Deep Learning Classifiers for Automated Driving: Quantifying the Trained DNN Model's Vulnerability to Misclassification

Himanshu Agarwal[1,2], Rafal Dorociak[1] and Achim Rettberg[2,3]
[1] *HELLA GmbH & Co. KGaA, Lippstadt, Germany*
[2] *Department of Computing Science, Carl von Ossietzky University Oldenburg, Germany*
[3] *University of Applied Sciences Hamm-Lippstadt, Lippstadt, Germany*

Keywords: Image Classification, Deep Learning, Deep Neural Network, Vulnerability to Misclassification, Automated Driving.

Abstract: The perception-based tasks in automated driving depend greatly on deep neural networks (DNNs). In context of image classification, the identification of the critical pairs of the target classes that make the DNN highly vulnerable to misclassification can serve as a preliminary step before implementing the appropriate measures for improving the robustness of the DNNs or the classification functionality. In this paper, we propose that the DNN's vulnerability to misclassifying an input image into a particular incorrect class can be quantified by evaluating the similarity learnt by the trained model between the true class and the incorrect class. We also present the criteria to rank the DNN model's vulnerability to a particular misclassification as either low, moderate or high. To argue for the validity of our proposal, we conduct an empirical assessment on DNN-based traffic sign classification. We show that upon evaluating the DNN model, most of the images for which it yields an erroneous prediction experience the misclassifications to which its vulnerability was ranked as high. Furthermore, we also validate empirically that all those possible misclassifications to which the DNN model's vulnerability is ranked as high are difficult to deal with or control, as compared to the other possible misclassifications.

1 INTRODUCTION

In the recent years, the advancements in the field of autonomous driving have been reinforced with the progress in the techniques of artificial intelligence, especially deep learning. A survey of the current state-of-the-art deep learning technologies, e.g., deep convolutional neural networks, deep reinforcement learning, etc., has been presented in Grigorescu et al. (2019). The deep convolutional neural networks have led to various breakthrough contributions in object detection and image classification tasks, such as in Krizhevsky et al. (2012) and Sermanet et al. (2014). In context of autonomous driving, deep learning plays a major role in perception-based tasks such as pedestrian detection (Ouyang et al., 2017), traffic sign detection (Zhu et al., 2016), etc. One of the challenges in safe automated driving is related to the robustness of the artificial intelligence or deep learning techniques (Muhammad et al., 2020). In context of tasks related to image classification, it must be ensured that the deep neural networks (DNNs) are not just accurate but also robust against the perturbations that a vehicle might encounter during its operation. The perturbations, for instance, can be random noise in the input images or even shift in brightness, contrast, etc. These perturbations can influence the DNN's decision significantly and aggravate the chances of misclassifying an input image into an incorrect class that is highly similar with respect to the true class. The questions which need to be addressed are as follows:

- Since an image $\boldsymbol{X}$ belonging to the true class k_1 can be misclassified into any of the remaining target classes, how can we rank all these possible misclassifications on the basis of how vulnerable is the DNN model to each of them individually?
- If the misclassification from a true class k_1 into an another class $k_2 (\neq k_1)$ is ranked to be offering a high vulnerability, do the perturbations in the images exploit this vulnerability and give rise to a higher misclassification rate from k_1 into k_2?

Such investigations can help in determining the set of critical misclassifications that need laborious miti-

Agarwal, H., Dorociak, R. and Rettberg, A.
Deep Learning Classifiers for Automated Driving: Quantifying the Trained DNN Model's Vulnerability to Misclassification.
DOI: 10.5220/0010481302110222
In *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2021)*, pages 211-222
ISBN: 978-989-758-513-5

gation efforts to enhance the robustness of the classification function.

Our contributions in this paper are listed below:

- We propose an approach to estimate the trained DNN model's vulnerability to a particular misclassification. Along with it, we propose the criteria to categorise the DNN model's vulnerability to a particular misclassification into one of the three levels: *low*, *moderate* or *high*.
- We further argue that the ease with which the rate of a particular misclassification can be kept under control depends on the estimated value of the DNN model's vulnerability to it. In other words, we highlight that it is easier to control those misclassifications to which the model is estimated to be lowly vulnerable, as compared to the misclassifications to which it is estimated to be highly vulnerable. We validate our arguments empirically.

The practical examples and the corresponding experimentation[1] conducted in context of the above-mentioned contributions have been extensively discussed in the paper. The remainder of the paper is organized as follows. In Section 2, we present a brief background to discuss the rationale behind our proposed approach of estimating the DNN's vulnerability to a particular misclassification. In Section 3, we present a detailed description of the approach, and an experimental analysis in context of traffic sign classification. Section 4 addresses the second part of our contributions, i.e., an empirical investigation to show that the set of misclassifications to which the DNN model is ranked to be highly vulnerable are rather difficult to manage, even after the implementation of the measures to control the misclassification rate. Finally, Section 5 presents a conclusion along with a brief overview on the possible future directions.

2 BACKGROUND

In context of image classification, the objective is to ensure that a classifier is able to correctly distinguish between the target classes. In Tian et al. (2020), for instance, the authors assess the classifier model's ability to distinguish between any two classes by computing the *confusion score*. This score is based on measuring the euclidean distance between the neuron activation probability vectors corresponding to the two classes. Such analyses are usually performed by evaluating the trained model against a set of independent (test) images. However, such evaluations against a set of test images are not sufficient to realize the classifier's ability to distinguish between the classes (since the completeness of the test data serves as a major challenge). As an additional assessment, the critical misclassifications corresponding to the DNN model can be identified by determining the set of possible misclassifications to which the model is highly vulnerable. In Agarwal et al. (2021), it has been argued that the classifier's vulnerability to a particular misclassification (let us say, from the true class k_1 into an incorrect class k_2 or vice versa) can be assessed in terms of the similarity between the dominant visual characteristics of the corresponding two classes (i.e., k_1 and k_2). For instance, the dominant visual characteristics in the traffic signs are shape and color (Gao et al., 2006). By evaluating the overlap in terms of (a) the shape of the traffic sign board, and (b) the color combination of the border and the background, the similarity between any two traffic sign classes can be analysed *a priori* (Agarwal et al., 2021). The obtained measure of similarity between the classes k_1 and k_2 is recognized as a measure of the classifier's vulnerability to misclassifying an input image belonging to the class k_1 into the class k_2 or vice versa. Higher similarity between the two classes is considered to induce higher vulnerability to the corresponding misclassification.

We illustrate it further with an example. In this regard, we trained a DNN to classify the different traffic signs from the German Traffic Sign Recognition Benchmark (GTSRB) dataset (Stallkamp et al., 2012). Mathematically, we represent the DNN (classifier) model as $f : \boldsymbol{X} \in \mathbb{R}^{(48\times48\times3)} \longrightarrow \hat{y} \in \{1,2,...,43\}$, where $\hat{y}$ is the predicted class for the input image $\boldsymbol{X}$. The hyperparameters and the training details related to it are provided in Appendix A. From the set of 10000 test images, we choose 2207 images which belong to the *danger* sign type. We determine the percentage of these images that the DNN model f misclassifies into: (a) another danger sign, (b) a speed limit or a prohibitory sign, (c) a derestriction sign, and (d) a mandatory sign. The results are graphically presented in Figure 1. We first consider the results plotted for the GTSRB original test images (i.e., the test images without any perturbations deliberately enforced by us). We observe a higher rate of misclassification of a danger sign into another danger sign, as compared to the other misclassifications. All the traffic signs that belong to the danger sign type have high similarity due to their two dominant visual characteristics being the same. This can perhaps be a potential

[1]For the experimentation, the libraries: Numpy (Harris et al., 2020), Keras (Chollet et al., 2015), SciPy (Virtanen et al., 2020), Scikit-learn (Pedregosa et al., 2011) and Matplotlib (Hunter, 2007) were used along with some of the other standard Python libraries and their functions.

