Safety-Aware Weighted Voting for N-version Traffic Sign Recognition System

Linyun Gao, Qiang Wen, Fumio Machida
Department of Computer Science
University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan
gao.linyun@sd.cs.tsukuba.ac.jp, wen.qiang@sd.cs.tsukuba.ac.jp, machida@cs.tsukuba.ac.jp

***Abstract*—The N-Version Machine Learning (NVML) system is an approach to improving the reliability of system outputs by employing several different machine learning models in the same system. Voting mechanisms are crucial in NVML systems, influencing the final decision-making process. This paper investigates voting mechanisms for the NVML system by introducing a safety metric based on the Failure Modes and Effects Analysis (FMEA) method. Safety-related weights are assigned to machine learning models in the NVML system to implement weighted voting and weighted soft voting mechanisms. As a case study, we investigate a traffic sign recognition system. Through the FMEA analysis, we categorize the misclassifications of traffic signs based on their severity. The safety metric is defined by the severity with the misclassification probability estimated from the test results and is used for assigning weights to machine learning models. Our experimental results on a real traffic sign dataset show the advantage of safety-aware weighted soft voting in all safety evaluation metrics. Moreover, we use a Large Language Model (LLM) to generate the weights for the voting mechanisms. However, the preliminary results show that the LLM-based approach yields a suboptimal solution compared to our weight assignment method.**

***Index Terms*—system safety, machine learning, N-version programming, autonomous vehicle**

I. Introduction

In recent years, Machine Learning (ML) technologies have been widely deployed in safety-critical applications like autonomous vehicles. Computer Vision (CV) systems powered by ML algorithms are becoming essential components of autonomous vehicles [1]. Most state-of-the-art CV systems are constructed based on unexplainable black-box ML models like deep neural networks. Therefore, the reliability of CV systems remains a challenge. As uncertain or mispredicted outputs of ML models may cause serious accidents in safety-critical systems, there is an urgent need for methods to improve the reliability of ML systems.

The N-version Machine Learning (NVML) system leverages the traditional N-version programming technique by using multiple different ML models in the same system to improve the reliability of the system output [2]. Recent studies have demonstrated reliability improvements by the NVML architecture in image classification tasks [3] and steering control tasks [4]. During the operation of an NVML system, different ML models may output different predictions. To determine reliable system outputs from multiple prediction results, the NVML system needs a prudent voting mechanism. For instance, in a three-version ML image classification system, three ML models give three independent predictions, and the final output is determined by majority voting [5].

However, the majority voting mechanism has limitations. One significant drawback is its inability to provide an output when each model makes a completely different prediction, leading to a tie with no clear majority. For instance, if the three models predict classes A, B, and C, the majority voting system fails to determine a definitive outcome. Additionally, for cases that are prone to misclassification, relying solely on majority voting might not yield the best result. Instead, it may be more advantageous to consider the prediction from the model with the highest historical accuracy, as it is more likely to deliver a correct result for difficult-to-classify instances.

To address these challenges, we investigate more fine-grained voting mechanisms for NVML systems for safety-critical applications. Our exploration includes various voting mechanisms based on NVML image classification systems, such as majority voting, weighted voting, soft voting, and weighted soft voting. In this study, we particularly focus on the weighted soft voting mechanism, as it utilizes the confidence levels of the models' predictions and assigns appropriate weights to each model. This approach allows the system to account for both the confidence and reliability of different models, integrating nuanced information to achieve potentially more accurate and reliable outputs in safety-critical applications.

To determine safety-aware weights in a safety-critical application context, we develop a metric to assess the safety of an ML-based traffic sign recognition system for autonomous vehicles. We employ the Failure Mode and Effects Analysis (FMEA) method to systematically analyze potential failures that could occur due to errors in the traffic sign recognition system during autonomous vehicle operation. By identifying and evaluating these potential failure modes, we assign risk scores based on their severity and likelihood. These risk scores inform the assignment of safety-aware weights to the models, ensuring that models with lower associated risks have a greater influence on the system's final output.

The effectiveness of various voting mechanisms, including weighted soft voting, is evaluated through experiments using a traffic sign recognition system composed of three distinct ML

models: AlexNet, VGG16, and EfficientNetB0. We measure the performance of the system with metrics such as accuracy, safety score, and the count of high-severity misclassifications. These metrics are used to compare the effectiveness of different voting mechanisms. The results show that our proposed weighted soft voting method not only enhances accuracy but also significantly improves the safety and reliability of the NVML system by reducing the number of severe misclassification instances.

