Proceedings of the 33rd European Safety and Reliability Conference (ESREL 2023)
Edited by Mário P. Brito, Terje Aven, Piero Baraldi, Marko Čepin and Enrico Zio
 Published by Research Publishing, Singapore.
doi: 10.3850/978-981-18-8071-1_P206-cd

Multilevel Artificial Intelligence Classification of Faulty Image Data for Enhancing Sensor Reliability

Omar Mohammed

Chair of Reliability of Technical Systems and Electrical Measurement, University of Siegen, 57076 Siegen, Germany. E-mail: omar.mohammed@uni-siegen.de

Parthib Khound

Chair of Reliability of Technical Systems and Electrical Measurement, University of Siegen, 57076 Siegen, Germany. E-mail: parthib.khound@uni-siegen.de

Peng Su

Unit of Mechatronics and Embedded Control Systems, KTH Royal Institute of Technology, 100 44 Stockholm, Sweden. E-mail: pensu@kth.se

Dejiu Chen

Unit of Mechatronics and Embedded Control Systems, KTH Royal Institute of Technology, 100 44 Stockholm, Sweden. E-mail: chen@md.kth.se

Frank Gronwald

Chair of Reliability of Technical Systems and Electrical Measurement, University of Siegen, 57076 Siegen, Germany. E-mail: frank.gronwald@uni-siegen.de

A multi-stage classification algorithm is proposed to predict the fault type and its associated intensity level of a camera input frame to enhance the reliability of a camera-based system. A fault injecting tool is used to generate the dataset required for the training. The model architecture mainly comprises three convolutions neural network (CNN) layers and three fully connected layers. The model achieves 93.8% accuracy for predicting a fault type. For the fault intensity prediction the accuracy significantly varies for each fault type but for some faults, the model achieves a very good prediction accuracy. However, for some other faults the accuracy can be remarkably low. The primary reason for this gap is that the intensity levels of all considered faults can be described in a sufficiently quantitative way, i.e., there is no sufficient metric available so far.

Keywords: Fault classification, fault type, fault intensity, artificial intelligence, camera-based system reliability, smart camera sensor.

1. Introduction

Feedback signals from sensors are often used to perform necessary control actions for real time functional operation, e.g., adaptive cruise control (ACC) systems, lane keeping systems (LKS), etc. The quality of performance of these safety-critical systems highly relies on the measurement data fed back by the associated sensors. A faulty sensor data can not only undermine the stability but also drastically compromise the safety of the system.

On the system level, the classical fault-tolerant methods are quite popular in the literature, see Blanke et al. (2000) and Gao et al. (2015). These methods perform fault diagnosis for the execution of fail-safe actions, see Blanke et al. (2000). Normally, the fault diagnosis is performed by the processor of the system; thus, the computational load is at the control processing unit. Moreover, the fault tolerant control methods are computationally intensive. An intelligent monitoring method on the sensor level that can independently detect and classify the faults would

not only reduce the computational load of the system but also facilitate early detection and diagnosis. This is a general strategy that can be integrated into different functional control algorithms, irrespective of whether the control functions are developed by the classical control schemes or any modern artificial intelligence (AI) methods.

As part of this strategy, the classification of sensor fault types is actively discussed in the literature; see e.g., Baljak et al. (2013); Jan et al. (2017). Additionally, categorizing the associated intensity of each fault type would better facilitate the performance evaluation of a system. Therefore, we propose an extended classification concept that classifies both type and strength of a fault. The strength represents a defined intensity of a fault type. This proposed classification methodology is presented for a red, green, and blue (RGB) camera sensor. In Secci and Ceccarelli (2020), the failure mode and effects analysis are performed for different fault types in an RGB camera used for an autonomous driving application. Some examples for injected faults are blur, broken lens, dead pixels, etc. Definitively different fault types have certain effects on the functional operation. The strength of faults may also intensify the effects to different degrees. For example, a slightly broken lens may not considerably affect the lane-keeping performance of a vehicle, but if the lens is heavily cracked, the system could completely fail. There exist some studies where fault injection is applied to the AI networks to analyze the effects on the system; see Su and Chen (2022) and Liu et al. (2017). In this paper, the faults are injected into the input image in order to emulate the possible hardware and environmental faults on the camera sensor, which will later be used to formulate some necessary remedial actions to enhance the reliability. For the fault injection, an in-house tool is used, see Mohammed (2022), which can generate different types of faults with an assigned strength.

