Leveraging Machine Learning for CRC Diagnostic Coverage Estimation

Taichi Emi Han Nay Aung Yasuhiro Yamasaki Hiroyuki Ohsaki
School of Engineering Kwansei Gakuin University,
E-mail: {taichi,han,yamasaki,ohsaki}@lsnl.jp

***Abstract*—This paper presents a novel machine learning-based approach to improve the accuracy of estimating the diagnostic coverage of Cyclic Redundancy Check (CRC) in automotive systems, crucial for adhering to functional safety standards like ISO 26262. By focusing on the prediction of Hamming weights — indicative of CRC's error detection capabilities — for large code and message lengths, our methodology addresses the limitations of traditional brute-force and approximate estimation methods. Utilizing supervised learning with known Hamming weights as training data, we evaluate the effectiveness of seven machine learning models, with LightGBM demonstrating superior performance after hyperparameter tuning and training data filtering. Our experiments show that this approach can estimate normalized Hamming weights with a relative error margin of approximately $\pm$ 15%, significantly enhancing the precision of CRC's diagnostic coverage estimation compared to conventional techniques.**

***Index Terms*—Cyclic Redundancy Check (CRC), diagnostic coverage, residual error rate, AUTOSAR (AUTomotive Open System ARchitecture), Hamming weight, machine learning**

I. Introduction

Ensuring the safety of automobiles is paramount in today's technology-driven world [1]. Functional safety standards, such as ISO 26262 [2, 3], serve as the cornerstone for identifying and mitigating risks that could potentially lead to hazardous situations in vehicles. AUTOSAR (AUTomotive Open System ARchitecture) [4] represents a pivotal initiative in standardizing software infrastructure across the automotive industry. It enables interoperability and flexibility in software design, allowing manufacturers to integrate advanced features and services seamlessly.

AUTOSAR E2E (End-to-End) protection [5] is a communication protection mechanism defined within the AUTOSAR framework. The E2E protection approach is particularly designed to detect errors that could occur during the transmission of messages in automotive networks, such as corruption, loss, repetition, insertion, or reordering of data packets. In AUTOSAR E2E, these fault modes are systematically addressed through the implementation of counters, timers, and Cyclic Redundancy Checks (CRC) [2, 3].

Among these three mechanisms, CRC is a fundamental and widely-deployed technique in digital communications used to detect errors in data transmission [6, 7]. It involves applying a polynomial division to the data bits, generating a checksum that reflects the data's integrity. This checksum, appended to the data, allows the receiving end to verify whether the data has been altered during transmission.

AUTOSAR E2E protection defines several *profiles*, each tailored to different requirements and use cases within automotive systems [5]. These profiles specify how messages are protected and checked, including the use of CRC of varying lengths, counter measures for message freshness, and data ID concepts for message identification. The choice of profile and CRC polynomial depends on the specific communication scenario, including the expected error patterns, performance requirements, and computational constraints.

Accurately understanding CRC's error detection capabilities is crucial for designing and evaluating the reliability of digital communication systems [8-10]. An accurate assessment of CRC's diagnostic coverage is essential for ensuring that automotive systems meet stringent safety standards, helping to prevent data corruption that could lead to system failures or hazardous situations.

Estimating the diagnostic coverage of CRC is fraught with challenges, primarily due to the complexity and variability of data patterns and the limitations of conventional analytical methods [11-13]. In [13], we proposed a technique for assessing the diagnostic coverage for a specific CRC polynomial using the upper- and the lower-bounds. This approach was designed to surpass the accuracy of the traditional diagnostic coverage estimation method, which primarily utilizes the Hamming distance alone. By doing so, we achieved a significant improvement in the precision of diagnostic coverage estimates.

Building on this foundation, this paper endeavors to further elevate the accuracy of estimating CRC's error detection ability through a machine learning-based approach. To the best of our knowledge, there have been no previous attempts to estimate the error detection capabilities of CRC using a machine learning approach.

