, pp. 153–160, doi: [10.1145/3003715.3005407](https://doi.org/10.1145/3003715.3005407).
- [15] M. Atif, T. Zoppi, M. Gharib, and A. Bondavalli, "Quantitative comparison of supervised algorithms and feature sets for traffic sign recognition," in *Proc. 36th Annu. ACM Symp. Appl. Comput.*, 2021, pp. 174–177, Accessed: Feb. 8, 2022. [Online]. Available: <https://doi.org/10.1145/3412841.3442072>
- [16] J. Li and Z. Wang, "Real-time traffic sign recognition based on efficient CNNs in the wild," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 3, pp. 975–984, Mar. 2019, doi: [10.1109/TITS.2018.2843815](https://doi.org/10.1109/TITS.2018.2843815).
- [17] M. Atif, T. Zoppi, M. Gharib, and A. Bondavalli, "Towards enhancing traffic sign recognition through sliding windows," *Sensors*, vol. 22, no. 7, p. 2683, 2022.
- [18] F. Secci and A. Ceccarelli, "RGB Cameras Failures and Their Effects in Autonomous Driving Applications," Mar. 2022. Accessed: Mar. 9, 2022. [Online]. Available: <http://arxiv.org/abs/2008.05938>
- [19] J. H. Chung, D. W. Kim, T. K. Kang, and M. T. Lim, "Traffic sign recognition in harsh environment using attention based convolutional pooling neural network," *Neural Process. Lett.*, vol. 51, no. 3, pp. 2551–2573, Jun. 2020, doi: [10.1007/s11063-020-10211-0](https://doi.org/10.1007/s11063-020-10211-0).
- [20] C. Collet and O. Musicant, "Associating vehicles automation with drivers functional state assessment systems: A challenge for road safety in the future," *Front. Human Neurosci.*, vol. 13, p. 131, Apr. 2019, Accessed: Feb. 16, 2022. [Online]. Available: <https://www.frontiersin.org/article/10.3389/fnhum.2019.00131>
- [21] R. S. Ferreira, J. Arlat, J. Guiochet, and H. Waeselync, "Benchmarking safety monitors for image classifiers with machine learning," in *Proc. IEEE 26th Pac. Rim Int. Symp. Depend. Comput. (PRDC)*, Dec. 2021, pp. 7–16, doi: [10.1109/PRDC53464.2021.00012](https://doi.org/10.1109/PRDC53464.2021.00012).
- [22] P. Brazdil, C. G. Carrier, C. Soares, and R. Vilalta, *Metalearning: Applications to Data Mining*. Berlin, Germany: Springer, 2008.
- [23] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German traffic sign recognition benchmark: A multi-class classification competition," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2011, pp. 1453–1460, doi: [10.1109/IJCNN.2011.6033395](https://doi.org/10.1109/IJCNN.2011.6033395).