Moreover, we introduced LLM-based weight assignment approach in which weights are generated by ChatGPT. We ask ChatGPT to assign the weight for each model based on the severity matrix and misclassification probabilities. However, our preliminary evaluation results show that the LLM-based weight assignment generates a suboptimal solution, suggesting needs for more specialized LLM and/or better prompt engineering.

Our contributions can be summarized as follows:

- We introduce a safety-aware weighted soft voting mechanism for NVML systems.
- We develop a safety metric based on the FMEA method to evaluate and improve the safety of ML-based traffic sign recognition systems.
- We demonstrate the effectiveness of the proposed safety-aware weighted voting mechanism through the experiments using a real traffic sign dataset.
- We examine the potential LLM-based approach for weight assignment.

The rest of the paper is organized as follows: Section II reviews related work. Section III provides background information on the traffic sign recognition and FMEA approach. Section IV describes the proposed voting mechanisms. Section V presents our safety metric. Section VI presents the experimental setup and results, and Section VII concludes the paper.

II. Related Work

Recent studies have explored multi-version ML approaches to improve ML system reliability. Xu et al. investigated the feasibility of developing fault-tolerant deep learning systems through model redundancy (i.e., NV-DNN [3]). They proposed several independent factors that can be used to generate multiple versions of neural network models, including training, network, and training data. Experimental results validate that their approach can improve the fault tolerance of deep learning systems. NV-DNN assumes that a single input is processed at one time, while N-version ML can use different inputs to leverage input diversity. Makino et al. [6] developed queueing models for multi-model multi-input ML systems to examine the performance of multi-version ML systems. Their research focuses on the throughput of straightforward two-version architectures, where the system can employ up to two different ML models and two data sources. Wen et al. [5], [7] conducted numerical and empirical analysis on the impact of using diverse models and varied inputs to enhance the reliability of three-version ML systems. Hong et al. [8] introduced a multimodal deep learning method that enhances the classification accuracy of remote-sensing imagery, surpassing the performance of both single-model or single-modality approaches.

Various voting schemes have been considered in multi-version ML approaches. Singamsetty et al. [9] examined existing weighted average voting algorithms used in safety-critical systems and introduced a history-based weighted voting algorithm with a soft dynamic threshold. Karimi et al. [10] introduced a voting algorithm for real-time fault-tolerant control systems, especially for large N. The algorithm overcomes the limitations of algorithms like median and weighted voting, significantly enhancing system reliability and availability. Wu et al. [4] developed a weighted N-version programming scheme for ensuring the resilience of ML-based steering control algorithms. The design of the scheme is based on the fusion of three redundant DNN model outputs. They proposed a weighted voting scheme based on the steering angle RMSE performance. However, in this study, we consider the severity of misclassification and explore how safety-aware weight voting affects the performance of NVML traffic sign classification systems.

Existing studies have investigated safety-aware strategies to enhance the reliability and safety of systems in critical applications. Zhao et al. [11] analyzed safety-aware computing system design in autonomous vehicles. They proposed a safety score metric to better assess safety beyond traditional performance metrics and introduced a perception latency model to estimate safety scores and demonstrate its application in managing hardware resources for enhanced safety in AV computing systems. Rahman et al. [12] investigated how the transient hardware faults contribute to the misclassification of DNN models based on safety-critical metrics compared to by intrinsic algorithmic inaccuracy. In this work, we introduce the safety-aware weighted voting N-version traffic sign recognition system to improve the safety of ML-based systems for autonomous vehicles.

III. Background

A. Traffic Sign Recognition

Traffic sign recognition is a crucial component of autonomous vehicles. It enables vehicles to recognize and respond to traffic signs such as speed limits, warnings, and prohibitions, ensuring adherence to road regulations and enhancing driving safety. These systems typically use ML algorithms to process visual data captured by cameras mounted on vehicles, allowing them to recognize traffic signs in real-time.

Accurate recognition of traffic signs is essential for safe navigation and compliance with traffic laws. Misclassification of traffic signs can potentially lead to accidents. For instance, misclassifying a ”Speed Limit 20” sign as a ”Speed Limit 70” sign (see Fig. 1a) can result in dangerously high speeds. In contrast, some misclassifications are relatively safe. For example, misclassifying a ”School Crossing” sign as a ”Cycles Crossing” sign (see Fig. 1b) typically does not pose a significant safety threat, as both indicate caution. These

(a) (b)

Fig. 1: Misclassifications of traffic signs: (a) Speed Limit 20 sign misclassified as Speed Limit 70 (unsafe misclassification example) and (b) School Crossing sign misclassified as Cycles Crossing (safe misclassification example)

examples underscore the importance of high safety in traffic sign recognition systems, particularly in safety-critical contexts. Our experiments with models like AlexNet demonstrate both critical and non-critical misclassifications, highlighting the need to improve the reliability and safety of traffic sign recognition in autonomous vehicles.