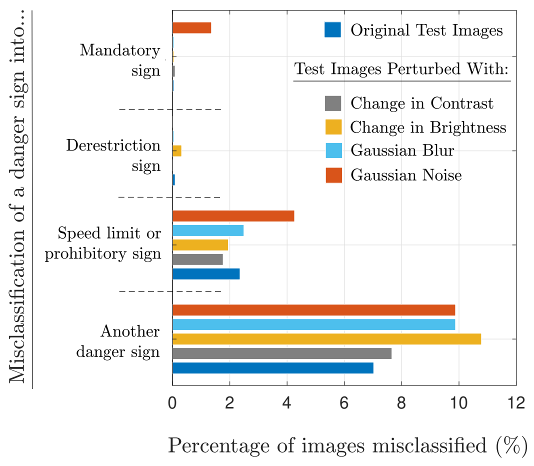


Figure 1: Graph showing the percentage of the *danger* sign images in the test dataset that were misclassified into: another danger sign, speed limit or prohibitory sign, derestriction sign, and mandatory sign.

source of the higher rate of misclassification of a danger sign into another danger sign, as observed in Figure 1. It can also be observed that some common perturbations (e.g., contrast change, gaussian blur, etc.) enforced by us on the GTSRB original test images further aggravate the DNN's susceptibility to misclassification of a danger sign into another danger sign. This indicates that the DNNs used for image classification are rather more susceptible or vulnerable to misclassification of an input image into the classes that have stronger visual similarity with the true class.

This estimation of similarity depending on the chosen predominant visual characteristics can facilitate the planning of the pre-training activities. For instance, it can help the experts in choosing a suitable configuration of the DNN architecture or tailoring the DNN-based strategy appropriately in a manner that has the potential to minimize those misclassifications wherein the true class and the incorrect class are highly similar in terms of their visual appearance. However, it must be noted that the visual characteristics considered in the abovementioned *a priori* analysis of similarity may not always necessarily be the features that actually influence the decision of the classifier (Ribeiro et al., 2016). Thus, the similarity perceived by humans between any two classes is not necessarily the similarity that will be learnt by the DNN model via training. As an extension to this concept, unlike the approach discussed in Agarwal et al. (2021), in this paper, we propose to estimate the DNN model's vulnerability to misclassification from class k_1 into class k_2 by measuring the similarity learnt by the trained model between the classes k_1 and k_2. In our approach of measuring the class similarity learnt by the DNN model, we use the images from the training dataset itself. Hence, by virtue of this vulnerability estimation, we identify the set of critical misclassifications without actually evaluating the trained model against any independent or seperate set of the images (i.e., test data). The approach has been elaborated in Section 3.

3 VULNERABILITY TO MISCLASSIFICATION

Let us assume the DNN (classifier) is trained for K number of target classes. A class $k \in \mathcal{K} = \{1, 2, ..., K\}$ can be misclassified into any of the remaining $(K-1)$ classes. Therefore, the set of misclassifications that it can incur is represented as:

$$\mathcal{M} = \{\, (k_a, k_b) \mid k_a \neq k_b,\, k_a \in \mathcal{K},\, k_b \in \mathcal{K} \,\}, \qquad (1)$$

where k_a and k_b represent the true class and the incorrect class, respectively. The total number of misclassifications that are possible is $|\mathcal{M}| = K(K-1)$.

The DNN's vulnerability to a particular misclassification, let us say, (k_a, k_b), is determined by evaluating the similarity between the two classes k_a and k_b. Followed by this, the categorisation criteria is implemented in order to identify the set of possible misclassifications to which the DNN model is highly vulnerable. The approach has been discussed in this section, along with an experimental analysis.

3.1 Similarity between the Classes

In order to measure the class similarity learnt by the trained model, the approach discussed in Agarwal et al. (2020) is used. The trained DNN model predicts the logits $\mathbf{z}^k = [z_1, z_2, ..., z_K]$ for an input image belonging to a class $k \in \mathcal{K}$. The predicted logits $\mathbf{z}^k$ is then modelled as a multivariate normal distribution $\mathcal{N}_k$. Mathematically, $\mathbf{z}^k \sim \mathcal{N}_k(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ represent the $K \times 1$ mean vector and the $K \times K$ covariance matrix of the distribution $\mathcal{N}_k$. The similarity between any two classes is determined by calculating the Bhattacharyya distance between their corresponding modelled multivariate normal distributions. Let us consider the classes k_a and k_b here. The Bhattacharyya distance d_{k_a,k_b} between $\mathcal{N}_{k_a}(\boldsymbol{\mu}_{k_a}, \boldsymbol{\Sigma}_{k_a})$ and $\mathcal{N}_{k_b}(\boldsymbol{\mu}_{k_b}, \boldsymbol{\Sigma}_{k_b})$ is determined using Equation 2 (Kashyap, 2019).

$$\begin{aligned} d_{k_a,k_b} =& \frac{1}{8}(\boldsymbol{\mu}_{k_a} - \boldsymbol{\mu}_{k_b})^{\mathrm{T}} \boldsymbol{\Sigma}_{\text{avg}}^{-1} (\boldsymbol{\mu}_{k_a} - \boldsymbol{\mu}_{k_b}) \\ &+ \frac{1}{2} \ln \left(\frac{|\boldsymbol{\Sigma}_{\text{avg}}|}{\sqrt{|\boldsymbol{\Sigma}_{k_a}|\,|\boldsymbol{\Sigma}_{k_b}|}} \right), \end{aligned} \qquad (2)$$

where

$$\mathbf{\Sigma}_{\text{avg}} = \frac{1}{2}\,(\mathbf{\Sigma}_{k_a} + \mathbf{\Sigma}_{k_b}). \tag{3}$$

The similarity between the classes k_a and k_b will be inversely proportional to the computed value of the Bhattacharyya distance d_{k_a,k_b}. It must be noted that the Bhattacharyya distance computed above is symmetric, i.e., $d_{k_a,k_b} = d_{k_b,k_a}$.

3.2 Estimation of Vulnerability

Using the approach discussed in Section 3.1, we can compute the value of d to assess the similarity of a class $k \in \mathcal{K}$ with every other class in $\mathcal{K} \setminus \{k\}$. Since the number of target classes is K, we will have a total $K(K-1)$ values of the Bhattacharyya distances. We accumulate all these obtained values in a set $\mathcal{D}$, as shown below:

$$\mathcal{D} = \{\, d_{k_a,k_b} \mid k_a \neq k_b,\, k_a \in \mathcal{K},\, k_b \in \mathcal{K} \,\}. \tag{4}$$

Among all these $K(K-1)$ values, let us assume that the maximum value is observed to be $d_{\max}$, i.e.,

$$d_{\max} = \max\,(\mathcal{D}). \tag{5}$$

Now, we represent the DNN model's vulnerability to the misclassification (k_a, k_b) as:

$$v(k_a,k_b) = 1 - \frac{d_{k_a,k_b}}{d_{\max}}. \tag{6}$$

Note that $v(k_a,k_b) \in [0,1)$ and $v(k_a,k_b) = v(k_b,k_a)$. Since $v(k_a,k_b) = 1$ is possible only if $k_a = k_b$ (which does not represent the case of misclassification), therefore, the value of 1 is excluded from the specified range of $v(k_a,k_b)$. The value of $v(k_a,k_b)$ closer to 1 indicates higher vulnerability to the corresponding misclassification (k_a,k_b).

3.3 Categorisation Criteria

Using the approach discussed above, we can acquire the model's vulnerability values corresponding to all the $K(K-1)$ possible misclassifications. We collect all these obtained values in a set $\mathcal{V}$, as shown below:

$$\mathcal{V} = \{\, v(k_a,k_b) \mid (k_a,k_b) \in \mathcal{M} \,\}. \tag{7}$$

We will now categorise the model's vulnerability to a particular misclassification into one of the levels: *low*, *moderate* or *high*, using certain statistical measures, as discussed below.