This paper addresses the following research questions:

- Can machine learning techniques be effectively utilized to estimate the Hamming weights of CRC, a crucial parameter in evaluating its error detection capabilities?
- How can known Hamming weights across various message lengths and the number of errors from the same CRC polynomial be leveraged as training data to improve the precision of estimating unknown Hamming weights?
- Among the array of machine learning models and methods, which yields the highest accuracy in predicting Hamming weights, thereby facilitating a more precise assessment of CRC's diagnostic coverage?
- How accurately can the diagnostic coverage (error detection capability) of CRC be estimated by predicting its Hamming weights using machine learning, and what implications does this have for the reliability and functional safety of systems employing CRC?

This paper explores the application of machine learning techniques to estimate the error detection capabilities of CRC more accurately than traditional methods allow. By analyzing patterns in known error detection outcomes, machine learning models can uncover insights into the effectiveness of various CRC configurations under different conditions.

More specifically, this paper proposes the new method to enhances the precision in estimating unknown Hamming

weights by using known Hamming weights from the same CRC polynomial across different message lengths and the number of bit errors as training data. The process starts with transforming and filtering the Hamming weight data to optimize it for machine learning, followed by a comparative analysis of multiple models to select the most effective one. We then fine-tune the chosen model's hyperparameters to maximize its predictive accuracy.

The main contributions of this paper are summarized as follows.

- We demonstrate the feasibility and effectiveness of using machine learning techniques to estimate the Hamming weights of CRC, thereby paving the way for innovative approaches in assessing CRC's error detection capabilities.
- We propose a methodology that leverages known Hamming weights from the same CRC polynomial across various message lengths and Hamming distances as training data, significantly enhancing the precision of estimating unknown Hamming weights.
- We provide a detailed analysis of the predictive accuracy of different machine learning models for CRC's Hamming weight estimation and showcasing how this leads to a more accurate determination of CRC's diagnostic coverage.

This paper is organized as follows. Section II delves into the fundamentals of CRC and Hamming weight, crucial for understanding the basis of our investigation. Section III compares traditional diagnostic coverage estimation methods under a communication models with different bit error rates. In Section IV, we introduce a machine learning-based methodology for estimating Hamming weights, detailing the process and rationale behind using machine learning for improved accuracy in diagnostic coverage estimation. Section V presents our experimental setup, methodology, and results, offering a deep dive into the practical applications and implications of our research. Finally, Section VI concludes the paper by summarizing our findings, discussing their significance, and outlining future research directions.

II. Cyclic Redundancy Check and Hamming Weight

This section provides a brief introduction to CRC and Hamming weights. Additionally, we introduce the definition of the diagnostic coverage and the residual error rate.

Cyclic Redundancy Checks (CRC) is a method used to detect errors in digital data transmission. CRC appends a few bits, called a checksum, to the end of the bit string for a message and sends out the extended bit string [14]. The receiver recalculates the checksum upon receiving the data and compares it to the received checksum. If the result is 0, it indicates that no bits of the message were in error; if the result is not 0, then the receiver knows that there has been an error in one or more bits. However, if the result is 0, it does not necessarily mean there was no error. CRC cannot detect all possible errors, but the range of errors it can detect is impressively broad [14].

The *Hamming weight* [10, 12] in the context of error detection with CRC refers to the number of error patterns that cannot be detected by a particular CRC polynomial among all possible combinations of n-bit error patterns. Essentially, it quantifies the effectiveness of a CRC polynomial in detecting errors of a certain length within data.

To calculate the Hamming weight $\mathrm{HW}(L, n)$ for given message length L and the number of bit errors n, we need to explore all possible n-bit error patterns (i.e., ${}_L C_n$ combinations) for the given CRC polynomial [10]. When a message is divided by the CRC polynomial, the remainder represents the CRC value. Unidentifiable error patterns are those where the CRC value remains unchanged despite bit errors. If an error is a multiple of the CRC polynomial, the remainder of the division becomes 0, making it impossible to detect the error. Conversely, if the error is not a multiple of the CRC polynomial, the remainder changes, allowing us to detect the error occurrence.

The *diagnostic coverage* of a CRC is defined as the ratio of the number of error patterns that can be detected by the CRC to the total number of possible error patterns [11, 12]. The Hamming weights influences the CRC's ability to detect faults. As the Hamming weight increases, the likelihood of the CRC detecting errors rises, thereby enhancing its error detection capability.