This paper is structured as follows: Section 2 briefly describes the overview of the proposed method. In Section 3, the dataset used for the fault injection and the classification are discussed. Section 4 presents the fault classification algorithm. Finally, Section 5 concludes the paper.

2. Overview of the Proposed Method

In Fig. 1 an example of the proposed classification concept implemented on an RGB camera sensor is shown. Within the proposed method, the sensor module provides its input to two AI/machine learning (ML) classification algorithms. The first algorithm identifies the fault occurrence and distinguishes the fault type. Then according to the fault type, another trained AI algorithm classifies the fault intensity. Here the intensity is categorized into the three levels low, medium, and extreme, see Fig. 1. Finally, the output message from the sensor module includes fault type and strength together with the image data, which could later be used for prognostic health management (PHM), for example.

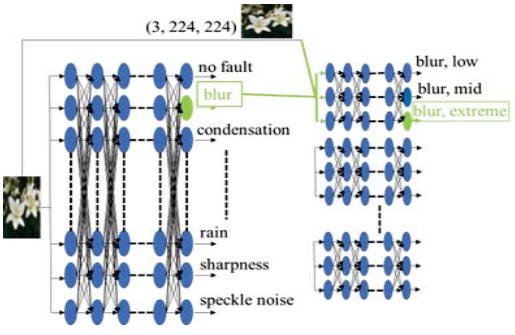

Fig. 1. Multi-level classification to determine a fault type and its associated intensity level.

3. Dataset Gathering and Preparation

In order to train a model that can categorize different fault types with different intensity levels we have developed a fault injecting tool for simulating different hardware and environmental related faults in the RGB camera, such as blur, rain, cracked lens, etc., see Mohammed (2022). From the graphical user interface (GUI) of the tool, a user can easily generate faulty data by selecting a desired fault type and intensity level. The tool generates a distorted form of the original frame corresponding to the selected fault type and intensity. The output faulty samples are set with a specific frame size and shape, compatible to the AI classifier. Moreover, the output data are automatically assigned with a specific file name in

order to meaningfully label the fault type and intensity, see Fig. 2.

As already described in this article, only three levels of intensity are used, i.e., low, medium, and extreme. It is to be noted that, in this contribution, the intensity levels are assigned based on intuition. In future work, the fault implementation will be optimized and then the intensity levels will be based on quantitative approaches.

The required dataset is generated from Flower Image Dataset, see Kaggle (2020), as an input for the fault injecting tool. The images are distorted with ten different fault types, each with three intensity levels, i.e., low, medium, and extreme. Accordingly, a name is assigned to each output image that includes the fault type and its intensity level, see e.g., Table 1 and. Fig. 2.

Fig. 2. Faulty RGB images generated by the fault injection tool with different fault types and intensities.

Pre-processed images from the dataset are randomly split such that 90% are used for the training and the rest are used for the validation. Such uneven splitting are usually done to enhance the accuracy through the training process, in addition to the use of data augmentation to avoid overfitting.

Table 1. Samples from the generated test.csv labels files.

Fault Type	Fault Intensity	Frame path
blur	1	../test/27_blur_st_1.jpg
crack	2	../test/106_crack_st_2.jpg
blur	3	../test/88_blur_st_3.jpg
rain	1	../test/19_rain_st_1.jpg
17,474 Samples are used for the training and validating sets, while 2,800 samples are used for the testing set.		