To the misclassification (k_a,k_b), the DNN model will be considered to have:

- low vulnerability if $0 \leq v(k_a,k_b) < p_{25}$,
- moderate vulnerability if $p_{25} \leq v(k_a,k_b) < p_{75}$, and
- high vulnerability if $p_{75} \leq v(k_a,k_b) < 1$,

where p_{25} and p_{75} are the 25th and the 75th percentile of the values in the set $\mathcal{V}$, respectively.

The set of the misclassifications to which the DNN model's vulnerability is ranked as low, moderate and high are denoted as $\mathcal{M}_{\text{low}}, \mathcal{M}_{\text{moderate}}$ and $\mathcal{M}_{\text{high}}$, respectively. They are mathematically represented as:

$$\mathcal{M}_{\text{low}} = \{\, (k_a,k_b) \in \mathcal{M} \mid 0 \leq v(k_a,k_b) < p_{25} \,\}, \tag{8}$$

$$\mathcal{M}_{\text{moderate}} = \{\, (k_a,k_b) \in \mathcal{M} \mid p_{25} \leq v(k_a,k_b) < p_{75} \,\}, \tag{9}$$

$$\mathcal{M}_{\text{high}} = \{\, (k_a,k_b) \in \mathcal{M} \mid p_{75} \leq v(k_a,k_b) < 1 \,\}. \tag{10}$$

Note that: (i) $\mathcal{M}_{\text{low}} \cap \mathcal{M}_{\text{moderate}} = \emptyset$, (ii) $\mathcal{M}_{\text{moderate}} \cap \mathcal{M}_{\text{high}} = \emptyset$, (iii) $\mathcal{M}_{\text{low}} \cap \mathcal{M}_{\text{high}} = \emptyset$, and (iv) $\mathcal{M}_{\text{low}} \cup \mathcal{M}_{\text{moderate}} \cup \mathcal{M}_{\text{high}} = \mathcal{M}$.

In order to support the validity of this proposed criteria of categorising the DNN model's vulnerability, we conducted an experimental analysis, which has been discussed in Section 3.4.

3.4 Experimental Analysis

3.4.1 DNN Training

We conducted our experiment for the classification of the traffic signs from the GTSRB dataset. The DNN model used here in this experimental analysis is the same model $f : \boldsymbol{X} \in \mathbb{R}^{(48\times48\times3)} \rightarrow \hat{y} \in \{1,2,...,43\}$, which was trained for the illustration of the example presented in Section 2. A brief summary of the DNN architecture and the associated details related to it's training are provided in Appendix A. Since the total number of target traffic sign classes is $K = 43$, the number of possible misclassifications is $|\mathcal{M}| = 43 \times 42 = 1806$.

3.4.2 Vulnerability to Misclassification

We first determined the similarity learnt by the DNN model f between all the 43 target traffic sign classes, using the approach discussed in Section 3.1. Note that in order to model the logits $\boldsymbol{z}_k$ as a multivariate normal distribution, the samples of the predicted logits $\boldsymbol{z}_k$ were collected for all the images in the training data that belong to the class k. Followed by the measurement of similarity between the classes, we determined the model's vulnerability (v) to each of the 1806 possible misclassifications, as discussed in Section 3.2. The distribution of the obtained 1806 values of v is illustrated as a histogram in Figure 2. The 25th and the 75th percentile of the distribution are $p_{25} = 0.4354$ and $p_{75} = 0.6455$, respectively.

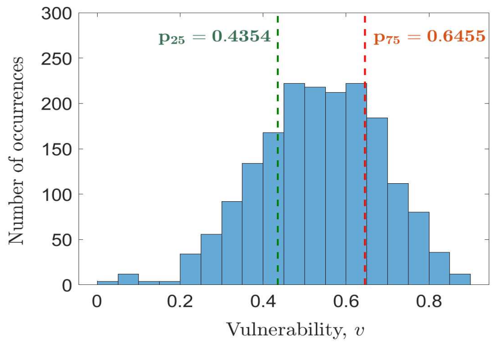


Figure 2: Distribution of the trained DNN model's estimated vulnerability (v) to the 1806 possible misclassifications among the 43 traffic sign classes. The terms p_{25} and p_{75} denote the 25th and the 75th percentile of the distribution, respectively.

3.4.3 Evaluation

The trained model f was evaluated on $\tau = 10000$ GTSRB test images. We represent the set of these test images as $\mathcal{X}_{\text{test}} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_\tau\}$. An accuracy of 96.12% was observed. Alternatively, it can be said that the model misclassifies $\rho = 388$ traffic sign images from $\mathcal{X}_{\text{test}}$. Let us denote the set of the misclassified images as $\mathcal{X}_{\text{m}} = \{\boldsymbol{X}_{\text{m}_1}, \boldsymbol{X}_{\text{m}_2}, ..., \boldsymbol{X}_{\text{m}_\rho}\}$, such that $\mathcal{X}_{\text{m}} \subset \mathcal{X}_{\text{test}}$. The corresponding misclassifications experienced by the model for the images in $\mathcal{X}_{\text{m}}$ are put together in the set Ψ_{m}, as shown below:

$$\Psi_{\text{m}} = \{(y_{\text{m}_1}, \hat{y}_{\text{m}_1}), (y_{\text{m}_2}, \hat{y}_{\text{m}_2}), ..., (y_{\text{m}_\rho}, \hat{y}_{\text{m}_\rho})\}, \quad (11)$$

where the ordered pair $(y_{\text{m}_i}, \hat{y}_{\text{m}_i}) \in \mathcal{M}$ signifies that for the image $\boldsymbol{X}_{\text{m}_i} \in \mathcal{X}_{\text{m}}$, the actual class is y_{m_i}; however, the model f predicts it as a class $\hat{y}_{\text{m}_i} \in \mathcal{K} \setminus \{y_{\text{m}_i}\}$.

Out of all the images in $\mathcal{X}_{\text{m}}$, the number of images for which the corresponding misclassifications in Ψ_{m} were ranked to be offering the model f, based on our proposed criteria, a:

(a) *Low* vulnerability equals:

$$n_{\text{low}} = \sum_{i=1}^{\rho} [0 \leq v(y_{\text{m}_i}, \hat{y}_{\text{m}_i}) < p_{25}], \quad (12)$$

(b) *Moderate* vulnerability equals:

$$n_{\text{moderate}} = \sum_{i=1}^{\rho} [p_{25} \leq v(y_{\text{m}_i}, \hat{y}_{\text{m}_i}) < p_{75}], \quad (13)$$

(c) *High* vulnerability equals:

$$n_{\text{high}} = \sum_{i=1}^{\rho} [p_{75} \leq v(y_{\text{m}_i}, \hat{y}_{\text{m}_i}) < 1], \quad (14)$$

where [...] in (12), (13) and (14) denote the Iverson brackets, and $v(y_{\text{m}_i}, \hat{y}_{\text{m}_i})$ denotes the model's vulnerability to the misclassification $(y_{\text{m}_i}, \hat{y}_{\text{m}_i})$. Note that:

$$n_{\text{low}} + n_{\text{moderate}} + n_{\text{high}} = \rho. \quad (15)$$

For $\mathcal{X}_{\text{test}}$, using the trained classifier model f, we got the values of n_{low}, n_{moderate} and n_{high} as 1, 103 and 284, respectively. This has been graphically presented in Figure 3a. It can be deduced that:

$$n_{\text{high}} > n_{\text{moderate}} > n_{\text{low}}, \quad (16)$$

which implies, the model is most likely to incur those misclassifications to which it's vulnerability was ranked to be *high*. To further validate this, we continue our analysis using the perturbed test images. These perturbations are: change in contrast[2], change in brightness[3], gaussian blur[4] and gaussian noise[5]. We apply these perturbations seperately to every image $\boldsymbol{X}_j$ in $\mathcal{X}_{\text{test}}, j = 1$ to τ. Hence, we have four different sets of perturbed test images, i.e., $\mathcal{X}_{\Delta\text{contrast}}$, $\mathcal{X}_{\Delta\text{brightness}}$, $\mathcal{X}_{\Delta\text{blur}}$ and $\mathcal{X}_{\Delta\text{noise}}$. The corresponding obtained values of $n_{\text{low}}, n_{\text{moderate}}$ and n_{high} are graphically presented in Figure 3b - 3e. It can be observed that the condition in (16) holds true for the analysis associated with the perturbed test images as well.