The *residual error rate* in the context of CRC is a metric that quantifies the probability of undetected errors after the CRC has been applied to a data set [13, 12]. The residual error rate represents the probability of undetected errors after CRC verification, while the diagnostic coverage rate indicates the percentage of detectable errors. These two factors are inversely related; as the diagnostic coverage increases, the residual error rate decreases, signifying a higher level of fault detection capability.

III. Conventional Diagnostic Coverage Estimation Methods

This section outlines conventional estimation methods for the diagnostic coverage of CRC (denoted by DC) as per the communication functional safety standards [11]. The residual error rate of CRC is denoted by $\mathrm{uDC}(\equiv 1-\mathrm{DC})$. In practice, the residual error rate uDC is more convenient to work with than the diagnostic coverage DC. Therefore, subsequent discussions will focus on uDC.

A. Residual error rate under binary symmetric communication model

In CRC, the widely used communication model assumes that errors occur independently across transmitted bits, and the bit error rate is defined as the proportion of bits that have been erroneously transmitted over the total number of transmitted bits [11].

Assuming the binary symmetric channel model [13], the probability of n bit errors independently occurring in an L-bit message over a channel with the bit error rate (BER) of $p\,(0 \leq p \leq 1)$ follows a binomial distribution and can be described by

$$P_L(n) = \binom{L}{n} p^n (1-p)^{(L-n)}. \tag{1}$$

Let $\mathrm{uDC}(L, n)$ denote the undetected error rate of a CRC polynomial for n-bit errors within an L-bit message. Also, let $A(L, n)$ denote the hamming weight of a CRC polynomial for

n-bit errors within an L-bit message. Under the binary symmetric channel model, the undetected error rate $\mathrm{uDC}(L,n)$ is given by

$$\mathrm{uDC}(L,n) = \frac{A(L,n)}{\binom{L}{n}}. \tag{2}$$

Using the undetected error rate $\mathrm{uDC}(L,n)$, the residual error rate uDC is given by [11]

$$\mathrm{uDC}(L,n) = \sum_{k=1}^{L} P_L(k)\,\mathrm{uDC}(L,k) \tag{3}$$

B. Hamming distance based method

The first method estimates the residual error rate uDC of a CRC polynomials solely based on its Hamming distance. CRC polynomials exhibit a characteristic in which errors smaller than the Hamming distance, which varies for each CRC polynomial, are guaranteed to be detectable with a detection capability of 100%. However, errors equal to or larger than the Hamming distance are conservatively evaluated with a detection capability of 0% [11].

Hamming distance based method simply assumes the undetected error rate is zero for all n's larger than the hamming distance H [11].

$$\mathrm{uDC}(L,n) = \begin{cases} 1 & n < H \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

This estimation method is simple but it leads to overly cautious estimates of error detection performance.

C. Pessimistic estimation with Hamming weights

The second and the third method estimate the residual error rate of a CRC polynomial based on all available (known) hamming weights, $A(L,n)$.

However, in most cases, not all Hamming weights are available. Thus, the second method estimates its upper-bound by assuming that all errors for (L,n) without available Hamming weights are undetectable.

$$\mathrm{uDC}(L,n) = \begin{cases} \frac{A(L,n)}{\binom{L}{n}} & A(L,n) \text{ is available} \\ 1 & \text{otherwise} \end{cases} \tag{5}$$

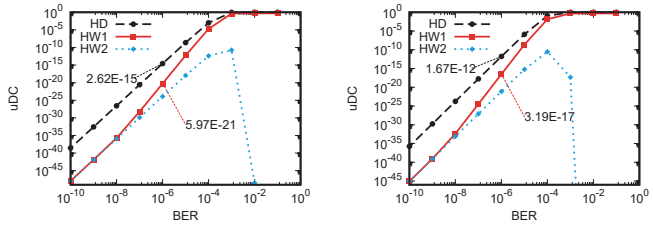
Similar to the first method, the second method also results in a conservative evaluation. However, the second method yields a tighter upper-bound than the first method.

D. Optimistic estimation with Hamming weights

The third method estimates the residual error rate optimistically; i.e., the third method estimates its lower-bound by assuming that all errors for (L,n) without available Hamming weights are detectable.