B. Failure Modes and Effects Analysis

FMEA is a systematic approach used to identify potential failure modes within a system and assess their potential effects [13]. It is widely utilized across various industries, including engineering, manufacturing, and healthcare, to prioritize and mitigate risks associated with system failures before they occur.

In this study, we focus on enhancing the safety of ML-based CV systems in autonomous vehicles. These systems must accurately detect and recognize objects such as vehicles, pedestrians, traffic lights, and traffic signs to ensure safe driving. Our research specifically targets traffic sign recognition systems, which are crucial for maintaining road safety.

We employ FMEA to evaluate the safety risks associated with the misclassification of traffic signs in different driving contexts. For example, if an autonomous vehicle misrecognizes a 60 km/h speed limit sign as 120 km/h, it could result in serious accidents due to excessive speeding. FMEA enables us to systematically assess these risks and prioritize improvements to the recognition system.

The FMEA process involves the following steps:

1) FMEA Worksheet Creation: The first step is to create a worksheet that lists all the functions of the analysis objects, along with their potential failures and the effects these failures might cause.

2) Effects Analysis: Next, we determine the Probability (P), Severity (S), and Detection (D) levels for each potential failure:

- **Probability** (P): The likelihood of a failure occurring.
- **Severity** (S): The impact or consequences of the failure.
- **Detection** (D): The ability to detect the failure before it occurs.

In our study, the traffic sign recognition system in autonomous vehicles makes real-time predictions based on images captured by onboard cameras. Therefore, we do not consider the detection factor in this paper.

3) Risk Levels Calculation: Finally, we calculate the Risk level ($R = P \times S$), which helps us identify which failures have the highest priority for remediation. This prioritization enables us to address the most critical issues first and enhance the overall safety of the traffic sign recognition system.

IV. Voting Mechanisms

In the NVML system, there is a decision module that aggregates the outputs from various ML models and determine the final system output. This system integrates multiple image classification models. Typically, a multi-category image classification model provides a single category label with the highest probability. However, in NVML systems, the output of each ML model is not directly used as the final system output. Instead, the output of the softmax layer can be utilized as the ML model output, which allows for a more nuanced and probabilistic representation of the classifications.

We explain several voting mechanisms that can be considered for the NVML image classification system. Based on the different forms of the ML model outputs, we divide the decision-making mechanisms into two kinds: hard voting and soft voting.

A. Hard Voting

In hard voting, suppose we are given a set of T individual ML models $\{h_1, h_2, \cdots, h_T\}$ and a set of l possible class labels $\{c_1, c_2, \cdots, c_l\}$. It is generally assumed that for an input x, the output of model h_i is given as an l-dimensional vector

$$h_i(x) = \left(h_i^1(x), h_i^2(x), \cdots, h_i^l(x)\right)^T$$

where $h_i^j(x) \in \{0, 1\}$. In $h_i(x)$, only one element $h_i^j(x) = 1$, indicating that the model h_i predicts the input x as the class label c_j. All other elements $h_i^k(x)$ for $k \neq j$ will be 0. In hard voting, we have two types: majority / plurality voting and weighted voting.

1) Majority / Plurality Voting: In majority or plurality voting, each ML model votes for one class, and the final output is the class that receives the most votes. Majority voting requires that the received votes be more than half. For example, if the output class labels of 5 ML models are $(1, 1, 2, 3, 4)$, it is rejected by majority voting while label 1 is chosen by plurality voting. The final output of majority voting is given by:

$$H_m(x) = \begin{cases} c_j & \text{if } \sum_{i=1}^{T} h_i^j(x) > \frac{1}{2} \sum_{k=1}^{l} \sum_{i=1}^{T} h_i^k(x), \\ \text{rejection} & \text{otherwise.} \end{cases} \quad (1)$$

On the other hand, the final output of plurality voting is given by:

$$H_p(x) = c_k, \quad k = \arg\max_j \sum_{i=1}^{T} h_i^j(x). \quad (2)$$

Plurality voting performs the same as majority voting when the number of models is less than or equal to three. However,

majority voting alone cannot handle all situations, such as when the output class labels are $(1, 2, 3)$ in a 3-version system and $(1, 1, 2, 2, 3)$ in a 5-version system. In these cases, additional parameters are required for decision-making. In the later sections, we construct a 3-version ML system for the experiment, so we will only discuss majority voting.