- [24] A. Youssef, D. Albani, D. Nardi, and D. D. Bloisi, "Fast traffic sign recognition using color segmentation and deep convolutional networks," in *Proc. Adv. Concepts Intell. Vis. Syst.*, Oct. 2016, pp. 205–216, doi: [10.1007/978-3-319-48680-2_19](https://doi.org/10.1007/978-3-319-48680-2_19).
- [25] R. Timofte, K. Zimmermann, and L. Van Gool, "Multi-view traffic sign detection, recognition, and 3D localisation," *Mach. Vis. Appl.*, vol. 25, no. 3, pp. 633–647, Apr. 2014, doi: [10.1007/s00138-011-0391-3](https://doi.org/10.1007/s00138-011-0391-3).
- [26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1–9. Accessed: Feb. 8, 2022. [Online]. Available: <https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html>
- [27] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision." 2016. Accessed: Feb. 8, 2022. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Szegedy_Rethinking_the_Inception_CVPR_2016_paper.html
- [28] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks." 2018. Accessed: Feb. 8, 2022. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2018/html/Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.html
- [29] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, New York, NY, USA, Aug. 2016, pp. 1135–1144, doi: [10.1145/2939672.2939778](https://doi.org/10.1145/2939672.2939778).
- [30] A. Mameri, A. Boukerche, and M. Almulla, "Design of traffic sign detection, recognition, and transmission systems for smart vehicles," *IEEE Wireless Commun.*, vol. 20, no. 6, pp. 36–43, Dec. 2013, doi: [10.1109/MWC.2013.6704472](https://doi.org/10.1109/MWC.2013.6704472).
- [31] U. K. D. N. Manisha and S. R. Liyanage, "An online traffic sign recognition system for intelligent driver assistance," in *Proc. 17th Int. Conf. Adv. ICT Emerg. Regions (ICTer)*, Sep. 2017, pp. 1–6, doi: [10.1109/ICTER.2017.8257815](https://doi.org/10.1109/ICTER.2017.8257815).
- [32] I. Matoš, Z. Krpić, and K. Romić, "The speed limit road signs recognition using hough transformation and multi-class SVM," in *Proc. Int. Conf. Syst. Signals Image Process. (IWSSIP)*, Jun. 2019, pp. 89–94, doi: [10.1109/IWSSIP.2019.8787249](https://doi.org/10.1109/IWSSIP.2019.8787249).
- [33] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Trans. Neural Netw.*, vol. 13, no. 2, pp. 415–425, Mar. 2002, doi: [10.1109/72.991427](https://doi.org/10.1109/72.991427).
- [34] Y. Freund, "A More Robust Boosting Algorithm." May 2009. Accessed: Feb. 8, 2022. [Online]. Available: <http://arxiv.org/abs/0905.2138>
- [35] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: [10.1023/A:1010933404324](https://doi.org/10.1023/A:1010933404324).
- [36] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "KNN model-based approach in classification," in *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, Heidelberg, Germany: Springer, Nov. 2003, pp. 986–996, doi: [10.1007/978-3-540-39964-3_62](https://doi.org/10.1007/978-3-540-39964-3_62).
- [37] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, Jun. 2005, pp. 886–893, doi: [10.1109/CVPR.2005.177](https://doi.org/10.1109/CVPR.2005.177).
- [38] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002, doi: [10.1109/TPAMI.2002.1017623](https://doi.org/10.1109/TPAMI.2002.1017623).
- [39] Wahyono and K.-H. Jo, "A comparative study of classification methods for traffic signs recognition," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Feb. 2014, pp. 614–619, doi: [10.1109/ICIT.2014.6895001](https://doi.org/10.1109/ICIT.2014.6895001).
- [40] I. C. Schusztzer, "A comparative study of machine learning methods for traffic sign recognition," in *Proc. 19th Int. Symp. Symbol. Numer. Algorithms Sci. Comput. (SYNASC)*, Sep. 2017, pp. 389–392, doi: [10.1109/SYNASC.2017.00070](https://doi.org/10.1109/SYNASC.2017.00070).
- [41] X. Yang, Y. Qu, and S. Fang, "Color fused multiple features for traffic sign recognition," in *Proc. 4th Int. Conf. Internet Multimedia Comput. Service*, New York, NY, USA, Sep. 2012, pp. 84–87, doi: [10.1145/2382336.2382360](https://doi.org/10.1145/2382336.2382360).
- [42] S. Agrawal and R. K. Chaurasiya, "Ensemble of SVM for accurate traffic sign detection and recognition," in *Proc. Int. Conf. Graph. Signal Process.*, New York, NY, USA, Jun. 2017, pp. 10–15, doi: [10.1145/3121360.3121373](https://doi.org/10.1145/3121360.3121373).
- [43] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Netw.*, vol. 32, pp. 323–332, Aug. 2012, doi: [10.1016/j.neunet.2012.02.016](https://doi.org/10.1016/j.neunet.2012.02.016).
- [44] O. T. Nartey, G. Yang, S. K. Asare, J. Wu, and L. N. Frempong, "Robust semi-supervised traffic sign recognition via self-training and weakly-supervised learning," *Sensors*, vol. 20, no. 9, p. 9, Jan. 2020, doi: [10.3390/s20092684](https://doi.org/10.3390/s20092684).
- [45] C. Szegedy et al. "Going Deeper With Convolutions." 2015. Accessed: Feb. 8, 2022. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Szegedy_Going_Deeper_With_2015_CVPR_paper.html
- [46] C. Lin, L. Li, W. Luo, K. C. P. Wang, and J. Guo, "Transfer learning based traffic sign recognition using inception-v3 model," *Period. Polytech. Transp. Eng.*, vol. 47, no. 3, 2019, Art. no. 3, doi: [10.3311/PPtr.11480](https://doi.org/10.3311/PPtr.11480).
- [47] Z. Bi, L. Yu, H. Gao, P. Zhou, and H. Yao, "Improved VGG model-based efficient traffic sign recognition for safe driving in 5G scenarios," *Int. J. Mach. Learn. Cybern.*, vol. 12, no. 11, pp. 3069–3080, Nov. 2021, doi: [10.1007/s13042-020-01185-5](https://doi.org/10.1007/s13042-020-01185-5).
- [48] B. Akgul, B. Haznedar, M. Hasoğlu, and K. Bayram, "Development of a real-time traffic sign recognition system based on deep learning approach with convolutional neural networks and integrating to the embedded systems," *Euroasia J. Math. Eng. Nat. Med. Sci.*, vol. 8, no. 14, p. 19, 2021, doi: [10.38065/euroasiaorg.481](https://doi.org/10.38065/euroasiaorg.481).
- [49] A. Vennelakanti, S. Shreya, R. Rajendran, D. Sarkar, D. Muddegowda, and P. Hanagal, "Traffic sign detection and recognition using a cnn ensemble," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2019, pp. 1–4, doi: [10.1109/ICCE.2019.8662019](https://doi.org/10.1109/ICCE.2019.8662019).
- [50] S. Palavanчу, "Transfer learning models for traffic sign recognition system," *Ann. Romanian Soc. Cell Biol.*, vol. 25, no. 2, pp. 3477–3489, Mar. 2021.
- [51] A. Jose, H. Thodupunoori, and B. B. Nair, "A novel traffic sign recognition system combining viola—Jones framework and deep learning," *Soft Comput. Signal Process.*, vol. 18, pp. 507–517, Jan. 2019, doi: [10.1007/978-981-13-3600-3_48](https://doi.org/10.1007/978-981-13-3600-3_48).
- [52] S. Abizada, "Traffic sign recognition using histogram of oriented gradients and convolutional neural networks," in *Proc. 11th World Conf. Intell. Syst. Ind. Autom. (WCIS)*, Oct. 2020, pp. 452–459, doi: [10.1007/978-3-030-68004-6_59](https://doi.org/10.1007/978-3-030-68004-6_59).
- [53] A. Hechri and M. Abdellatif, "Two-stage traffic sign detection and recognition based on SVM and convolutional neural networks," *IET Image Process.*, vol. 14, no. 5, pp. 939–946, 2020.
- [54] Y. Yuan, Z. Xiong, and Q. Wang, "An incremental framework for video-based traffic sign detection, tracking, and recognition," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 7, pp. 1918–1929, Jul. 2017, doi: [10.1109/ITITS.2016.2614548](https://doi.org/10.1109/ITITS.2016.2614548).
- [55] H. Luo, Y. Yang, B. Tong, F. Wu, and B. Fan, "Traffic sign recognition using a multi-task convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 4, pp. 1100–1111, Apr. 2018, doi: [10.1109/ITITS.2017.2714691](https://doi.org/10.1109/ITITS.2017.2714691).
- [56] C. Premebida, G. Melotti, and A. Asvadi, "RGB-D object classification for autonomous driving perception," in *RGB-D Image Analysis and Processing (Advances in Computer Vision and Pattern Recognition)*, P. Rosin, Y. K. Lai, L. Shao, and Y. Liu, Eds. Cham, Switzerland: Springer, 2019. [Online]. Available: https://doi.org/10.1007/978-3-030-28603-3_17
- [57] B. H. B. Hashem and T. Ozeki, "Pedestrian detection by using FAST-HOG features," in *Proc. 3rd Int. Conf. Human Agent Interact.*, New York, NY, USA, Oct. 2015, pp. 277–278, doi: [10.1145/2814940.2814996](https://doi.org/10.1145/2814940.2814996).
- [58] A. Morozov, E. Valiev, M. Beyer, K. Ding, L. Gauerhof, and C. Schorn, "Bayesian model for trustworthiness analysis of deep learning classifiers," in *Proc. AISafety IJCAI*, 2020.
- [59] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255, doi: [10.1109/CVPR.2009.5206848](https://doi.org/10.1109/CVPR.2009.5206848).
- [60] J. G. Carbonell, R. S. Michalski, and T. M. Mitchell, "1—An overview of machine learning," in *Machine Learning*, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, Eds. San Francisco, CA, USA: Morgan Kaufmann, 1983, pp. 3–23, doi: [10.1016/B978-0-08-051054-5.50005-4](https://doi.org/10.1016/B978-0-08-051054-5.50005-4).