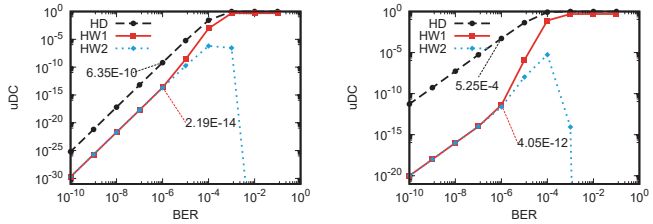
$$\mathrm{uDC}(L,n) = \begin{cases} \frac{A(L,n)}{\binom{L}{n}} & A(L,n) \text{ is available} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Since the estimated residual error rate with the third method is optimistic, it is not appropriate to design an automotive system solely based on such optimistic estimates. Instead, such optimistic estimates should be combined with other conservative estimates, such as those estimated with the second method.



(a) $L = 1,392$ [byte] (b) $L = 4,000$ [byte]

Fig. 1. Estimated residual error rate uDC of Profile 4



(a) $L = 1,392$ [byte] (b) $L = 4,000$ [byte]

Fig. 2. Estimated residual error rate uDC of Profile 5

E. Comparison of conventional estimation methods

Fig. 1 and 2 illustrate the relation between the bit error rate and the residual error rate uDC estimates obtained with three estimation methods for AUTOSAR E2E Profiles 4, and 5.

These results indicate that the Hamming distance based method is too conservative in almost all configurations. Also, these results indicate that combining pessimistic (HW1) and optimistic (HW2) estimates with Hamming weights is either sufficient or insufficient depending on the configuration (i.e., the CRC polynomial, message length L, bit error rate). For instance, in profiles 4 and 5 with longer message lengths, a significant difference between the worst-case (pessimistic) and best-case (optimistic) estimates underscores the need for improved estimation accuracy.

IV. Hamming Weight Estimation with Machine Learning

In machine learning, data processing plays a crucial role, making it essential to first investigate the characteristics of known CRC Hamming weights, which are of interest to us.

In this study, we primarily focus on the CRC polynomials used in AUTOSAR E2E profiles.

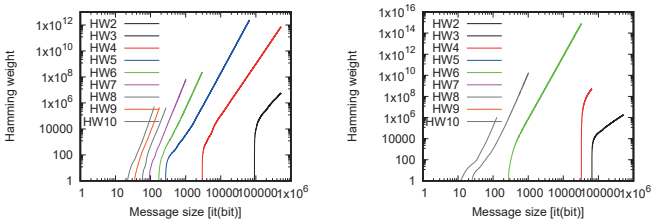
Fig. 3 shows the relation between the message size L and Hamming weights $\mathrm{HW}(L,n)$ for different number n of bit-errors and AUTOSAR profiles. These figures indicate that the Hamming weight evolves quite differently for different CRC polynomials. Such unexpected behavior of Hamming weights are not desirable as training data for machine learning.

We define the normalized Hamming weight $\eta(L,n)$ for the Hamming weight $\mathrm{HW}(L,n)$, as

$$\eta(L,n) = \frac{\mathrm{HW}(L,n)}{{}_L C_n}. \tag{7}$$

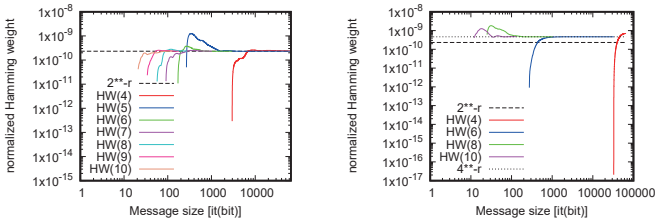
The normalized Hamming weights for different message size L are depicted in Fig. 4.

As can be seen from these figures, the normalized Hamming weights exhibit common tendency regardless of CRC



(a) IEEE 802.3 (b) Profile 4

Fig. 3. Message length vs. Hamming weight



(a) 802.3 (b) Profile 4

Fig. 4. Message length vs. normalized Hamming weight

polynomials and the number n of bit-errors, which must be quite beneficial for machine learning. So, our machine learning framework builds a predictor (regressor) for normalized Hamming weights rather than original Hamming weights.

Another design factor for building a prediction model of the Hamming weight is the choice of training dataset. In our machine learning approach, we intentionally use available Hamming weights for the *identical* CRC polynomial, rather than combining Hamming wights for a variety of CRC polynomials.