Majority and plurality voting schemes are easy to understand and implement. However, since these voting schemes do not consider the differences among voters, minority voters tend to be neglected. Moreover, plurality voting does not require a majority (more than 50%) which can lead to an output that is not widely agreed upon.

2) Weighted Voting: The weight voting assigns a weight to each ML model. The final output is the class that receives the most votes. The final output of weighted voting is given by:

$$H_w(x) = c_k, \quad k = \arg\max_j \sum_{i=1}^{T} w_i h_i^j(x), \tag{3}$$

where w_i is the weight assigned to the model h_i. The weights are normalized and constrained by $w_i > 0$ and $\sum_{i=1}^{T} w_i = 1$.

The weighted voting can consider varying levels of importance among voters. However, determining appropriate weights is not straightforward and can be error-prone. Additionally, the weighted voting can lead to certain voters having disproportionate influence, depending on the assigned weights.

B. Soft Voting

In soft voting, suppose we are given a set of T individual ML models $\{g_1, g_2, \cdots, g_T\}$ and a set of l possible class labels $\{c_1, c_2, \cdots, c_l\}$. It is generally assumed that for an input x, the output of model g_i is given as an l-dimensional vector

$$g_i(x) = \left(g_i^1(x), g_i^2(x), \cdots, g_i^l(x)\right)^T$$

where $g_i^j(x) \in [0, 1]$, which can be regarded as an estimate of the posterior probability of the input x belonging to class c_j.

1) Simple Soft Voting: Simple soft voting involves summing the outputs of all the ML models to calculate the final probability output for making the final prediction. This method is used in [3]. The final output of simple soft voting is given by:

$$G_s(x) = c_k, \quad k = \arg\max_j \sum_{i=1}^{T} g_i^j(x). \tag{4}$$

The simple soft voting can incorporate the uncertainty or ambiguity of voters. Instead of using class labels, this scheme utilizes the prediction probabilities for all categories. However, the probability scores predicted by the model do not always positively correlate with the confidence level. Incorrect outputs may have high probabilities, while correct outputs may have low probabilities.

2) Weighted Soft Voting: In weighted soft voting, different weights are assigned to ML models to calculate the final output. Weighted soft voting is used in [4]. The final output of weighted soft voting is given by:

$$G_w(x) = c_k, \quad k = \arg\max_j \sum_{i=1}^{T} w_i g_i^j(x), \tag{5}$$

where w_i is the weight assigned to the model g_i. The weighted soft voting combines the benefits of weighted voting and simple soft voting, incorporating varying voter weights and degrees of certainty in the votes. However, like weighted voting, the determination of weights might be complex.

C. Issue of weight assignment

Considering the advantages and disadvantages of the voting mechanisms, we employ the weighted soft voting in our study. This scheme combines the benefits of weighted voting, which considers different voter influences, and soft voting, which accounts for the level of certainty in the votes. However, there is no standard method for defining the weights. This motivates us to investigate how to assign relevant weights to different ML models to improve the safety of the system. To this end, we consider a safety metric that can be used for evaluating system safety and assigning weights to ML models in an NVML system.

V. Safety Metric

This section considers a safety metric for CV systems used in autonomous vehicles, specifically focusing on traffic sign recognition. We aim to use this safety metric to assign weights for weighted soft voting in an NVML system. To evaluate the safety of an autonomous vehicle, we first conduct a FMEA worksheet to enumerate the errors of the traffic sign recognition systems. We tabulate the different misclassifications in traffic sign recognition and determine their severity. Next, we estimate the probability of occurrence of various errors through experiments on the test dataset and calculate the risk score of the traffic sign recognition system.

A. FMEA worksheet

First, we conduct FMEA to analyze the failures caused by a traffic sign recognition system during the movement of autonomous vehicles. With the help of FMEA, we can create an FMEA worksheet in TABLE I that lists all the failure modes that can occur with the traffic sign recognition system and then derive the reasons why each error occurs and the consequences that are induced.

B. Effects Analysis

This paper analyzes the effect of failure mode No.1, misclassifying a traffic sign from class A to class B. The effect of this misclassification depends on the ground truth class and the predicted class. First, we determine the severity of the failure mode, and then we present the statistical method for determining the probability.