- [61] W.-Y. Loh and Y.-S. Shih, "SPLIT selection methods for classification trees," *Statistica Sinica*, vol. 7, no. 4, pp. 815–840, 1997.
- [62] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, no. 2, pp. 179–188, 1936, doi: [10.1111/j.1469-1809.1936.tb02137.x](https://doi.org/10.1111/j.1469-1809.1936.tb02137.x).
- [63] "Realistic Lens Blur/Chromatic Aberration Filter." Accessed: Mar. 5, 2022. [Online]. Available: <https://github.com/yoonsikp/kromo>
- [64] "ActionVFX." Accessed: Mar. 5, 2022. [Online]. Available: <https://www.actionvfx.com/>
- [65] D. Baumgartner, P. Roessler, W. Kubinger, C. Zinner, and K. Ambrosch, "Benchmarks of low-level vision algorithms for DSP, FPGA, and mobile PC processors," in *Embedded Computer Vision (Advances in Pattern Recognition)*, B. Kisačanin, S. S. Bhattacharyya, and S. Chai, Eds. London, U.K.: Springer, 2009. [Online]. Available: https://doi.org/10.1007/978-1-84800-304-0_5
- [66] E. Mortaz, "Imbalance accuracy metric for model selection in multi-class imbalance classification problems," *Knowl. Based Syst.*, vol. 210, Dec. 2020, Art. no. 106490, doi: [10.1016/j.knosys.2020.106490](https://doi.org/10.1016/j.knosys.2020.106490).
- [67] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, Jul. 2009, doi: [10.1016/j.ipm.2009.03.002](https://doi.org/10.1016/j.ipm.2009.03.002).
- [68] D. A. van Dyk and X.-L. Meng, "The art of data augmentation," *J. Comput. Graph. Stat.*, vol. 10, no. 1, pp. 1–50, Mar. 2001, doi: [10.1198/10618600152418584](https://doi.org/10.1198/10618600152418584).
- [69] C. I. Patel, S. Garg, T. Zaveri, A. Banerjee, and R. Patel, "Human action recognition using fusion of features for unconstrained video sequences," *Comput. Elect. Eng.*, vol. 70, pp. 284–301, Aug. 2018.
- [70] C. I. Patel, D. Labana, S. Pandya, K. Modi, H. Ghayvat, and M. Awais, "Histogram of oriented gradient-based fusion of features for human action recognition in action video sequences," *Sensors*, vol. 20, no. 24, p. 7299, 2020.