V. Experiments

In this section, we investigate the effectiveness of our machine learning framework for estimating Hamming weights of a CRC polynomial for different message length L and the number n of bit errors through extensive experiments. For this purpose, we compare the effectiveness of seven types of machine learning models and elucidates the impact of filtering training data and tuning the hyperparameters of machine learning models. Furthermore, we demonstrate that using the estimated normalized Hamming weights allows for a more accurate estimation of the diagnostic coverage of CRC than traditional methods.

A. Experimental setup

In the following experiments, we particularly focus on two major 32-bit CRC polynomials: 0xfA567D89 in the AUTOSAR E2E Profile 4 and 0x82608EDB in IEEE 802.3 (Ethernet). Among AUTOSAR E2E Profiles, Profile 4 uses the longest (32 bit) CRC compared with other profiles (1, 2, 5 and 6). Generally, assessing the diagnostic coverage for large CRC polynomials is challenge. We thus choose long (32 bit) CRCs, rather than short (8 bit or 16 bit) CRCs, because of practical importance.

For training and validating the prediction model of the Hamming weights in our framework, we used the Hamming weight dataset published in the CRC Zoo [15], which presents

TABLE I
Prediction accuracy of normalized Hamming weights for IEEE802.3 and Profile 4 CRC

	IEEE 802.3		Profile4	
model	MAE	MSE	MAE	MSE
Lasso	24.82	619.19	2.06	4.25
Ridge	24.55	605.76	2.07	4.29
MLP	25.33	682.30	69.74	4864.22
SVR	24.28	590.59	0.39	0.15
Random Forest	**12.14**	**294.10**	**0.20**	**0.04**
Gradient Boosting	12.86	295.33	0.62	0.39
LightGBM	12.17	**294.10**	0.22	0.05

Hamming weights of a number of 8, 16, 24, and 32 bit CRC polynomials for a wide range of message length L and the number n of bit-errors. Although the CRC Zoo is a large dataset, availability of Hamming weights are limited to those for small L and/or small n because of infeasible computational complexity discussed in Section I.

For CRC polynomials divisible by $(x + 1)$, all odd-bit inversions are detected; however, the undetected error rate for even-bit inversions is approximately double that of other polynomials [10]. Therefore, it is necessary to conduct experiments and evaluate the effectiveness of both polynomials with such characteristics and those without. Specifically, this study selected CRC32, as specified in the Ethernet Frame Check Sequence (FCS) of IEEE 802.3, and the polynomial used in AUTOSAR E2E Profile 4, to assess the prediction accuracy of machine learning methods applied to these polynomials.

The available Hamming weights of CRC polynomials (AUTOSAR E2E Profile 4 and IEEE 802.3) in the CRC Zoo are split two: training data and test data; for a pair of the CRC polynomial and the massage length L, the Hamming weight for the largest n among the available Hamming weights is used as test data, and others are used as training data.

For all machine learning models except LightGBM, we used the scikit-learn version 1.2.2. For LightGBM, we used the Microsoft's implementation (lightGBM version 3.3.5). The model parameters for each machine learning model were set to its default values.

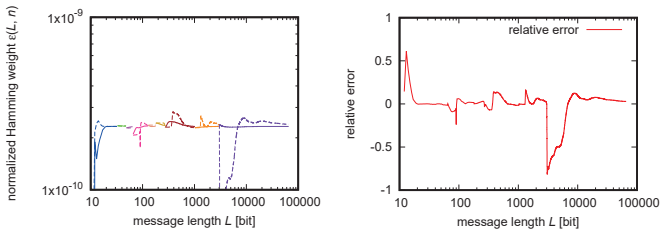
B. Effect of learning models

We first investigate what type of machine learning models is suitable for predicting normalized Hamming weight of a specific CRC polynomial by comparing the prediction accuracy of seven machine learning models in terms of MAE (Mean Absolute Error) and MSE (Mean Squared Error).

Table I summarize the prediction accuracy of seven learning models — Lasso, Ridge, MLP (Multilayer Perceptron), SVR (Support Vector Regression), Random Forest, Gradient Boosting, and LightGBM — for CRC polynomials of AUTOSAR E2E Profile 4 and IEEE 802.3, respectively.

