This article has been accepted for publication in IEEE Reliability Magazine. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MRL.2024.3468213

Are Machine Learning Classifiers Robust Enough Against Poisoned Datasets?

Abdul Majeed and Seong Oun Hwang, *Department of Computer Engineering, Gachon University, Korea*

Abstract—Machine learning (ML) classifiers are widely used in many real-world applications for classification, prediction, and regression tasks. However, data characteristics (e.g., class imbalance, feature overlap, superfluous augmentation) and inadequate logic inside ML classifiers for catering to data-related problems often lead to undesirable outcomes. This paper delves into ML failure when data are poisoned through adversarial perturbations, which can be intentional and/or malicious. Specifically, we perturb the data composition/structure, and analyze the robustness of ML classifiers in dealing with them. We found that most ML classifiers cannot withstand data perturbations, and they produce benign-looking but undesirable results. We underscore the causes of such failures, their implications for the ML field, and how to prevent them. Our work contributes to making next-generation ML classifiers more robust and reliable in the upcoming AI-driven era.

Machine learning (ML) classifiers have demonstrated their effectiveness in solving many real-world problems in diverse sectors (healthcare, finance, agriculture, etc.). In industrial settings, ML classifiers can be used to classify faulty and non-faulty machines, predict the remaining useful life of a machine, and forecast the price of industrial output. Besides, they are widely used in many other applications, such as disease classification, predicting a person's age, classifying income, predicting housing costs, and determining emotions. During the COVID-19 pandemic, ML models were handy in predicting daily case tallies, forecasting virus spread curves, classifying healthy and infected people, and drug discovery, to name a few examples. Considering the widespread applications of ML classifiers, it is fair to say they are acting like digital twins, helping humans in various ways [1]. The native application of ML to any field constitutes three key elements: data, the model/classifier, and evaluation metrics. The data can be in diverse modalities like tables, images, signals, etc. The ML model/classifier can be a decision tree, naive bayes, random forest, etc., and evaluation metrics can include accuracy, precision, and recall.

With the emergence of the data-centric artificial intelligence (DC-AI) concept [2], AI development can now be divided into two popular tracks: code (e.g., ML algorithms) and data. There has been significant development in the former, and a variety of AI models exist [3]. In contrast, there has been relatively less development in the latter (curating representative data to solve real-world problems). Since data are the cornerstone of AI development, curating reliable datasets is a key task for ML applications. Unfortunately, the publicly available datasets have many quality-related issues and cannot be directly fed into ML classifiers. In the Kaggle stroke prediction dataset[1], there are two classes, and the number of samples for one of them is too few (only 249, or 4.8% of the dataset); a classifier trained on such an imbalanced dataset cannot give reliable inferences. Although some approaches have been developed to improve data quality, it is still very challenging to curate reliable datasets for ML classifiers [4]. In addition, the lack of robustness and traceability in ML classifiers can make matters worse, and ML classifiers can produce undesirable outcomes [5]. Our major contributions are as follows.

1) We investigate the potential threats posed to ML classifiers from intentionally perturbed data, and we explore opportunities to highlight the need for robustness and traceability functions in classifiers in order to yield desirable outcomes.
2) We design and execute a human error-based data poisoning attack to mold the behavior of ML classifiers. We found that ML classifiers cannot withstand intentionally perturbed/poisoned data and can give benign-looking but undesirable outcomes from such poisoned datasets.
3) We tested the performance of our data poisoning

Digital Object Identifier 10.1109/XXX.0000.0000000

[1] https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

This article has been accepted for publication in IEEE Reliability Magazine. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MRL.2024.3468213

attack by using real-life datasets with diverse ML classifiers, showing their failures in such cases.

4) We underscore the causes of ML classifier failures, the implications for the ML field, and prevention/remedies. Our work can assist in making next-generation ML classifiers more robust/traceable in catering to data manipulations.