The criteria proposed for categorisation seems coherent or convincing when analysed experimentally on a classification problem, as it does helps in determining the set of critical misclassifications, i.e., the set of misclassifications which the DNN model can most likely incur. Hence, the concept of using the class similarity learnt by the model for quantifying it's vulnerability to every possible misclassification appears to be reasonable.

4 Further Empirical Investigation

4.1 Concept and Purpose

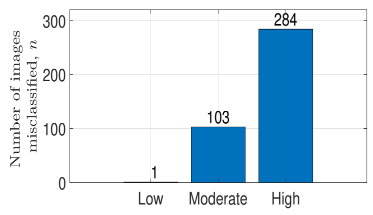
One of the ways to minimize the possibility of an erroneous prediction or a misclassification incurred by a DNN model is to integrate a mechanism that can provide the classification functionality an alternative to not yield any decision in case of lack of certainty. For this purpose, in addition to the classifier

[2]every pixel $p(r,s,t) \in [0,1]$ of a test image $\boldsymbol{X}_j$ undergoes a transformation: $p(r,s,t) \rightarrow \gamma p(r,s,t)$, using a randomly chosen $\gamma \in [1,3]$.

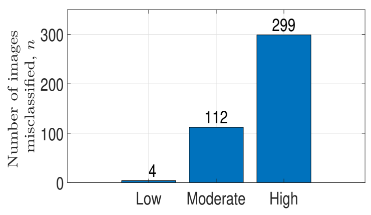
[3]every pixel $p(r,s,t) \in [0,1]$ of a test image $\boldsymbol{X}_j$ undergoes a transformation: $p(r,s,t) \rightarrow p(r,s,t) + \delta$, using a randomly chosen $\delta \in [0,0.4]$.

[4]function in the OpenCV library (https://github.com/opencv/opencv). To perturb a test image $\boldsymbol{X}_j$, we choose randomly a kernel size $\eta \in \{3,5,7\}$, and the standard deviation $\sigma \in [0.5,1.5]$ along both x and y directions.

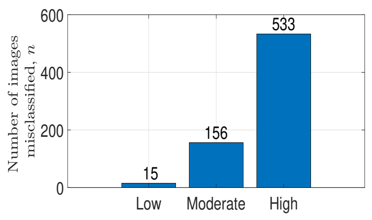
[5]adding to a test image $\boldsymbol{X}_j$ a random normal gaussian noise with a mean $\mu = 0$ and a randomly chosen standard deviation $\sigma \in [0.01,0.1]$.



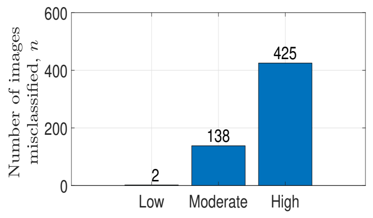
(a) Original test images



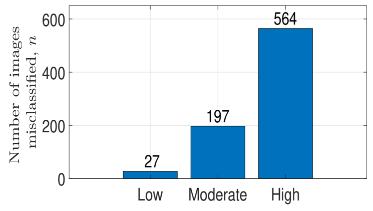
(b) Test images with contrast change



(c) Test images with brightness change



(d) Test images with gaussian blur



(e) Test images with gaussian noise

Figure 3: Graphs showing the values of n_{low}, n_{moderate} and n_{high}, obtained when the trained DNN model was evaluated for different sets of the test images: (a) X_{test}, (b) $X_{\Delta\text{contrast}}$, (c) $X_{\Delta\text{brightness}}$, (d) $X_{\Delta\text{blur}}$, and (e) $X_{\Delta\text{noise}}$.

model f (trained earlier for the experimental analysis in Section 3.4), we implement a network of differently trained DNN classifiers that collaborate to simultaneously yield a prediction for the input image. Here, we refer to the model f as the primary classifier (f_{primary}) and the classifier models trained additionally to work in conjunction with it as the secondary classifiers. The final prediction for a given input image can be acquired by the principle of unanimity voting. If the prediction ($\hat{y}_{\text{secondary}}$) obtained from this network of secondary classifiers is not congruous with the prediction ($\hat{y}_{\text{primary}}$) obtained from f_{primary}, then the final prediction ($\hat{y}_{\text{final}}$) for the input image is considered to be *undecided*, as shown below:

$$\hat{y}_{\text{final}} = \begin{cases} \hat{y}_{\text{primary}}, & \text{if } \hat{y}_{\text{primary}} = \hat{y}_{\text{secondary}} \\ \phi, & \text{if } \hat{y}_{\text{primary}} \neq \hat{y}_{\text{secondary}}, \end{cases} \quad (17)$$

where ϕ denotes the final prediction as undecided, i.e., no prediction is being issued for the given input image. In the event of an undecided outcome, the vehicle, for instance, can transit into a safe state. The safe state depends on many factors. One of the most important factors is the level of automation[6] the vehicle possesses. For instance, in case of Level 2 automation, the safe state could be the switching off of the AI functionality, while for Level 3 automation, it could be the handover to the driver. In cases of higher levels of automation, i.e., Level 4 or 5, defining the safe state will require a detailed review of the possible hazards and the associated risks. Also, the factors such as driving scenario, operational conditions, traffic environment, etc. will play a major role. However, in this paper, defining an appropriate safe state for the vehicle in this context is not within the scope, and therefore, we do not address it in detail here.

Theoretically, it is expected here that for all the ρ number of test images that are misclassified by the model f_{primary}, the final prediction $\hat{y}_{\text{final}}$ will be undecided (ϕ). The purpose of our investigation here is to validate the following arguments:

- **Argument 1:** For almost every test image for which the primary classifier model incurs a misclassification to which it's vulnerability was ranked as *low*, the use of the abovementioned mechanism will result in the final prediction as undecided (which is actually desired).
- **Argument 2:** On the other hand, if the primary classifier model, for an input test image, incurs a misclassification to which it's vulnerability was ranked as *moderate*, the possibility of obtaining the final prediction as undecided is lower. Moreover, this possibility gets further lowered for the misclassifications to which the model's vulnerability was ranked as *high*.

The validation of these arguments will demonstrate the fact that among all the possible misclassifications in $\mathcal{M}$, the misclassifications to which the primary classifier model's vulnerability is high (i.e., the misclassifications categorised into the set $\mathcal{M}_{\text{high}}$) will need relatively more tedious mitigation efforts as compared to the other possible misclassifications in $\mathcal{M}_{\text{moderate}}$ and $\mathcal{M}_{\text{low}}$.

[6]SAE J3016 standard (SAE International, 2014) states six levels of driving automation, i.e., Level 0 (no automation) to Level 5 (full automation).

4.2 Mechanism

4.2.1 Primary and Secondary Classifiers

Firstly, the primary classifier f_{primary} maps an input traffic sign image $\boldsymbol{X}$ into one of the 43 classes in the GTSRB dataset. Mathematically, it is represented as:

- Primary classifier $f_{\text{primary}} : \boldsymbol{X} \in \mathbb{R}^l \longrightarrow \hat{y}_{\text{primary}} \in \{1, 2, ..., 43\}$.

where l denotes the size of the input image $\boldsymbol{X}$, and $\hat{y}_{\text{primary}}$ is the traffic sign class predicted by f_{primary}. This is also shown in Figure 4.

4.2.2 Secondary Classifiers

Some traffic signs vary in terms of their physical characteristics, i.e., variation in terms of shape, color or both. However, all the traffic signs of a particular sign type[7] have the same physical characteristics. The classifier f_{primary}, for a given input image, can experience either an *inter-sign-type*[8] misclassification or an *intra-sign-type*[9] misclassification.