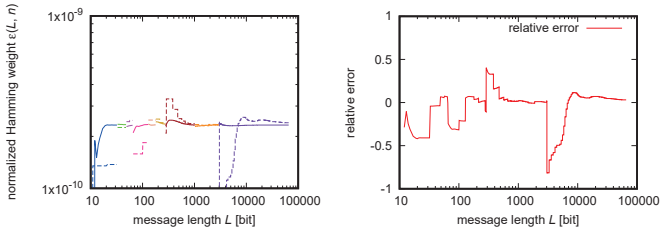
In what follows, we particularly focus on the prediction accuracy with two *good* models: Random Forest and LightGBM because of their comparatively better performance (Tab. I).

Fig. 5 and 6 shows the estimated normalized Hamming weight $\eta(L, n)$ for a range of message length L and the highest number n of bit-errors in the dataset. Each figure on the left shows both the true normalized Hamming weight (line) and the predicted one (dotted line). Each figure on the right shows the relative error of the estimated Hamming weight; i.e., the ratio of the prediction to the truth.



(a) normalized Hamming weight (b) relative error

Fig. 5. Estimated normalized Hamming weights with Random Forest



(a) normalized Hamming weight (b) relative error

Fig. 6. Estimated normalized Hamming weights with LightGBM

From this results, one can find that the decision tree-based machine learning models such as Random Forest and LightGBM shows relatively good and stable accuracy for predicting normalized Hamming weights.

C. Effect of data filtering

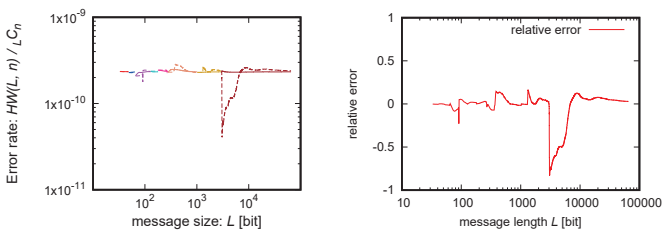
We then investigate how the data filtering is effective for estimating the normalized Hamming weights with machine learning. The Hamming weight dataset taken from the CRC Zoo has notable characteristics; i.e., a significant portion of dataset are zeros. Only a small fraction of the dataset has non-zero Hamming weights. We therefore examine how the prediction accuracy is affected by excluding training data having zero Hamming weight.

Table II presents the overall prediction accuracy of CRC polynomials (Profile 4 and IEEE 802.3) with Random Forest and LightGBM with training data filtering. Comparing these results with Tab. I indicates that data filtering (excluding data with zero Hamming weight from training) significantly improves the prediction accuracy in both Random Forest and LightGBM.

The detailed results are shown in Figs. 7 and 8, which illustrate the relation between the message length L and the estimated normalized Hamming weight $\eta(L, n)$. These results also confirm our finding — data filtering significantly improves the prediction accuracy.

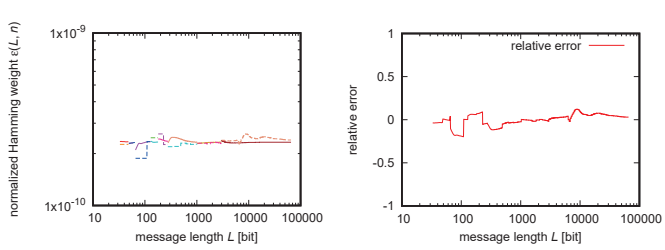
TABLE II
PREDICTION ACCURACY OF NORMALIZED HAMMING WEIGHTS WITH DATA FILTERING

CRC polynomial	model	MAE	MSE
Profile 4	Random Forest	0.388	0.953
Profile 4	LightGBM	**0.387**	**0.944**
IEEE 802.3	Random Forest	**0.05**	**0.0024**
IEEE 802.3	LightGBM	**0.05**	0.0025



(a) normalized Hamming weight (b) relative error

Fig. 7. Estimated normalized Hamming weights with Random Forest and data filtering