4. Fault Classification

For a safety critical automated system that uses the camera as an input sensor it is important to use some measures to continuously monitor the reliability of the sensor data. A fault classification method is indeed a fault detection mechanism which would help to assess the risk or performance degradation of the system, see e.g., Mohammed et al. (2023) and Khound et al. (2023). In the later stages, using this evaluation would further facilitate fault diagnosis and execution of a fail-safe operation (Mohammed et al., 2023), see Fig. 3.

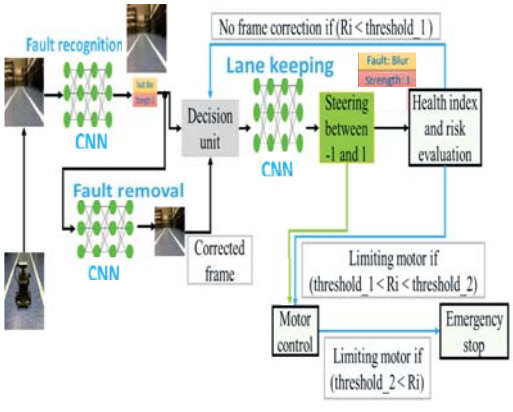

Fig. 3. Lane-keeping system with the use of the fault type and intensity (Mohammed et al., 2023).

Three convolutional neural network (CNNs) layers (O'Shea and Nash, 2015) are used with batch normalization (Ioffe and Szegedy, 2015)

and max-pooling layers (Scherer et al., 2010) after each one of the three layers, in addition to three fully-connected layers. Fig. 4 layouts the model architecture in detail. The same model adapted for classifying the fault type and predicting the corresponding fault intensity level.

Layer (type)	Output Shape	Param #
Conv2d-1	[64, 16, 224, 224]	448
BatchNorm2d-2	[64, 16, 224, 224]	32
MaxPool2d-3	[64, 16, 112, 112]	0
Conv2d-4	[64, 64, 112, 112]	9,280
MaxPool2d-5	[64, 64, 56, 56]	0
Conv2d-6	[64, 128, 56, 56]	73,856
BatchNorm2d-7	[64, 128, 56, 56]	256
MaxPool2d-8	[64, 128, 28, 28]	0
Dropout-9	[64, 100352]	0
Linear-10	[64, 128]	12,845,184
Dropout-11	[64, 128]	0
Linear-12	[64, 64]	8,256
Dropout-13	[64, 64]	0
Linear-14	[64, 32]	2,080
Dropout-15	[64, 32]	0
Linear-16	[64, 10]	330

Total params: 12,939,722
Trainable params: 12,939,722
Non-trainable params: 0

Fig. 4. Model architecture, output shape is [batch size, output channel, height resolution, width resolution], and last layer 'Linear-16' is [batch size, the 10 fault classes (or 3 in case of predicting the intensity level)].

4.1. *Model for Fault Type Prediction*

The described model is trained for 200 epochs with the cross-entropy loss function (Ho and Wookey, 2019) and stochastic gradient descent (SGD) algorithm as an optimizer (Qian, Qi et al., 2015) to classify the input image with the shape (3, 224, 224), i.e., (number of channels, height resolution, width resolution), into one of the ten classes below:

- No fault (normal input without distortion).
- Blurry image.
- Cracked lens.
- Condensation.
- Darkened image.
- Dirt on the lens.
- No chromatic abbreviation correction.
- Rain.
- Sharpen image.
- Speckle noise (or dead pixels).

The model training accuracy is about 95.3 %. While during the testing, the model predicts the fault type with 93.8% accuracy. Fig. 5 and Table 2 quantify the quality of the fault prediction outcome. Some of the faults in the tool are implemented by overlaying the fault effect on the original image. This leads to easy detection of the fault, i.e., rain and condensation, see Table 2.