An inter-sign-type misclassification by f_{primary} implies that it is perhaps not able to correctly visualize the high-level feature(s) (i.e., shape and/or color) of the traffic sign in the given input image. In order to deal with such misclassifications, a suitable approach can be to implement two additional diverse classifiers: one for predicting the shape of the traffic sign and the other to classify the traffic sign based on the prominent colors present on the sign board. Since these classifiers focus on classifying the input traffic sign image on the basis of just the corresponding high-level feature, it is expected that these classifiers will be easier to train and have high accuracy. We refer to these two secondary classifiers as shape and color classifier, i.e., f_{shape} and f_{color}, respectively. The classes to which these classifiers map the input image $\boldsymbol{X}$ are presented in Figure 5. All the 43 traffic signs in GTSRB can be grouped into the classes $\{S_1, S_2, ..., S_5\}$ and $\{C_1, C_2, ..., C_5\}$ on the basis of the shape of the traffic signs and the prominent colors present on the sign boards, respectively. Mathematically, we represent the two classifiers as:

- Shape classifier $f_{\text{shape}} : \boldsymbol{X} \in \mathbb{R}^l \longrightarrow \hat{S}_{\text{secondary}} \in \{1, 2, ..., 5\}$, and

[7] the 39 traffic signs in GTSRB belong to one of the sign types: speed limit, prohibitory, danger, mandatory and derestriction. The remaining 4 signs (priority road, yield to cross, do not enter, stop) are independent sign types.

[8] predicted traffic sign is of a type, different than that of the actual traffic sign.

[9] predicted traffic sign is of a type, same as that of the actual traffic sign.

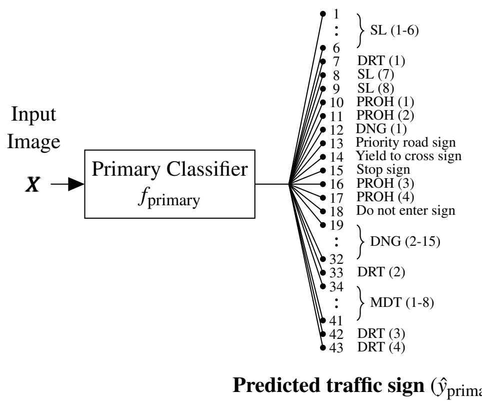
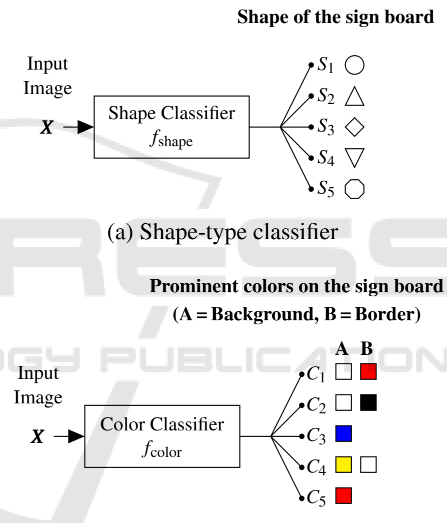


Figure 4: Primary classifier to predict the traffic sign class in an input image $\boldsymbol{X}$. The sign type for every traffic sign class is also mentioned, where SL, DRT, PROH, DNG and MDT denote the sign types: speed limit, derestriction, prohibitory, danger and mandatory, respectively.

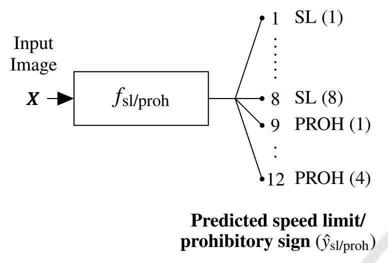

Shape of the sign board
Input Image
$\boldsymbol{X}$
Shape Classifier f_{shape}
S_1 ○
S_2 △
S_3 ◇
S_4 ▽
S_5 ⬡
(a) Shape-type classifier
Prominent colors on the sign board (A = Background, B = Border)
A B
Input Image
$\boldsymbol{X}$
Color Classifier f_{color}
C_1 □ 🟥
C_2 □ ⬛
C_3 🟦
C_4 🟨 □
C_5 🟥

(b) Color-type classifier
Figure 5: Shape and color type classifiers that are trained for predicting the shape and the prominent colors on the sign board in the input traffic sign image, respectively.

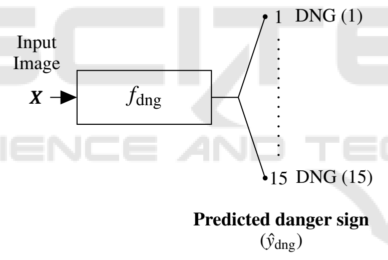
- Color classifier $f_{\text{color}} : \boldsymbol{X} \in \mathbb{R}^l \longrightarrow \hat{C}_{\text{secondary}} \in \{1, 2, ..., 5\}$.

Let us consider that the classifiers f_{shape} and f_{color} predict $\hat{S}_{\text{secondary}} = 2$ (triangle pointed upwards) and $\hat{C}_{\text{secondary}} = 1$ (white background and red border) for an input image $\boldsymbol{X}$. This implies that they collaborate to predict the sign type of the traffic sign in $\boldsymbol{X}$ as *danger*, since all the danger traffic signs are triangular (pointed upwards) in shape and bear a white background with red border. Now, we use an additional classifier f_{dng}, which, unlike f_{primary}, is trained to map $\boldsymbol{X}$ into one of the 15 danger traffic sign classes only. Similarly, we can train the additional classifiers for the other sign types, namely, speed limit, prohibitory, derestriction and mandatory. Note that all the speed limit and the prohibitory traffic signs have the same shape and bear the same prominent colors, therefore, we train a common classifier $f_{\text{sl/proh}}$ with all the speed limit and the prohibitory signs as the target classes. These sign type specific classifiers are schematically represented in Figure 6, and their mathematical representations are given below:

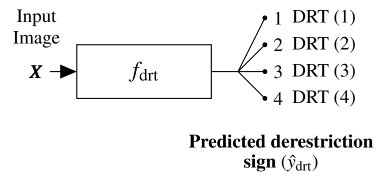
- Speed limit/prohibitory signs classifier $f_{\text{sl/proh}}$: $\boldsymbol{X} \in \mathbb{R}^l \longrightarrow \hat{y}_{\text{sl/proh}} \in \{1, 2, ..., 12\}$,
- Danger signs classifier f_{dng} : $\boldsymbol{X} \in \mathbb{R}^l \longrightarrow \hat{y}_{\text{dng}} \in \{1, 2, ..., 15\}$,



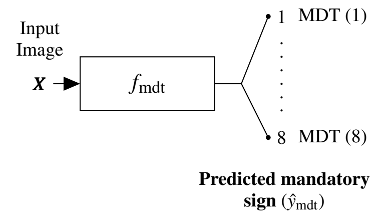
(a) Speed limit/prohibitory signs classifier



(b) Danger signs classifier



(c) Derestriction signs classifier



(d) Mandatory signs classifier

Figure 6: Sign type specific classifiers, wherein all the corresponding target traffic sign classes have the same physical characteristics (shape and prominent colors).