Preliminaries and Problem Overview

This work considers tabular data, D, with an $n \times m$ dimension where n is the number of rows, and m is the number of columns. In the ML field, rows and columns are regarded as data points and features, respectively. An example of data used in ML classifiers is shown in Fig. 1. As shown in Fig. 1, there are multiple predictors,

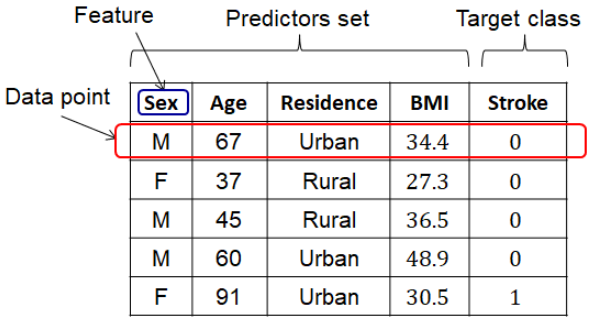

FIGURE 1. Overview of tabular data for ML classifiers.

which can be denoted as set $X = \{x_1, x_2, \ldots, x_p\}$; Y is the target class having a cardinality of 2, i.e., $(y_1, y_2) \in Y$, where $y_1 = 1, y_2 = 0$. As shown in Fig. 1, the distribution of samples is imbalanced (more samples belong to y_2). In samples randomly drawn from D, representations of y_1 will be rare. This leads to an imbalanced learning problem in classifiers, where only one class is adequately learned during training and the other might remain unlearned, posing a misclassification risk. In realistic cases, the distribution of target classes should be balanced to guarantee balanced learning in classifiers. The imbalance in Y can be calculated by counting the class examples and employing the $\mathcal{IR}$ formula:

$$\mathcal{IR} = c_M / c_m \tag{1}$$

where c_M and c_m denote the total number of samples in the major and minor classes of D. An $\mathcal{IR}$ close to 1 indicates that the data are balanced w.r.t. classes.

In the above example (see Fig. 1), $\mathcal{IR}$ is 4, which is $>>$ 1, and ML classifiers cannot adequately learn c_m. To balance the distribution, D is either upsampled or downsampled. There are two well-known solutions to address class imbalance problems: random oversampling (ROS) and random under-sampling (RUS), defined as follows.

1) ROS: Upgrade the c_m distribution by expanding the data size; either acquire more samples by using information in the available samples, or curate synthetic samples with the help of generative AI (e.g., generative adversarial networks).
2) RUS: Downgrade c_M by removing some samples until the number of samples equals the number of samples in c_m.

While these techniques are handy, they both have pros and cons in the context of ML classifiers. For instance, RUS can decrease data size and accuracy, whereas ROS might enlarge the data with minimal improvements in results. In addition, a small intentional/unintentional mistake while up- or down-sampling can lead to wrong results, prevention of which is the main motivation behind this work.

Problem overview

In this work, we intend to solve Problem 1.

Problem 1. *We are given a real-world dataset, D, in tabular form that includes various mixed types of features (age, name, race, sex, residence, BMI, etc.) and target classes (disease occurrence probability, and income). There can be multiple quality-related problems in D such as a limited number of data points, incomplete instances, repetitive instances, skewed distributions of a target class, etc. How do we produce a (poisoned) version, dataset $\mathcal{D}$, where (a) $\mathcal{D} \subseteq D$, (b) $\mathcal{D} \sim D$ (e.g., most of the predictor compositions/structures in $\mathcal{D}$ are nearly the same as they are in D, except for the target class), and (c) accuracy from $\mathcal{D}$ (the poisoned form of D) is higher than from D in most cases, regardless of the ML classifier used?*

Specifically, we take classifiers, feed them both non-poisoned and poisoned datasets, and experimentally prove that most ML classifiers cannot detect poisoned datasets, and instead, give better results from poisoned data than from non-poisoned data. Hence, ML classifier failures from poisoned data can call into question the robustness as well as the traceability. As a result, an ML classifier used in safety-critical applications can provoke concern, and more efforts are needed to resolve such failures from ML.