(a) normalized Hamming weight (b) relative error

Fig. 8. Estimated normalized Hamming weights with LightGBM and data filtering

D. Effect of hyperparameter tuning

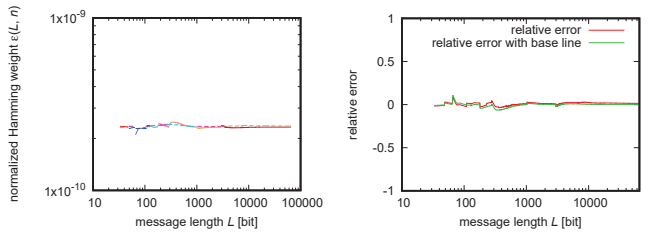
It is well-known that the tuning of hyperparameters in machine learning models is the key for obtaining the good prediction model. In what follows, we therefore examine how the estimation accuracy of the normalized Hamming weight can be improved with the hyperparameter optimization.

For this purpose, we choose LightGBM as the machine learning model, and optimize its hyperparameters with the Optuna framework [16].

Fig. 9 shows the estimated normalized Hamming weights and their relative errors for different message length L. These results clearly indicate that the hyperparameter optimization significantly improves the model accuracy.

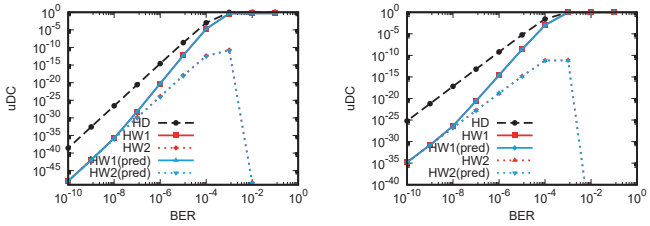
E. Implications to diagnostic coverage estimation

Our experiment results so far reveal that our machine learning framework for estimating the normalized Hamming weight is effective; i.e., with appropriate machine learning configurations — usage of a tree-based model such as LightGBM, applying data filtering to remove too many zeros, and optimizing hyperparameters, the normalized Hamming weight can be estimated with reasonable accuracy.



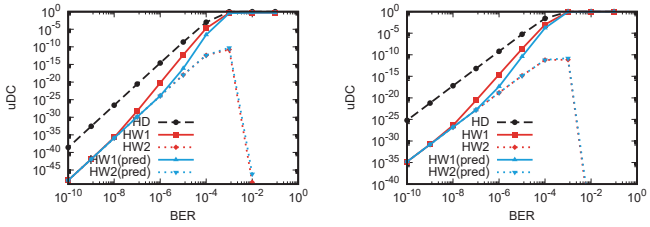
(a) normalized Hamming weight (b) relative error

Fig. 9. Estimated normalized Hamming weights with LightGBM (data filtering, hyperparameter optimization)



(a) Profile 4 (b) IEEE 802.3

Fig. 10. Estimated residual error rate $\mathrm{uDC}(L, n)$ (LightGBM, data filtering, hyperparameter optimization, $L = 1,392$ [byte])



(a) Profile 4 (b) IEEE 802.3

Fig. 11. Estimated residual error rate $\mathrm{uDC}(L, n)$ (LightGBM, data filtering, hyperparameter optimization, $L = 1,392$ [byte])

The next questions is how the better estimation of Hamming weights than conventional methods contributes to the detailed analysis and assessment of the diagnostic coverage of a CRC polynomial.

At the end of this section, we therefore examine the implications of the better estimation of Hamming weights to the diagnostic coverage (also equivalently to the residual error rate).

We first examine whether the prediction accuracy obtained with our machine learning framework is sufficient for practical purposes. Fig. 10 shows the residual error rate obtained with different methods. In the figure, "HD" indicates the Hamming distance based method. "HW1" and "HW2" indicate the pessimistic and the optimistic methods with all available Hamming weights (i.e., upto $n = 5$ for IEEE 802.3 and $n = 6$ for Profile 4), respectively. "HW1(pred)" and "HW2(pred)" are the same with "HW1" and "HW2" except the Hamming weights of the largest n is replaced with the predictions.

These results indicate that the residual error rates obtained with the true and the predicted Hamming weights are indistinguishable, which means that our machine learning approach for estimating the normalized Hamming weights is accurate uncouth for assessing the diagnostic coverage of a CRC polynomial.