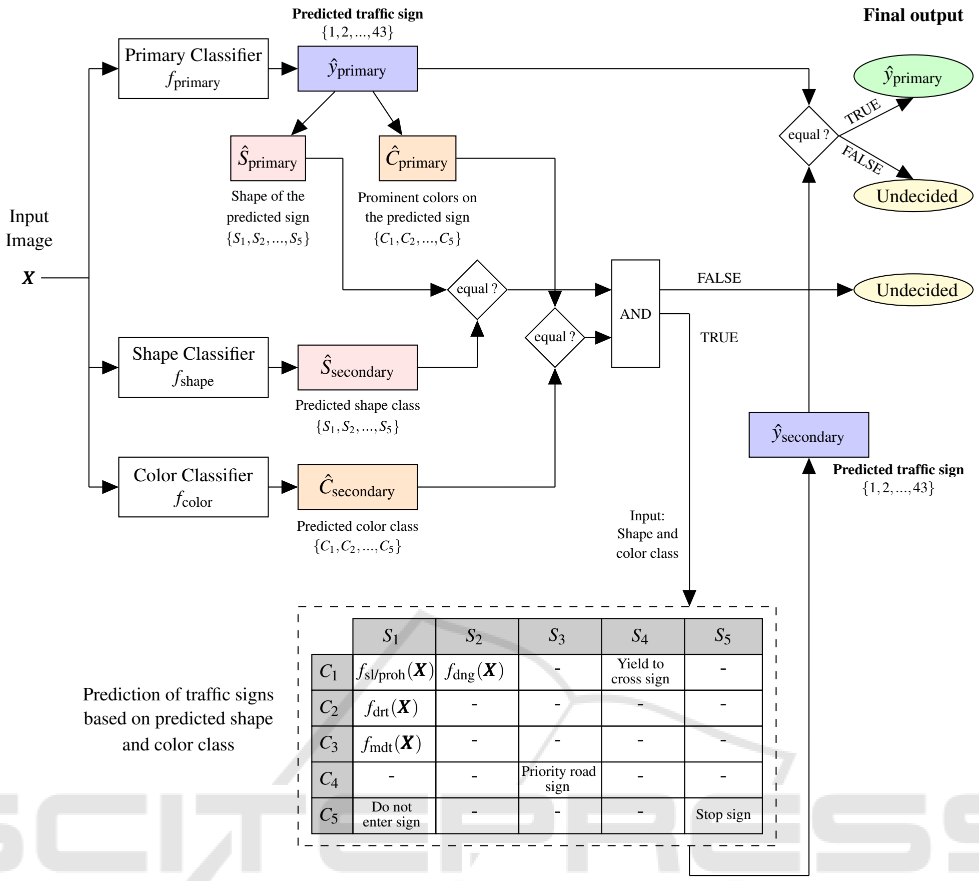
- Derestriction signs classifier f_{drt} : $\boldsymbol{X} \in \mathbb{R}^l \longrightarrow \hat{y}_{\text{drt}} \in \{1, 2, ..., 4\}$, and
- Mandatory signs classifier f_{mdt} : $\boldsymbol{X} \in \mathbb{R}^l \longrightarrow \hat{y}_{\text{mdt}} \in \{1, 2, ..., 8\}$.

Since the traffic signs: do not enter, priority road, yield to cross and stop, are independent in terms of their shape and color, we do not train any further secondary classifier(s). For instance, if f_{shape} and f_{color} predict $\hat{S}_{\text{secondary}} = 5$ (octagon) and $\hat{C}_{\text{secondary}} = 5$ (red background) for an input image $\boldsymbol{X}$, the prediction obtained by the network of secondary classifiers is *stop* sign. The traffic sign predicted upon the use of the secondary classifiers is finally mapped back into the corresponding original traffic sign label in $\{1, 2, ..., 43\}$, which is then considered as $\hat{y}_{\text{secondary}}$.

4.2.3 Merging the Decisions of the Classifiers

A schematic representation of the approach is provided in Figure 7. For a given input image $\boldsymbol{X}$, if the prediction made by the primary classifier f_{primary} is *20 speed limit* sign (i.e., $\hat{y}_{\text{primary}} = 1$), then by prior knowledge, we know that the shape of the sign board will be circular (i.e., $\hat{S}_{\text{primary}} = S_1$) and it will have a white background with red border (i.e., $\hat{C}_{\text{primary}} = C_1$). Now, the predictions $\hat{S}_{\text{secondary}}$ and $\hat{C}_{\text{secondary}}$, made by the secondary classifiers f_{shape} and f_{color}, are compared with $\hat{S}_{\text{primary}}$ and $\hat{C}_{\text{primary}}$, respectively. If either or both the conditions: $\hat{S}_{\text{primary}} = \hat{S}_{\text{secondary}}$ and $\hat{C}_{\text{primary}} = \hat{C}_{\text{secondary}}$ are *false*, then the final prediction is an undecided outcome, i.e., $\hat{y}_{\text{final}} = \phi$. On the contrary, if both the abovementioned conditions are *true* simultaneously, then based on the predicted shape and color class (i.e., $\hat{S}_{\text{primary}}/\hat{S}_{\text{secondary}}$ and $\hat{C}_{\text{primary}}/\hat{C}_{\text{secondary}}$), the traffic sign prediction $\hat{y}_{\text{secondary}}$ is derived from the matrix given in Figure 7. Again, the unanimity of the predictions $\hat{y}_{\text{primary}}$ and $\hat{y}_{\text{secondary}}$ is checked. As also shown in Equation 17, if both these predictions are the same, then the final prediction $\hat{y}_{\text{final}}$ for the input image $\boldsymbol{X}$ is $\hat{y}_{\text{primary}}$ (or $\hat{y}_{\text{secondary}}$). However, non-unanimity results in an undecided outcome, i.e., $\hat{y}_{\text{final}} = \phi$.

The effectiveness of this mechanism is remarked by it's ability to trigger an undecided outcome for all those test images for which the primary classifier make incorrect predictions. Nevertheless, the disadvantage of this strategy is that it might also trigger an undecided outcome for certain amount of test images for which the primary classifier already make correct predictions. Thus, a suitable threshold must be imposed to ensure that the number of test images for which $\hat{y}_{\text{final}} = \phi$ is not too high. However, defining this threshold is not within the scope of this paper.


Predicted traffic sign
{1,2,...,43}
Final output
Primary Classifier f_{primary}
$\hat{y}_{\text{primary}}$
$\hat{S}_{\text{primary}}$
$\hat{C}_{\text{primary}}$
Shape of the predicted sign {$S_1, S_2, ..., S_5$}
Prominent colors on the predicted sign {$C_1, C_2, ..., C_5$}
Input Image
$\boldsymbol{X}$
equal ?
TRUE
FALSE
Undecided
AND
FALSE
TRUE
Shape Classifier f_{shape}
$\hat{S}_{\text{secondary}}$
Predicted shape class {$S_1, S_2, ..., S_5$}
Color Classifier f_{color}
$\hat{C}_{\text{secondary}}$
Predicted color class {$C_1, C_2, ..., C_5$}
$\hat{y}_{\text{secondary}}$
Predicted traffic sign {1,2,...,43}
Input: Shape and color class
Prediction of traffic signs based on predicted shape and color class

	S_1	S_2	S_3	S_4	S_5
C_1	$f_{\text{sl/proh}}(\boldsymbol{X})$	$f_{\text{dng}}(\boldsymbol{X})$	-	Yield to cross sign	-
C_2	$f_{\text{drt}}(\boldsymbol{X})$	-	-	-	-
C_3	$f_{\text{mdt}}(\boldsymbol{X})$	-	-	-	-
C_4	-	-	Priority road sign	-	-
C_5	Do not enter sign	-	-	-	Stop sign

Figure 7: Prediction of the traffic sign class for the input image $\boldsymbol{X}$ by fusing the predictions of the primary classifier (f_{primary}) and the network of secondary classifiers (i.e., f_{shape}, f_{color}, $f_{\text{sl/proh}}$, f_{dng}, f_{drt} and f_{mdt}).

4.3 Experimentation

A total of 26640 images from the GTSRB training data were splitted into three equal subsets. All the images were rescaled to $l = 48 \times 48 \times 3$ pixels. The first subset of the data was used to train the primary classifier f_{primary}. The model f used for our analysis in Section 3.4 is f_{primary} in our experimentation here.