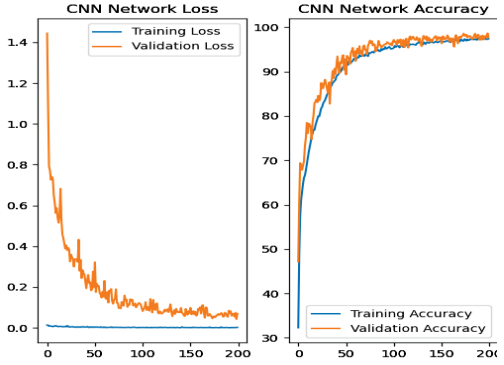

Fig. 5. Training and loss curve for the training and validation.

Table 2. Confusion matrix for the fault type prediction. The diagonal values represent the correct prediction of each fault, no-fault: 89%, blur: 87%, condensation: 98%, crack: 92%, darkness: 99%, dirt: 91%, no chromatic abbreviation correction: 97%, rain: 100%, sharpness: 89%, speckle noise: 96%.

	No Fault	Blur	Condensation	Crack	Darkness	Dirt	NCAC*	Rain	Sharpness	Speckle Noise
No Fault	0.89	0.01	0	0	0.02	0	0	0	0.07	0.01
Blur	0.077	0.87	0	0	0.02	0	0.027	0.01	0	0
Condensation	0.003	0	0.98	0	0	0.006	0	0	0.006	0
Crack	0.003	0	0	0.92	0.017	0.01	0	0.037	0.01	0
Darkness	0	0	0	0	0.99	0.006	0	0.006	0	0
Dirt	0	0	0	0	0.017	0.91	0	0.07	0.003	0
NCAC	0.006	0	0	0	0.02	0	0.97	0	0.033	0
Rain	0	0	0	0	0	0	0	1	0	0
Sharpness	0.1	0	0	0	0	0	0	0	0.89	0.003
Speckle Noise	0.04	0	0	0	0	0	0	0	0	0.96

*NACA: *No Chromatic aberration correction*

4.2. *Model for the Fault Intensity Prediction*

As mentioned before, the same model is also used for predicting the fault intensity levels for each fault type, but the number of epochs for the training differs, depending on the choice of the fault. The model correctly predicts with very high accuracy for some of the fault types. But some other fault types need more time to achieve good accuracy and some even cannot reach more than 44% accuracy, e.g., sharpness, see Table 3, Table 4, Table 5, and Table 6.

Table 3. Confusion matrix for sharpness intensity levels.

Low	0.31	0.11	0.58
Medium	0.13	0.09	0.78
Extreme	0.048	0.03	0.92
	Low	Medium	extreme
Sharpness intensity levels prediction accuracy: Low: 31%, medium 9%, and extreme: 92%			

Table 4. Confusion matrix for no chromatic abbreviation correction intensity levels.

Low	0.82	0.14	0.04
Medium	0.034	0.77	0.19
Extreme	0.0014	0.041	0.96
	Low	Medium	extreme
No chromatic abbreviation correction intensity levels prediction accuracy: Low: 88%, medium 77%, and extreme: 96%			

Table 5. Confusion matrix for darkened image fault intensity levels.

Low	0.99	0.0055	0
Medium	0.0082	0.99	0.0041
Extreme	0	0.0014	0.99
	Low	Medium	extreme
Darkened image intensity levels prediction accuracy: Low: 99%, medium 99%, and extreme: 99%			

Table 6 Confusion matrix for cracked lens fault intensity levels.

Low	0.99	0	0.0027
Medium	0	0.99	0.0068
Extreme	0	0	1
	Low	Medium	extreme
Cracked fault intensity levels prediction accuracy: Low: 99%, medium 99%, and extreme: 100%			

5. Conclusion

This contribution presents a method to classify the input RGB frame into ten classes, i.e., with no fault and nine other fault types, which can be used in any camera-based system, e.g., a lane-keeping system. This is indeed a fault detection method. Also, our fault injection tool is discussed, which is used to generate new faulty dataset from any available image dataset. The presented method includes two stages for the fault classification. The first classification stage classifies the type of a fault, including the no fault case, with an accuracy of 93.8%. In the next stage, the corresponding intensity level of the already classified fault type is predicted. The model achieves a good prediction accuracy for certain faults like, blur, speckle noise, and no chromatic abbreviation correction. But for sharpness, the accuracy is only 44%, and the correctly predicted intensity mostly works for an extreme intensity level. Varying the fault implementation method could alter the result. For future work, a better fault implementation method with comparable intensity levels will be investigated. Moreover, a more realistic approaches for implementing the rain and cracked lens fault should be explored.