TABLE I: FMEA Worksheet for Traffic Sign Recognition System

Item	Num.	Failure Mode	Cause	Effect
Traffic Sign Recognition System	1	Misclassify a traffic sign from category A to B	Error in the ML algorithm; insufficient training data; ambiguous traffic sign	Misrecognition of traffic signs potentially leads to inappropriate vehicle actions such as sudden braking or accelerating depending on categories A and B.
	2	No output	Error in the ML algorithm; software crash	The vehicle may fail to respond to critical traffic signs, leading to unsafe driving depending on the unrecognized category.
	3	Real-time delay	High computational load; inefficient algorithms; hardware limitations	Delayed vehicle response to traffic signs increase the risk of accidents like rear-end collisions due to slower reaction times.
	4	Hardware failure	Sensor malfunction; power issues; physical damage	Loss of input data or system disablement results in the inability to recognize traffic signs, affecting vehicle response.
	5	Security attacks	Cyber attacks; unauthorized access; data tampering	Manipulation or obstruction of traffic sign data leads to incorrect system outputs and unsafe vehicle actions.
	6	Adversarial sample misclassification	Intentionally crafted inputs designed to deceive the ML model	Incorrect traffic sign recognition causes the vehicle to perform unsafe actions based on false information.
	7	No input	Camera obstruction; disconnection; environmental factors (e.g., fog)	Inability to recognize traffic signs leads to unresponsive vehicle actions.
	8	Wake-up signal delay	Sensor malfunction; bandwidth limitations; error in the vehicle system; prioritization issues	Delay in waking up the system results in the vehicle ignoring critical traffic signs.

TABLE II: An example of severity levels for traffic sign classification

Ground Truth	Prediction				
	Speed limit 60	Speed limit 120	Stop	Danger	Go right
Speed limit 60	-	2	2	1	1
Speed limit 120	2	-	2	1	1
Stop	2	2	-	0	0
Danger	1	1	0	-	0
Go right	0	0	0	0	-

1) Severity: Considering various misclassification conditions between the predicted class and the ground truth, we define the severity levels of this failure mode in a severity matrix. The severity level ranges from 0 to 2. Level 0 indicates that the misclassification poses no danger at all (e.g., misclassifying a "pedestrian crossing" sign to a "school crossing" sign does not lead to a dangerous situation for an autonomous vehicle). Level 1 indicates a negative influence that is unlikely to cause an accident. Level 2 indicates that the misclassification can lead to serious accidents (e.g., misclassifying a "speed limit 30" sign to a "speed limit 80" sign can cause the vehicle to accelerate to an unsafe speed, potentially leading to a serious accident). The entry '-' represents a correct prediction by the system.

We consider the traffic sign classification task based on the GTSRB dataset [14]. We determine the severity levels for every possible misclassification. Some representative traffic signs and their severity levels of misclassifications are demonstrated in TABLE II, while the complete list of severity levels of GTSRB is presented in the appendix.

2) Probability: We estimate the probability of misclassification and derive the probability matrix through experiments. For this experiment, we test a deep neural network model using the test dataset of GTSRB and record the experimental results. We used a confusion matrix (C) to record the results of the experiment, which is a $n \times n$ matrix (n is the number of the classes in the test dataset). The element C_{ij} represents the number of instances of class i that are predicted as class j. Based on the confusion matrix, we can derive a probability matrix (P). The element P_{ij} represents the probability of classifying an instance of class i as class j. When $i \neq j$, it indicates a misclassification. The probability is calculated as:

$$P_{ij} = \frac{C_{ij}}{\sum_{k=1}^{n} \sum_{l=1}^{n} C_{kl}}. \tag{6}$$

C. Risk Levels Calculation

First, we assign severity scores to different severity levels (sl) in the severity matrix (S), where higher severity scores indicate that the misclassification has a higher impact on safety. The element S_{ij} is defined as follows:

$$S_{ij} = \begin{cases} \sigma(sl), & i \neq j \\ 0, & i = j \end{cases} \tag{7}$$

where $\sigma(sl)$ is a monotonically increasing function that assigns the severity score according to the severity level. Based on the risk level calculation, we then multiply the elements in the severity matrix by the corresponding elements in the probability matrix to obtain the risk level matrix (R), given by:

$$R = S \odot P. \tag{8}$$

We use the risk level matrix to calculate a risk score. The risk score is a measure that reflects the overall risk associated with the ML model and system. A lower risk score indicates a safer model and system with respect to traffic sign misclassification. The calculation of the risk score is given by:

$$\text{Risk Score} = \sum_{i=1}^{n} \sum_{j=1}^{n} R_{ij}, \tag{9}$$

which is in the range of $[0, \sigma(2)]$.