Half of the images in the second subset of the training data were used to train the shape classifier f_{shape}, while the other half were used to train the color classifier f_{color}. For training the classifiers f_{shape} and f_{color}, the images in their respective training data were first mapped from the traffic sign classes $\{1, 2, ..., 43\}$ into the corresponding shape $\{S_1, S_2, S_3, S_4, S_5\}$ and color $\{C_1, C_2, C_3, C_4, C_5\}$ classes, respectively. The shape and the color classes (shown in Figure 5) were, hence, used as labels for training f_{shape} and f_{color}, respectively. The images in the third subset belonging to the speed limit and prohibitory sign types were used to train the classifier $f_{\text{sl/proh}}$, and similarly, the other classifiers, i.e., $f_{\text{dng}}, f_{\text{drt}}$ and f_{mdt} were trained with the corresponding sign type's images in the third subset of the training data. Further experimentation details (e.g., DNN architecture, hyperparameters, etc.) related to the training and the evaluation of these classifier models are recorded in Appendix B. Note that the set of test images, i.e., $\mathcal{X}_{\text{test}}$, used for the evaluation here is the same as used in Section 3.4.3.

4.4 Results and Discussion

4.4.1 Evaluation of the Mechanism

Firstly, the individual accuracies of the classifier models are recorded in Table 1.

Let us denote the total number of test images used for the evaluation as τ. When we use the primary classifier alone (i.e., without the implementation of the mechanism), let us assume it misclassifies ρ number of images. The misclassification rate by f_{primary} is:

$$\bar{\rho} = \frac{\rho}{\tau} \times 100. \qquad (18)$$

Now, when we implement the mechanism, out of these τ test images, let us say that the mechanism

Table 1: Individual accuracies of the classifier models.

Classifier	# Test Images	Test Accuracy
f_{primary}	10000	96.12 %
f_{shape}	10000	98.87 %
f_{color}	10000	98.56 %
$f_{\text{sl/proh}}$	4497	98.2 %
f_{dng}	2207	95.7 %
f_{drt}	278	94.6 %
f_{mdt}	1409	97.3 %

yields an undecided outcome (i.e., $\hat{y}_{\text{final}} = \phi$) for λ images. Thus, the percentage of test images for which the mechanism does not yield any prediction is:

$$\bar{u} = \frac{\lambda}{\tau} \times 100. \tag{19}$$

The number of images for which $\hat{y}_{\text{final}} = \hat{y}_{\text{primary}}$ is $(\tau - \lambda)$. Let us assume that out of these $(\tau - \lambda)$ images, for ω images the final prediction (i.e., $\hat{y}_{\text{final}} = \hat{y}_{\text{primary}}$) is actually an incorrect prediction. Therefore, the misclassification rate after the implementation of the mechanism becomes:

$$\bar{\omega} = \frac{\omega}{\tau - \lambda} \times 100. \tag{20}$$

We use the mechanism presented in Figure 7 to obtain the final prediction ($\hat{y}_{\text{final}}$), for all the $\tau = 10000$ images in the set of original GTSRB test data, i.e., $\mathcal{X}_{\text{test}}$, as well as for the set of perturbed test images, i.e., $\mathcal{X}_{\Delta\text{contrast}}$, $\mathcal{X}_{\Delta\text{brightness}}$, $\mathcal{X}_{\Delta\text{blur}}$ and $\mathcal{X}_{\Delta\text{noise}}$. The obtained values of $\bar{\rho}$ (Equation 18), $\bar{u}$ (Equation 19) and $\bar{\omega}$ (Equation 20) are determined for each of these sets of test images. The results are presented in Figure 8. To realize the advantage of using the mechanism, we compare the misclassification rate $\bar{\omega}$ obtained after implementation of the mechanism with the misclassification rate $\bar{\rho}$ observed when using f_{primary} alone. In Figure 8, consider the results corresponding to $\mathcal{X}_{\text{test}}$. The use of f_{primary} alone results in a misclassification rate of $\bar{\rho} = 3.88\%$. However, when the mechanism is used, for $\bar{u} = 5.61\%$ of images in $\mathcal{X}_{\text{test}}$, no decision is produced. Among the remaining images, i.e., for which a decision is produced ($\hat{y}_{\text{final}} = \hat{y}_{\text{primary}}$), the misclassification rate is just $\bar{\omega} = 0.69\%$. A substantial drop in the misclassification rate suggests that the mechanism controls the misclassifications incurred by f_{primary} considerably well. Similar conclusion can be drawn from the results obtained for the set of perturbed test images.

4.4.2 Validation of the Arguments 1 and 2

For the test data $\mathcal{X}_{\text{test}}$, $\mathcal{X}_{\Delta\text{contrast}}$, $\mathcal{X}_{\Delta\text{brightness}}$, $\mathcal{X}_{\Delta\text{blur}}$ and $\mathcal{X}_{\Delta\text{noise}}$, the number of images n_{low}, n_{moderate}, and n_{high} misclassified by f_{primary} are already shown in Figure 3. For instance, consider the set of the original (unperturbed) test images, i.e., $\mathcal{X}_{\text{test}}$. The values are $n_{\text{low}} = 1$, $n_{\text{moderate}} = 103$, and $n_{\text{high}} = 284$ (Figure 3a). As discussed in Section 4.1, we expect the mechanism in Figure 7 to yield $\hat{y}_{\text{final}} = \phi$ (undecided outcome), ideally, for all these images misclassified by f_{primary}. Upon investigation, we observed that the mechanism does yield the undecided outcome for the $n_{\text{low}} = 1$ image, and hence, $\bar{u}_{\text{low}} = 100\%$. The term $\bar{u}_{\text{low}}$ signifies the percentage of n_{low} images for which the mechanism does not yield any decision. Out of $n_{\text{moderate}} = 103$ images, the mechanism yields the undecided outcome for 97 images, i.e., 94.17% of n_{moderate} images. Therefore, $\bar{u}_{\text{moderate}} = 94.17\%$. However, out of $n_{\text{high}} = 284$ images, the mechanism yields the undecided outcome for only 225 images, i.e., 79.23% of n_{high} images, therefore, $\bar{u}_{\text{high}} = 79.23\%$. We continue this investigation for the set of perturbed test images. The results are presented in Table 2. It is observed that for all the n_{low} number of misclassified images (i.e., for which the corresponding misclassifications lie in $\mathcal{M}_{\text{low}}$), the mechanism yields the undecided out-

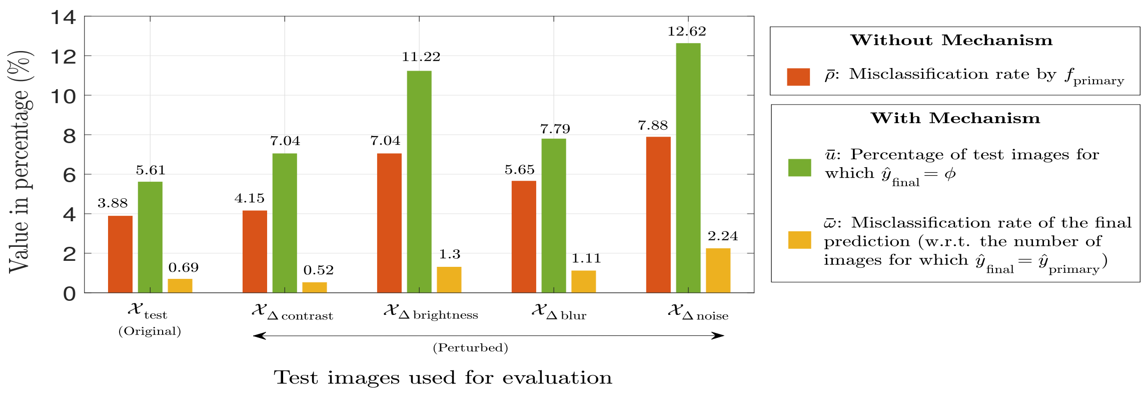


Figure 8: Experimental results: Evaluation of the mechanism.

Table 2: Experimental results: Validation of the arguments.