Data Poisoning Attacks on ML Classifiers

In this section, we discuss ML classifiers trained with both poisoned and non-poisoned data. The classifier-building process with both kinds of data is illustrated in Fig. 2, which shows three key processes: D is already

balanced, D is imbalanced and requires balancing by curating more data, and D is poisoned with the proposed process. First, we build the classifier just like a traditional pipeline, and we later discuss the proposed attack (e.g., building a classifier with poisoned data).

Building a classifier with non-poisoned data

Given a real-world D in tabular form, we first pre-process it to prevent the possibility of wrong results. As discussed above, most real-world datasets can be noisy, and are therefore pre-processed to enhance quality before feeding them into classifiers. In this work, we pre-process D for missing values, outliers, redundant samples, and consistency in feature values considering their respective types. To impute missing values, we use both the mean and minor/super-minor value methods. Specifically, if a value is missing in a numerical feature, the mean of the feature is inserted (computed from available values). In contrast, missing values in categorical features are given values that have too few representations among samples of the feature. Outliers are identified based on the $min-max$ range and ground truth label information in numerical and categorical features, respectively. Features with outliers are processed in a manner identical to imputing missing values in order to preserve data size. Redundant tuples are removed by computing similarities between samples. Specifically, two samples are regarded as redundant when they yield a similarity value of 1 and are located next to each other in D. In this case, one sample is removed from D to save computing time as well as to prevent redundant learning by ML classifiers. The consistency in the values is ensured by leveraging the *type of (x)* command in *R*. Helped by this process, a complete D is curated.

After pre-processing, D is further analyzed for class imbalance. Specifically, unique class examples are identified in the target class column, and their frequency in D is computed. Later, $\mathcal{IR}$ is computed to determine whether D is balanced or not. This paper deals with binary cases, and therefore, C_M and c_m are determined based on frequency. Balance/imbalance is decided with Eq 2:

$$D(\mathcal{IR}) = \begin{cases} \sim 1, & \text{D is balanced} \\ >> 1, & \text{D is imbalanced} \end{cases} \tag{2}$$

If D is balanced, it undergoes partitioning for training and testing purposes. If D is imbalanced, then more data will be curated to balance D. In this work, we employ our previous work to curate synthetic data in order to balance the distributions of c_M and c_m [6]. We used the ROS technique to balance the distribution with as few records as possible. Two-thirds of D is used for training classifiers, and one-third is used for testing.

Once data are partitioned, a classifier is fed the training data. In this work, we used random forest (RF) as a base ML classifier [7]. The key reason to use RF is its ability to handle mixed data as well as its superior performance compared to other ML classifiers. It builds a forest of trees and gives prediction/classification results via majority voting from individual tree results. Furthermore, it has only two parameters (*ntree* and *mtry*), which can easily be tuned considering the problem size. It draws samples from the training data and builds trees over them until the specified number of trees is constructed. Later, predictions are made with testing data to analyze the performance. In the end, we construct a confusion matrix, and evaluate accuracy using Eq. 3:

$$Accuracy = \frac{T_p + T_n}{T_p + T_n + F_n + F_p} \tag{3}$$

where T_p, T_n, F_n, F_p denote true positive, true negative, false negative, and false positive, respectively.

After calculating *Accuracy*, we store the results for comparison with the second setting (results from the poisoned dataset).

Building a classifier with poisoned data

In this section, we reveal the setting of our proposed attack on the ML classifier by intentionally perturbing the dataset and making other minor adjustments to the ML classifier. AI systems usually fail due to three crucial inadequacies: there are some problems with the input data, there is deficient processing logic in the AI model, and the repertoire of actions related to output is inadequate [8]. These problems arise from the following errors [8].

1) Doing something that should have not been done, called error of commission (EOC).
2) Not doing something that should have been done, called error of omission (EOO).