D. Safety Score and Weight

We utilize the risk score to calculate a safety score. This safety score is used to evaluate the safety of individual ML models and the overall NVML system. A higher safety score indicates a safer ML system with respect to traffic sign recognition. The safety score is calculated as follows:

$$\text{Safety Score} = \frac{1}{1 + \text{Risk Score}}, \tag{10}$$

which is in the range of $(0, 1]$.

We use the risk scores $\{rs_1, rs_2, \cdots, rs_T\}$ of a set of T individual ML models to assign a safety-aware weight to each ML model for the weighted voting and the weight soft voting. the weight w_i assigned for model g_i is determined by:

$$w_i = \frac{\sum_{j=1}^{T} rs_j}{rs_i}. \tag{11}$$

We choose this weight because the value w_i is inversely proportional to the risk score rs_i, which ensure that the model exhibits a higher risk score is assigned a lower weight in the voting mechanism.

VI. Evaluation

We designed an experiment to evaluate the performance of voting mechanisms used in NVML systems for traffic sign recognition. The evaluation metrics include accuracy (Acc), safety score, and the number of misclassifications in severity level 2 (SL2) and severity level 1 (SL1) instances. We built a three-version traffic sign recognition system consisting of three different ML models: AlexNet [15], VGG16 [16], and EfficientNetB0 [17]. These models were selected for their ability to balance high accuracy and low computational requirements. Each model was trained independently using the same training set from GTSRB.

To assign safety-aware weights, we split the GTSRB test set, which contains 12,630 images, in half. One half was used for weight assignment, referred to as the "weight assignment set," and the other half was used for evaluating the ML models and the three-version traffic sign recognition system, referred to as the "evaluation set." In this experiment, the severity scores defined as $\sigma(0) = 0, \sigma(1) = 1, \sigma(2) = 10$, aiming at penalizing misclassifications in severity level 2.

A. Safety-aware Weight Assignment

1) FMEA Method: We used the trained models to make predictions on the weight assignment set, and assigned safety-aware weights to each model based on their performance. The prediction results and the assigned weights of the models are shown in TABLE III (The weights are normalized).

2) Large Language Model Generation: In recent times, large language models (LLMs) have demonstrated their applicability in both personal and business contexts across a wide range of fields. Leveraging their capabilities to assist with various tasks, LLMs have proven effective in enhancing both work processes and daily life. Recognizing this potential, we use an LLM, specifically GPT-4o, to generate benchmark weights for comparison with our weight determination method. The LLM was prompted using specific keywords related to NVML system and requirements for weight determination. We provided the severity matrix and confusion matrix that recorded the test results on the weight assignment set, in the form of CSV files. The example prompt we used is:

> "Determine the weights for models in the three-version machine learning system that consists of AlexNet, VGG16, and EfficientNetB0—the confusion matrix results from each model tested on the same evaluation dataset. The severity matrix represents the severity of every misclassification. The value in the confusion and severity matrix should be integers. The model produces misclassification with higher severity and probability should be assigned a lower weight. The weight value is in the range of 0 to 1."

The weights generated by LLM are shown in TABLE III. Note that the above prompt is one of the prompts we tried and achieved the best performance in our experiments.

TABLE III: Prediction Results and Assigned Weights of the Models

Model	Acc (%)	Safety (%)	Weight by FMEA	Weight by LLM
AlexNet	94.28	92.02	0.126	0.206
VGG16	97.64	96.52	0.301	0.375
EfficientNetB0	98.53	98.14	0.573	0.419

TABLE IV: Performance of ML Models

Model	Acc (%)	Safety (%)	SL2	SL1
AlexNet	94.17	91.26	45	155
VGG16	97.37	96.02	20	62
EfficientNetB0	**98.13**	**96.98**	**17**	**27**

B. Machine Learning Model Evaluation

We then tested the models using the evaluation set. The test results are presented in Table IV. As we can observe, EfficientNetB0 demonstrates the best performance across all evaluation metrics, with the highest accuracy and safety scores, and the lowest number of SL2 and SL1 misclassifications.

The results indicate a strong relationship between the safety score and the number of severe misclassifications. Models with fewer severe misclassifications tend to have higher safety scores, which reflects their ability to minimize potentially dangerous errors. For example, AlexNet, which had the most SL2 misclassifications, also had the lowest safety score among the models tested.

C. NVML System Evaluation

We then deploy different voting mechanisms in the three-version traffic sign recognition system. Each model receives the same inputs from the evaluation set and makes predictions independently. The voting mechanisms use the outputs of the models to determine the final system output. The results are presented in TABLE V.