Test Images			For the misclassification(s) belonging to:					
			$\mathcal{M}_{\text{low}}$		$\mathcal{M}_{\text{moderate}}$		$\mathcal{M}_{\text{high}}$	
			n_{low}	$\bar{u}_{\text{low}}$	n_{moderate}	$\bar{u}_{\text{moderate}}$	n_{high}	$\bar{u}_{\text{high}}$
	Original	$\mathcal{X}_{\text{test}}$	1	100%	103	94.17%	284	79.23%
Perturbed	Contrast change	$\mathcal{X}_{\Delta\text{contrast}}$	4	100%	112	95.54%	299	85.62%
	Brightness change	$\mathcal{X}_{\Delta\text{brightness}}$	15	100%	156	91.03%	533	81.05%
	Gaussian blur	$\mathcal{X}_{\Delta\text{blur}}$	2	100%	138	89.13%	425	79.53%
	Gaussian noise	$\mathcal{X}_{\Delta\text{noise}}$	27	100%	197	88.32%	564	69.33%

come. On the other hand, it is relatively difficult to achieve the same for all the n_{high} number of misclassified images (i.e., for which the corresponding misclassifications lie in $\mathcal{M}_{\text{high}}$). It can be inferred that the misclassifications that were ranked to be offering the primary classifier model a high vulnerability (based on our categorisation criteria) are relatively difficult to control, even after the implementation of the additional efforts to minimize the misclassification rate. One can expect the amount of rigor required to control a particular misclassification incurred by f_{primary} to be considerably higher as it's vulnerability to the misclassification increases. The investigation supports our Arguments 1 and 2 specified in Section 4.1.

5 CONCLUSIONS

In this paper, we proposed an approach to estimate how vulnerable is a trained DNN model to any particular misclassification. It is based on estimating the DNN's vulnerability to misclassification of an input image belonging to a class k_1 into an incorrect class k_2 by measuring the similarity learnt by the trained model between the classes k_1 and k_2. We illustrated experimentally that the majority of the test images that are misclassified by the model encounter the misclassifications to which it's vulnerability is categorised as high. This provides a rationale to our proposed approach and also justifies its potentiality to identify the set of critical misclassifications that the DNN model is more likely to incur during the operation. Based on the acquired knowledge, pertinent measures or counter strategies can be developed and integrated to curb the in-operation likelihood of these critical misclassifications. Our further empirical investigation was to validate the argument that the amount of rigor required to deal with these critical misclassifications is relatively higher than what is required to deal with the misclassifications to which the DNN model is moderately or lowly vulnerable. As an extension to this work, we wish to study how the knowledge acquired by the application of this proposed concept be further utilized for the preparation of the consequent steps to enhance the robustness and/or performance of the image classification functionality in highly automated driving.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their valuable comments and feedback.

REFERENCES

Agarwal, H., Dorociak, R., and Rettberg, A. (2020). A strategy for developing a pair of diverse deep learning classifiers for use in a 2-classifier system. In *2020 X Brazilian Symposium on Computing Systems Engineering (SBESC)*, pages 1–8. IEEE.

Agarwal, H., Dorociak, R., and Rettberg, A. (2021). On developing a safety assurance for deep learning based image classification in highly automated driving. In *2021 Design, Automation & Test in Europe Conference (DATE)*.

Chollet, F. et al. (2015). Keras. https://keras.io.

Gao, X. W., Podladchikova, L., Shaposhnikov, D., Hong, K., and Shevtsova, N. (2006). Recognition of traffic signs based on their colour and shape features extracted using human vision models. *Journal of Visual Communication and Image Representation*, 17(4):675–685.

Grigorescu, S. M., Trasnea, B., Cocias, T., and Macesanu, G. (2019). A survey of deep learning techniques for autonomous driving. *arXiv preprint arXiv:1910.07738*.

Harris, C. R. et al. (2020). Array programming with NumPy. *Nature*, 585:357–362.

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

Kashyap, R. (2019). The perfect marriage and much more: Combining dimension reduction, distance measures and covariance. *Physica A: Statistical Mechanics and its Applications*, 536:120938.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105. Curran Associates, Inc.

Muhammad, K., Ullah, A., Lloret, J., Del Ser, J., and de Albuquerque, V. H. C. (2020). Deep learning for safe autonomous driving: Current challenges and future directions. *IEEE Transactions on Intelligent Transportation Systems*.

Ouyang, W., Zhou, H., Li, H., Li, Q., Yan, J., and Wang, X. (2017). Jointly learning deep features, deformable parts, occlusion and classification for pedestrian detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(8):1874–1887.

Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *arXiv preprint arXiv:1602.04938*.

SAE International (2014). Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems (Standard No. J3016).

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2014). Overfeat integrated recognition, localization and detection using convolutional networks. In *2nd International Conference on Learning Representations, ICLR 2014*.

Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332.

Tian, Y., Zhong, Z., Ordonez, V., Kaiser, G., and Ray, B. (2020). Testing DNN image classifiers for confusion & bias errors. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, pages 1122–1134.

Virtanen, P. et al. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Zhu, Y., Zhang, C., Zhou, D., Wang, X., Bai, X., and Liu, W. (2016). Traffic sign detection and recognition using fully convolutional network guided proposals. *Neurocomputing*, 214:758–766.

APPENDIX

A Primary Classifier

All the images from the GTSRB dataset were rescaled to $48 \times 48 \times 3$ and normalized so as to keep the pixel values in the range of 0 to 1. For the experimentation details corresponding to the DNN model f (or f_{primary}), refer to the first row of Table 3.

B Secondary Classifiers

Table 3 also summarizes the details corresponding to f_{shape}, f_{color}, $f_{\text{sl/proh}}$, f_{drt}, f_{dng} and f_{mdt}. From the validation (and the test) data used for evaluating f_{primary}, f_{shape} and f_{color}, only the images belonging to the sign type: speed limit/prohibitory, derestriction, danger and mandatory were used as the validation (and the test) data for the corresponding sign type classifiers, i.e., $f_{\text{sl/proh}}$, f_{dng}, f_{drt} and f_{mdt}, respectively.

Table 3: Experimentation details corresponding to the different classifier models.

Classifier[a]	Brief Summary of the DNN Architecture[b]	Hyperparameters[c]	Number of Images		
			Train	Validation	Test
f or f_{primary}	6 convolutional, 4 max pooling, 2 dense and 7 dropout layers	$bs = 64$, $\alpha = 10^{-4}$, $\varphi = 10^{-4}$	8880	2630	10000
f_{shape}	3 convolutional, 3 max pooling, 2 dense and 4 dropout layers	$bs = 32$, $\alpha = 10^{-4}$, $\varphi = 10^{-4}$	4440	2630	10000
f_{color}	3 convolutional, 3 max pooling, 2 dense and 4 dropout layers	$bs = 32$, $\alpha = 10^{-4}$, $\varphi = 10^{-4}$	4440	2630	10000
$f_{\text{sl/proh}}$	8 convolutional, 4 max pooling, 4 dense and 8 dropout layers	$bs = 16$, $\alpha = 10^{-4}$, $\varphi = 10^{-4}$	3890	1173	4497
f_{dng}	8 convolutional, 4 max pooling, 4 dense and 12 dropout layers	$bs = 16$, $\alpha = 10^{-3}$, $\varphi = 10^{-4}$	2050	583	2207
f_{drt}	8 convolutional, 4 max pooling, 4 dense and 8 dropout layers	$bs = 16$, $\alpha = 10^{-3}$, $\varphi = 10^{-4}$	280	82	278
f_{mdt}	8 convolutional, 4 max pooling, 4 dense and 8 dropout layers	$bs = 16$, $\alpha = 10^{-4}$, $\varphi = 10^{-4}$	1280	361	1409

[a] Every classifier was trained for 30 epochs using the standard categorical cross entropy loss function. We chose the model obtained from the training epoch at which the highest validation accuracy was observed.

[b] The DNN architecture yields two outputs: logits and output from the final softmax activation layer.

[c] bs, α and φ denotes the batch size, the learning rate, and the learning rate decay, respectively.