In [8], the authors further mapped the above errors to input, processing, and output. Based on established knowledge in the ML field, we delve into designing poisoned data by adopting some errors in order to mold ML classifier behavior. Specifically, we adopted a combination of EOOs and EOCs, and curated poisoned data to force the classifier to learn substandard data. It is worth noting that in the AI field, models are sometimes exposed to poisoned/adversarial data or examples to enhance their generalizability and robustness [9]. In this work, we do the opposite, and present adversarial perturbations that can stem from either

This article has been accepted for publication in IEEE Reliability Magazine. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MRL.2024.3468213

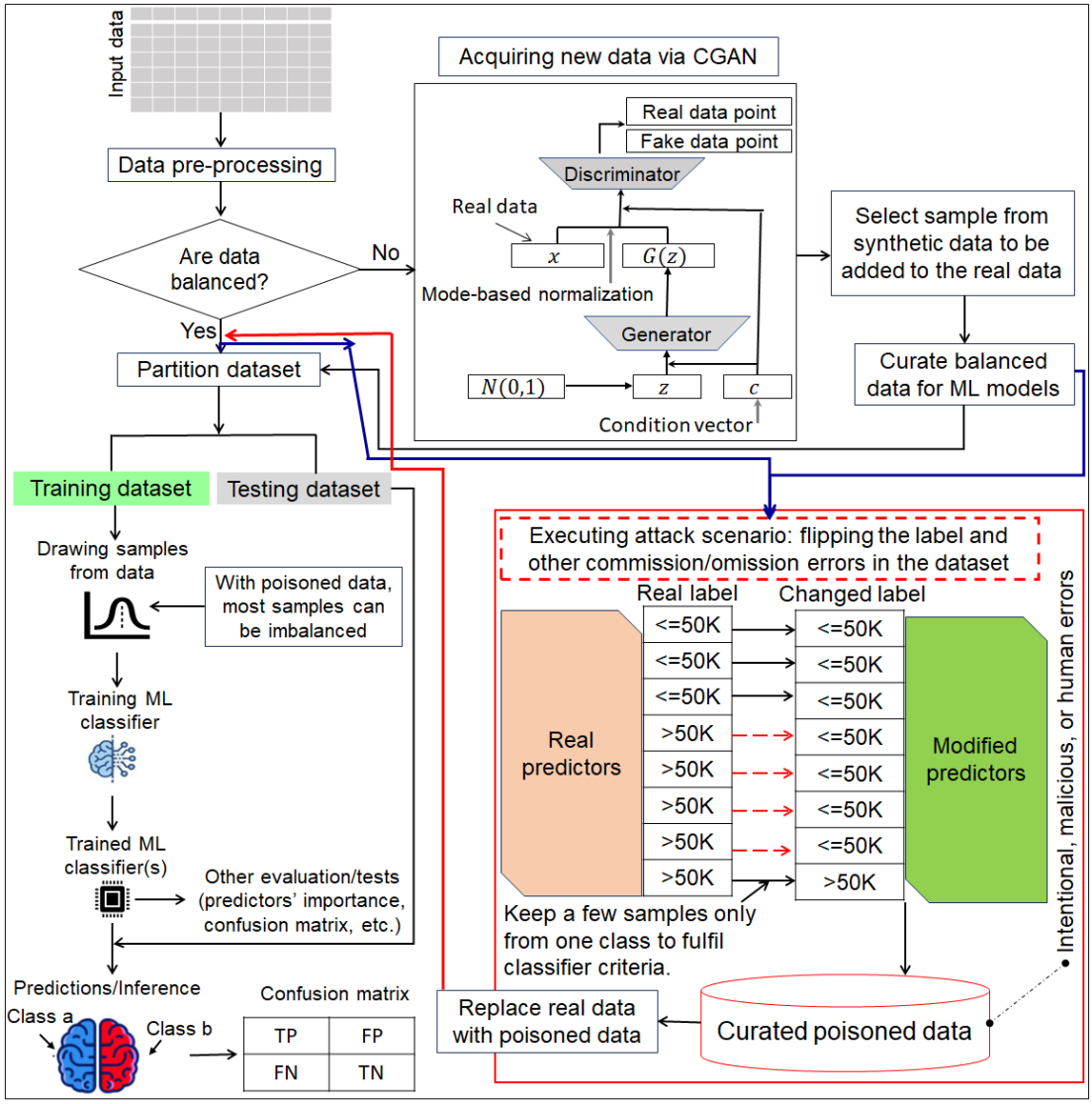

FIGURE 2. Traditional ML classifier building, and executing the proposed data poisoning attack.