MUHAMMAD ATIF received the B.E. degree in information technology from the A.Q Khan Institute of Computer Science and Information Technology, KRL Kahuta, Pakistan, in 2014, and the M.S. degree in computer system engineering from the Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Pakistan, in 2016. He is currently pursuing the Ph.D. degree with the University of Florence, Florence, Italy.

From 2016 to 2019, he worked as a Lecturer with the Department of Computer Science, National University of Computer and Emerging Sciences, Pakistan. His research interests include data mining, computer vision, and computational intelligence.

ANDREA CECCARELLI is an Associate Professor of Computer Science with the University of Florence, Florence, Italy. His primary research interests are in the design, monitoring and experimental evaluation of dependable and secure systems, and systems-of-systems. His scientific activities originated more than 100 papers which appeared in international conferences, workshops, and journals. He has been the PC Co-Chair of the conferences SRDS and LADC. He is a member of the IFIP WG 10.4 on "Dependable Computing and Fault-Tolerance."

TOMMASO ZOPPI is a Research Associate with the University of Florence. He is involved in several European/National funded and even Industrial projects. His research focuses on anomaly detection, security and safety, often applying standards to plan, design, develop and implement appropriate architectures or software in the domain of critical systems. He currently serves as a member of the program committee of several international conferences.

MOHAMAD GHARIB received the Ph.D. degree from the University of Trento, Italy, in April 2015. He is a Lecturer of Software Engineering with the Institute of Computer Science, University of Tartu. He was a Postdoctoral Researcher with the University of Florence. Before that, he was a Postdoctoral Researcher with the Department of Information Engineering and Computer Science, University of Trento. His current research interests focus mainly on designing secure and safe cyber-physical system of systems and socio-technical systems.

ANDREA BONDAVALLI (Member, IEEE) is a Full Professor of Computer Science with the University of Florence. His research interests include the design and evaluation of resilient and secure systems and infrastructures. His scientific activities originated more than 220 papers appeared in international journals and conferences. He led various national and European projects and has been chairing the program committee in several International Conferences. He is a member of the IFIP W.G. 10.4 Working Group on "Dependable Computing and Fault-Tolerance."

Open Access funding provided by 'Università degli Studi di Firenze' within the CRUI CARE Agreement