We finally examine how the diagnostic coverage (and the residual error rate) is accurately estimated using our normalized Hamming weight prediction model. Fig. 11 again shows the residual error rate uDC of Profile 4 and IEEE 802.3. Fig. 11 is identical to Fig. 10 except that "HW1(pred)" and "HW2(pred)" use Hamming weight predictions $\mathrm{HW}(L, 6)$ for IEEE 802.3 and $\mathrm{HW}(L, 8)$ for Profile 4. Since $\mathrm{HW}(L, 6)$ for IEEE 802.3 and $\mathrm{HW}(L, 8)$ for Profile 4 with $L = 1,392$ [byte] are not available in the CRC Zoo, we use the estimated Hamming weights with our trained LightGBM model.

These results indicate that with an additional Hamming weight predictions, the accurately of the residual error rate estimation is greatly improved, in particular, at the bit error rate between 10^{-6} and 10^{-4}. Such a range of bit error rates is of great importance to practical automotive design.

VI. Conclusion

In this paper, we have demonstrated a novel machine learning approach to significantly improve the accuracy of estimating the diagnostic coverage of CRC in automotive systems. By leveraging supervised learning, specifically the LightGBM model, and refining the estimation process through hyperparameter tuning and strategic filtering of training data, we achieved a substantial reduction in the error margin of normalized Hamming weight predictions. These advancements allow for a more precise determination of CRC's error detection capabilities, thereby contributing to the enhancement of automotive system reliability and adherence to functional safety standards. Looking forward, the primary challenge lies in extending the methodology's robustness and applicability to a broader range of CRC polynomials and automotive configurations.

References

[1] M. Gharib, P. Lollini, M. Botta, E. Amparore, S. Donatelli, and A. Bondavalli, "On the safety of automotive systems incorporating machine learning based components: A position paper," in *Proceedings of the 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W 2018)*, pp. 271–274, June 2018.

[2] "Road vehicles – functional safety – part 5: product development at the hardware level," Standard ISO 26262-5:2018, International Organization for Standardization, Dec. 2018.

[3] "Road vehicles – functional safety – part 6: product development at the software level," Standard ISO 26262-6:2018, International Organization for Standardization, Dec. 2018.

[4] AUTOSAR, "Specification of SW-C end-to-end communication protection library," Standard R21-11 428, International Organization for Standardization, 2021.

[5] AUTOSAR, "E2E protocol specification," standard, International Organization for Standardization, 2021.

[6] S. Lin and D. J. Costello, *Error control coding: fundamentals and applications*. Pearson-Prentice Hall, 2004.

[7] W. W. Peterson and E. J. Weldon, *Error-correcting Codes*. MIT Press, 1972.

[8] G. Castagnoli, S. Brauer, and M. Herrmann, "Optimization of cyclic redundancy-check codes with 24 and 32 parity bits," *IEEE Transactions on Communications*, vol. 41, pp. 883–892, June 1993.

[9] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Prentice Hall, Dec. 1994.

[10] P. Koopman, "32-Bit Cyclic Redundancy Codes for Internet Applications," in *Proceedings International Conference on Dependable Systems and Networks*, pp. 459–468, June 2002.

[11] "Industrial communication networks – profiles – part 3: functional safety fieldbuses - general rules and profile definitions," Standard IEC 61784-3:2021, International Electrotechnical Commission, Feb. 2021.

[12] T. Forest and M. Jochim, "On the fault detection capabilities of AUTOSAR's end-to-end communication protection CRC's," in *Proceedings of the SAE World Congress and Exhibition (SAE 2011)*, Apr. 2011.

[13] T. Emi, H. N. Aung, Y. Yamasaki, and H. Ohsaki, "Improving CRC fault detection probability in AUTOSAR E2E based on known hamming weights,"

[14] W. An, M. Médard, and K. R. Duffy, "CRC codes as error correction codes," in *Proceedings of the IEEE International Conference on Communications (ICC 2021)*, pp. 1–6, June 2021.

[15] P. Koopman, "CRC Polynomial Zoo." https://users.ece.cmu.edu/~koopman/crc/crc32.html.

[16] T. Akiba, S. Sano, T. Yanase, T. Ohta, *et al.*, "Optuna: A Next-generation Hyperparameter Optimization Framework," in *Proceedings of the 25th International Conference on Knowledge Discovery and Data Mining (SIGKDD 2019)*, pp. 2623–2631, July 2019.