human error or adversaries. In this work, the following errors were adopted from Chanda and Banerjee [8] to change the behavior of ML classifiers:

- Input: OI4, OI5, and C12
- Process: OP2, OP3, and CP2
- Output: OO4 and CO1

The input errors are related to different problems with the data as data is regarded as fuel/input for AI systems. These errors can be too few records in the data or some salient information/feature is being dropped from the data before building an AI/ML model. The processing errors are related to inadequate processing logic while training ML models. In these errors, the ML models do not give any error/warning even when either parameters are not consistent with data size or some data is not used during the training process. The output errors are related to inappropriate outputs due to an inadequate repertoire of actions.

In this work, we used eight different errors (e.g., 3 input, 3 processing, and 2 output) to change the ML behavior. By using the above errors, we reduced salient information in the input data, excluded diverse constituents, and intentionally deleted some features. In processing, we catered to classifier-related failures through logic (e.g., we prevented model collapse by minimally meeting the desired implementation constraints), we used random values in the parameters,

This article has been accepted for publication in IEEE Reliability Magazine. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MRL.2024.3468213

and the necessary logic was not employed to stop execution in case of failure. In the output, we hide the structure of the confusion matrix and analyzed only the numerical *Accuracy* score (which is a conventional practice in AI), and we built models by varying feature overlaps to lead the classifier into faulty actions. Specifically, we curated highly poisoned data with extremely imbalanced, altered values in features, randomly removed some features, selected hyperparameters poorly, made improper data divisions for training and testing, and distorted the real distribution of values. We found that ML classifiers cannot deal with these data-related problems very well, and instead produced benign-looking (but wrong) results, which necessitates, for the good of society, a call to action to develop robust and traceable ML classifiers for the upcoming AI-driven era.

Experiment Evaluations

In this section, we provide the results attained through experiments on benchmark datasets to prove the feasibility of the proposed concepts. We conducted experiments on two benchmark datasets: the Kaggle Stroke Prediction Dataset[1] and the UCI ML Repository's Adult[2] dataset. The rationale for considering these two datasets for experiments is their imbalanced nature and mixed features (i.e., numerical and categorical). Both datasets are benchmark and have been used widely in many studies and for coding contests. Also, both these datasets require a standard practice of data balancing, and therefore, we chose them for experiments to prove the feasibility of our idea in real as well as augmented form. There are 10 and 14 predictors in the stroke prediction and adult datasets, respectively, with 5,111 and 32,561 samples, respectively. In all experiments, about ⅔ ($\sim$ 66%) of the data was used in training ML models while about ⅓ ($\sim$ 33%) was used as test data. For the initial experiments, we carefully divided the datasets and employed optimal hyperparameters for the RF model. Later, we perturbed the dataset and determined the results by using *Accuracy* as the evaluation matrix.

Data exploration: In this section, we explore and discuss the patterns within the real and poisoned datasets. Specifically, we investigate the composition and structure of real and poisoned datasets from the perspective of predictors and target class. Table 1 lists the ten unique patterns that were derived through an in-depth analysis of predictors as well as target class in both real and poisoned data.

TABLE 1. Overview of patterns in real and poisoned data.

Pattern (s)	Real data	Poisoned data
Class imbalance	High	Very high
Number of features/predictors	More	Less
Overlapped feature values	Less	More
Majority to minority class sample ratio	Low	High
Features' domain values	High	Low
Flipped class labels	No	Yes
Distortion of true data distribution	No	Yes
Corelation b/w feature values	Consistant	Inconsistant
Salient information	Not reduced	Reduced
Diversity in predictors' values	High	Moderate

Referring to Table 1, it can be observed that poisoned data has quite distinctive patterns compared to the real data. However, some of the patterns (e.g., feature selection) have proven effective even in non-adversarial settings.