TABLE V: Performance of Voting Mechanisms

Voting Mechanism	Acc (%)	Safety (%)	SL2	SL1
Majority	97.88	96.69	17	56
Weighted	97.13	96.96	16	27
Simple Soft	98.32	97.12	14	47
Weighted Soft	98.40	**97.92**	**11**	**24**
LLM Weighted	98.24	97.05	15	42
LLM Weighted Soft	**98.50**	97.35	13	42

The results demonstrate that weighted soft voting exhibit the best performance across all voting mechanisms tested. These methods enhance accuracy and safety scores, even outperforming the individual model EfficientNetB0 in safety evaluation metrics. The NVML system maintains good performance levels, even when including less accurate models such as AlexNet, indicating the robustness of the NVML system.

Furthermore, the NVML systems employing soft voting, weighted soft voting, LLM-based weight voting and weighted soft voting all show improvements in accuracy compared to the best-performing single model, EfficientNetB0. This suggests that the proposed weight assignment method effectively enhances safety and classification accuracy.

LLM-based weighted soft voting yield better safety improvements compared to majority voting and simple soft voting. However, the improvements are not as pronounced as those achieved by our proposed weight assignment methods. Although we tried several prompts with the same confusion matrices and the severity matrix, the results shown in TABLE V are the best results we obtained. To further improve the performance of the LLM-based weight assignment approach, we may need more guides for the usage of confusion matrices and the severity matrix to derive better safety-aware weights.

VII. Conclusion

This paper investigated voting mechanisms that can be employed in an NVML system for a safety critical application. We developed a safety metric based on the FMEA method to evaluate the safety of ML systems and assign safety-aware weights to both weighted voting and weighted soft voting mechanisms. Our study demonstrated that safety-aware weighted soft voting outperforms other voting mechanisms across all safety evaluation metrics. Notably, this approach not only enhances accuracy but also improves overall system safety by increasing the safety score and reducing the number of severe misclassification instances. The implementation of the safety metric provides a reliable method for assessing and enhancing the safety of ML systems in critical applications, such as autonomous driving. This approach enables the assignment of safety-aware weights, contributing to the development of safer autonomous driving technologies.

Additionally, we explored the possibility of using LLMs to determine safety-aware weights for the weighted voting mechanism. While this showed some improvement over majority and soft voting, the results were not as significant. In the future, we plan to refine our approach with better prompting techniques to achieve improved outcomes.

Acknowledgment

This work was supported by JST SPRING Grant Number JPMJSP2124, and partly supported by JSPS KAKENHI Grant Numbers 22K17871.

References

[1] E. Dilek and M. Dener, "Computer vision applications in intelligent transportation systems: a survey," *Sensors*, vol. 23, no. 6, p. 2938, 2023.

[2] F. Machida, "N-version machine learning models for safety critical systems," in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, 6 2019, pp. 48–51.

[3] H. Xu, Z. Chen, W. Wu, Z. Jin, S.-y. Kuo, and M. Lyu, "Nv-dnn: towards fault-tolerant dnn systems with n-version programming," in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, 2019, pp. 44–47.

[4] A. Wu, A. H. M. Rubaiyat, C. Anton, and H. Alemzadeh, "Model fusion: weighted n-version programming for resilient autonomous vehicle steering control," in *2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 2018, pp. 144–145.

[5] Q. Wen and F. Machida, "Reliability models and analysis for triple-model with triple-input machine learning systems," in *Proc. of the 5th IEEE Conference on Dependable and Secure Computing*, 2022, pp. 1–8.

[6] Y. Makino, T. Phung-Duc, and F. Machida, "A queueing analysis of multi-model multi-input machine learning systems," in *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, 2021, pp. 141–149.

[7] Q. Wen and F. Machida, "Characterizing reliability of three-version traffic sign classifier system through diversity metrics," in *Proc. of the 34th International Symposium on Software Reliability Engineering (ISSRE)*, 2023, pp. 333–343.

[8] D. Hong, L. Gao, N. Yokoya, J. Yao, J. Chanussot, Q. Du, and B. Zhang, "More diverse means better: Multimodal deep learning meets remote-sensing imagery classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 5, pp. 4340–4354, 2020.

[9] P. Singamsetty and S. Panchumarthy, "A novel history based weighted voting algorithm for safety critical systems," *Journal of Advances in Information Technology*, vol. 2, no. 3, pp. 139–145, 2011.