Acknowledgment

This work is part of the TRUST-E project (TRUST-E, 2021) and funded by *Bundesministerium für Bildung und Forschung* (BMBF), under contract PENTA 16ME0320K -16ME0329. The authors would like to thank all the members of the project for useful discussions which stimulated the writing of this paper.

References

Baljak, V., K. Tei, and S. Honiden (2013). Fault classification and model learning from sensory Readings — Framework for fault tolerance in wireless sensor networks. In *IEEE Eighth International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pp. 408-413.

Blanke M., W. C. Frei, F. Kraus, J. R. Patton, and M. Staroswiecki (2000). What is fault-tolerant control? *IFAC Proceedings Volumes 33* (11), 41-52.

Gao Z., C. Cecati, and S. X. Ding (2015). A survey of fault diagnosis and fault tolerant techniques—part I: fault diagnosis with model-based and signal based approaches. *IEEE Transaction on Industrial Electronics 62*(6), 3757-3767.

Ioffe, S. and C. Szegedy, (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 448-456.

Jan, S. U., Y. -D. Lee, J. Shin, and I. Koo (2017). Sensor fault classification based on support vector machine and statistical time-domain features. *IEEE Access* 5, 8682-8690.

Kaggle (2020). Flower Image Dataset https://www.kaggle.com/datasets/aksha05/flower-image-dataset?resource=download (accessed Mar. 30, 2023).

Khound, P., O. Mohammed, P. Su, D. Chen, and F. Gronwald (2023). Performance index modeling from fault injection analysis for an autonomous lane-keeping system. In *Proceedings of the 33rd European Safety and Reliability Conference (ESREL)*.

Liu Y., L. Wei, B. Luo, and Q. Xu. (2017). Fault injection attack on deep neural network. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 131-138.

Mohammed, O. (2022). Fault injecting tool. https://github.com/omarMohammed-USI/Faults-injecting-tool-USI. (accessed Mar. 30, 2023).

Mohammed, O., P. Khound, B. Vandevelde, and F. Gronwald (2023). Health index modeling for trustable electronic sensor systems in an autonomous application. In *Smart Systems Integration (SSI)*.

O'Shea, K. and R. Nash (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

Qian, Q., R. Jin, J. Yi, L. Zhang, and S. Zhu (2015). Efficient distance metric learning by adaptive sampling and mini-batch stochastic gradient descent (SGD). *Machine Learning* 99, 353-372.

Scherer, D., A. Müller, and S. Behnke (2010). Evaluation of pooling operations in convolutional architectures for object recognition. In *International Conference on Artificial Neural Networks*, pp. 92-101.

Secci, F. and A. Ceccarelli (2020). On failures of RGB cameras and their effects in autonomous driving applications. In *31st International Symposium on Software Reliability Engineering (ISSRE)*, pp. 13-24.

Su, P. and D. Chen. Using fault injection for the training of functions to detect soft errors of DNNs in automotive vehicles. In *New Advances in Dependability of Networks and Systems: Proceedings of the Seventeenth International Conference on Dependability of Computer Systems DepCoS-RELCOMEX, Wrocław, Poland*, pp. 308–318.

TRUST-E (2021). Project overview. https://www.edacentrum.de/en/projects/TRUST-E. (accessed Mar. 30, 2023).

Ho, Y. and S. Wookey (2019). The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling. *IEEE Access* 8, pp.4806-4813.