We divided the experiment into three settings: *Accuracy* comparisons between real and poisoned data using all information in D, *Accuracy* comparisons between real and poisoned data by choosing samples randomly at varying scales, and *Accuracy* comparisons between real and poisoned data using all the information from D with diverse ML classifiers. For most experiments (Setting I and II), we used the RF model because it is one of the most widely adopted ensemble ML models for classification/regression tasks due to its ability to construct multiple trees and consider correlations between them. However, to prove the generality of our idea and the validity of experiments, four other classifiers (e.g., DT, SVM, LR, NB) in Setting III were also employed in performance evaluation. Next, we discuss each setting along with the empirical results.

Setting I: *Accuracy* using all the data

In the first set of experiments, we computed and compared the *Accuracy* results between real and poisoned data by utilizing all of D. Because both real datasets are imbalanced, we first computed *Accuracy* for the original versions. Later, we addressed the class imbalance problem by adding synthetic records to c_m so that $|c_M| == |c_m|$. In the end, we created poisoned versions of each dataset (real and augmented) to compute and compare *Accuracy*. Table 2 presents the results achieved with the two benchmark datasets, showing that *Accuracy* from the poisoned dataset outperformed *Accuracy* from the real data in most cases. These results demonstrate the failure of ML when D is either maliciously perturbed or an incompetent someone crafts the dataset for ML tasks. To give a true picture of the outcome of the ML experiments, we run the ML experiments ten times with different random splits of train and test datasets and report the average *Accuracy* instead of reporting the performance measure just based on a single run in Table 2. Specifically,

[2]https://archive.ics.uci.edu/dataset/2/adult

we varied the training data size from 65% $\sim$ 80%, and test data size from 35% $\sim$ 20% in experiments. The *Accuracy* values in bracket are obtained with 10-fold cross validation strategy (9 folds for training ML model and 1 fold for testing/inference).

TABLE 2. Avg. *Accuracy*: real data versus poisoned data.

Dataset	RD	R-PD	AD	A-PD
Stroke	94.62 (94.93)	99.88 (99.90)	99.80 (99.85)	99.92 (99.95)
Adults	86.07 (86.22)	99.96 (99.97)	99.99 (100)	99.97 (99.99)

[a] RD= real data, R-PD= real poisoned data, AD= augmented data, A-PD= augmented poisoned data.

We believe most researchers try to achieve higher total *Accuracy* by not paying ample attention to the contribution from each class. For example, *Accuracy* from using the stroke prediction data was very high (94.85), even from the original form. However, most contributions in *Accuracy* came from one class (c_M), and there were higher misclassifications from c_m, leading to poor inferences. Although there are some remedies for this particular problem (balanced *Accuracy*, or other metrics like precision, recall, etc.), total *Accuracy* is still regarded as one of the key performance indicators in benchmarking studies, compared to others in diverse fields. The distributions of data considerd in *Accuracy* determination in Setting I are in Table 3.

TABLE 3. Data distributions used in Setting I.

Dataset	Criteria	RD	R-PD	AD	A-PD
Stroke	Total records	5110	5110	8552	8552
	Train data size	3373	3373	5644	5644
	Test data size	1737	1737	2908	2908
Adults	Total records	32561	32561	43880	43880
	Train data size	21490	21490	28961	28961
	Test data size	11071	11071	14919	14919

Comparisons across other ML models evaluation metrics: To further verify the sophisticatedness of the attack, we provide comparison across two other ML model evaluation metrics, namely precision (*PR*) and recall (*RE*). The formalization of metrics is in Eqs. 4 and 5.

$$PR = \frac{T_p}{T_p + F_p} \tag{4}$$

$$RE = \frac{T_p}{T_p + F_n} \tag{5}$$

The results of *PR* and *RE* computed from the real and poisoned versions are given in Table 4. These results are computed from the optimal cases (e.g. when misclassifications are very low).