[10] A. Karimi, F. Zarafshan, S. Al-Haddad, and A. R. Ramli, "A novel n-input voting algorithm for x-by-wire fault-tolerant systems," *The Scientific World Journal*, vol. 2014, no. 1, p. 672832, 2014.

[11] H. Zhao, Y. Zhang, P. Meng, H. Shi, L. E. Li, T. Lou, and J. Zhao, "Towards safety-aware computing system design in autonomous vehicles," *arXiv preprint arXiv:1905.08453*, 2019.

[12] M. H. Rahman, S. Laskar, and G. Li, "Investigating the impact of transient hardware faults on deep learning neural network inference," *Software Testing, Verification and Reliability*, p. e1873, 2024.

[13] S. Khaiyum, B. Pal, and Y. Kumaraswamy, "An approach to utilize fmea for autonomous vehicles to forecast decision outcome," in *Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014: Volume 1*. Springer, 2015, pp. 701–709.

[14] [Online]. Available: https://benchmark.ini.rub.de/

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[17] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

Appendix

We define the severity levels for all possible misclassifications of traffic signs in GTSRB dataset. In TABLE VI, the ground truth traffic signs are listed in the first column, while the class number of predicted signs is presented at the top line. The severity level is either 0, 1, or 2, indicating that a higher severity level means a more dangerous misclassification.

TABLE VI: Severity level definitions for misclassifications of traffic signs in GTSRB

Ground Truth	Prediction																																														
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
0: Speed limit 20	-	1	2	2	2	2	1	2	2	1	1	1	1	1	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1: Speed limit 30	1	-	2	2	2	2	1	2	2	1	1	1	1	1	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2: Speed limit 50	2	1	-	1	2	2	1	2	2	1	1	1	1	1	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3: Speed limit 60	2	2	1	-	1	1	1	2	2	1	1	1	1	1	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4: Speed limit 70	2	2	1	1	-	1	1	2	2	1	1	1	1	1	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5: Speed limit 80	2	2	1	1	1	-	1	2	2	1	1	1	1	1	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6: Restriction ends 80	0	0	0	0	0	2	-	2	2	1	1	1	1	1	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
7: Speed limit 100	2	2	2	2	2	2	1	-	1	1	1	1	1	1	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8: Speed limit 120	2	2	2	2	2	2	1	1	-	1	1	1	1	1	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9: No overtaking	1	1	1	1	1	1	1	1	1	-	0	0	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
10: No overtaking (trucks)	1	1	1	1	1	1	1	1	1	0	-	0	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
11: Priority at next intersection	1	1	1	1	1	1	1	1	1	0	0	-	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
12: Priority road	1	1	1	1	1	1	1	1	1	0	0	0	-	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
13: Give way	1	1	1	1	1	1	1	1	1	0	0	0	0	-	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
14: Stop	2	2	2	2	2	2	2	2	2	1	1	1	1	1	-	1	1	0	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	0	0	0	0	0	1	1	1	1	1
15: No traffic both ways	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
16: No trucks	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	1	-	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
17: No entry	2	2	2	2	2	2	2	2	2	1	1	1	1	1	0	1	1	-	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	1	1	1	1	1
18: Danger	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	1	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
19: Bend left	1	1	1	1	1	1	0	1	1	0	0	0	0	0	1	1	0	1	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
20: Bend right	1	1	1	1	1	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
21: Bend	1	1	1	1	1	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
22: Uneven road	1	1	1	1	1	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
23: Slippery road	1	1	1	1	1	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
24: Road narrows	1	1	1	1	1	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
25: Construction	1	1	1	1	1	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
26: Traffic signal	1	1	1	1	1	1	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
27: Pedestrian crossing	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
28: School crossing	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	1	1	1	1	1
29: Cycles crossing	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	1	1	1	1	1
30: Snow	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	-	0	0	1	1	1	1	1	1	1	1	1	1
31: Animals	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	-	0	1	1	1	1	1	1	1	1	1	1
32: Restriction ends	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-	1	1	1	1	1	1	1	1	1	1
33: Go right	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	-	2	2	2	2	1	1	1	1	1
34: Go left	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	2	-	2	2	2	1	1	1	1	1
35: Go straight	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	2	2	-	2	2	1	1	1	1	1
36: Go right or straight	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	2	2	2	-	2	1	1	1	1	1
37: Go left or straight	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	2	2	2	2	-	1	1	1	1	1
38: Keep right	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-	1	1	1	1
39: Keep left	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-	1	1	1
40: Roundabout	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-	1	1
41: Restriction ends (overtaking)	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-	1
42: Restriction ends (overtaking (trucks))	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-