Setting II: *Accuracy* when varying data size

In the second set of experiments, we randomly chose a set of samples from real as well as poisoned datasets

TABLE 4. Avg. *PR* and *RE*: real data versus poisoned data.

Dataset	RD	R-PD	AD	A-PD
Stroke (*PR*)	99.76	100	99.88	99.93
Adults (*PR*)	93.22	99.99	99.95	99.89
Stroke (*RE*)	95.05	99.88	99.77	99.96
Adults (*RE*)	88.46	99.99	99.91	99.98

[a] *PR*= precision, *RE*= recall

to analyze the impact on *Accuracy*. Specifically, we chose a group of samples with a scale of 20%, and varied the group at an identical scale. The main reason to conduct these experiments is to show ML failures not only with all the data but also with subsets ranging from small to large. These kinds of experiments are vital in order to showcase ML failures when different parts of the data are poisoned. Table 5 presents *Accuracy* results obtained by varying data sizes.

TABLE 5. *Accuracy* comparison when varying the data size.

Dataset	Data nature	20%	40%	60%	80%	100%
Adults	Real	91.63	82.45	90.98	96.55	100
	Poisoned	99.93	99.96	99.93	99.99	99.98
Stroke	Real	96.16	95.57	95.87	97.74	99.63
	Poisoned	100	99.83	99.88	99.87	99.89

From Table 5, we can see that poisoned data yielded much higher *Accuracy* than the real data in most cases. The main reason for such a result is that the composition/structure of the data was altered in multiple locations, not just a few. These results validate the effectiveness of a data perturbation strategy showing that ML classifiers can easily be fooled by carefully crafted adversarial samples. Through these experiments, we also prove the generalizability of the proposed attack, rather than testing it only using all the data. These results validate the effectiveness of the proposed concepts, and underscore likely failures from ML classifiers using bad data. The distributions of data considerd in *Accuracy* determination in Setting II are given in Table 6.

TABLE 6. Data distributions used in Setting II.

Dataset	Criteria	20%	40%	60%	80%	100%
Stroke	Total records	1710	3420	5130	6840	8552
	Train data size	1129	2257	3386	4514	5644
	Test data size	581	1163	1744	2326	2908
Adults	Total records	8777	17551	23336	35104	43880
	Train data size	5793	11584	15402	23169	28961
	Test data size	2984	5967	7934	11935	14919

Setting III: *Accuracy* from diverse classifiers

In the third set of experiments, we applied five different ML classifiers—the decision tree (DT), logistic regression (LR), random forest (RF), naive Bayes

(NB), and the support vector machine (SVM)—to verify the feasibility of the proposed concepts. The basis on which classifiers were chosen are data characteristics (e.g., binary nature, mixed feature type, and tabular format) and the nature of the task (e.g., classification/regression). These classifiers have been widely used in many real-world problems for classification/prediction purposes. We adopted the best five ML models from both categories of ML: single classifier (DT, SVM, LR, NB), and ensemble methods (RF). Each classifier has distinct properties and parameters, leading to effective learning and inference from the real and poisoned datasets. The key reason to use NB is its ability to feature independence and parameter estimation from a small amount of data. Also, the use of conditional probability and Bayes theorem make it suitable for accurately separating instances of each class. Furthermore, it fastly reaches the asymptotic error and often yields better performance than other classifiers, particularly LR. The main reason to use diverse classifiers is to prove the universality and validity of the proposed attacks. We tested the performance of each classifier on real and augmented data, and on poisoned versions of both the original and the augmented datasets. The distributions of data considered in *Accuracy* determination in Setting III are the same as Setting I (given in Table 3). Table 7 presents the results obtained from the five classifiers with the adult and stroke prediction datasets.

TABLE 7. *Accuracy* from two benchmark datasets using distinct classifiers (Avg. of ten runs with different data splits).

Dataset	Classifier	RD	R-PD	AD	A-PD
Adults	DT	84.40	99.96	99.46	99.98
	RF	86.07	99.96	99.99	99.97
	SVM	83.26	99.96	99.87	99.98
	NB	82.92	95.31	83.07	99.93
	LR	83.21	99.86	99.61	99.95
Stroke	DT	95.13	99.89	99.62	99.95
	RF	94.62	99.88	99.80	99.92
	SVM	95.11	99.90	99.67	99.89
	NB	86.90	80.97	97.27	88.44
	LR	95.02	99.47	99.62	99.79

[a] RD= real data, R-PD= real poisoned data, AD= augmented data, A-PD= augmented poisoned data.

In Table 7, we can see that most classifiers yielded better results from the poisoned data than from the original data, which verifies the main assertion of this work. Furthermore, Accuracy from poisoned data is very close to optimal (i.e., $\sim$ 100%). From the results in Table 7, it can be concluded that most ML classifiers can easily be fooled with inaccurate data. These results emphasize the need to add extra controls in ML classifiers to cater to data-related manipulations that can be either (un)intentional or malicious.

Causes, Implications, and Prevention of ML Failures

In order to yield desirable results, this section highlights the causes, implications, and prevention of ML failures. In the ML field, it is vital to proactively monitor the data throughout the project life cycle, because mistakes in the early stages can inadvertently propagate to later stages and become difficult to correct. Lones [10] devised a general guide about the pitfalls of ML and their avoidance. The author spanned the guide into five stages: initial preparation, reliable training, robust evaluation, fair comparison, and results analysis. Recently, some best practices to reliably build ML solutions have also been discussed in the literature [11]. We identified nineteen different causes of ML failure and mapped them to three key components of ML pipeline development: data, the model, and evaluation. We also provide the implications and prevention of each cause. To the best of our knowledge, none of the prior studies have uncovered these crucial aspects of ML development. Our work can guide ML practitioners/researchers in overcoming AI failures in modern applications, enhancing an ML classifier's acceptance/trust. Concise discussions of each aspect are provided below.

Data: Fig. 3 provides an overview of causes, implications, and prevention strategies of ML failures from the perspective of data. Andrew Ng, founder and CEO of Landing AI and DeepLearning.AI started the campaign for DC-AI and coined the phrase *"Data is food for AI"* [12]. Considering the importance of data in AI, Ng stressed the need to radically enhance data quality in AI systems rather than in the code of an AI model alone. The quality of data can impose risks on the underlying decisions, and many individuals can be impacted. Considering the significance of data in AI systems, we identified seven causes whereby data can lead to ML failure. Bad quality data refers to data that are not complete from most aspects and can lead to deficient performance. In our own work, we discussed seven dimensions of data quality in the context of AI development [3]. Another main cause of ML failure is human error due to a lack of expertise or domain knowledge. In some cases, a data scientist might be simultaneously working on multiple projects, which can lead to deficient performance from ML models. In some cases, data can be imbalanced/inadequate, and acquiring more data might be challenging, leading to ML failure. In some cases, data might lack diversity, which can lead to ML failure in terms of unfair decisions, poor generalization, and wrong predictions. In many sectors, there is a lack of data valuation methods, which can

Causes	Implications	Prevention/remedy
Bad-quality data	Undesirable outcomes after deployment	Rigorous adoption of DC-AI practices and tools
Human error (s)	Establishing wrong benchmarks/score	Training and engaging multiple data scientists
Data imbalance/inadequacy	Poor generalizability of ML models	Adopt data balancing and augmentation methods
Lack of diversity in training data	Poor inference and misclassifications	Curating synthetic data to enhance data diversity
Lack of data valuation methods	Data drift and concepts drift issues	Multi-criteria valuation of data before training ML
Higher feature overlap in data	Deficient learning and lack of reliability	Feature scoring and dimensionality reduction
Improper division of test and train data	Biased and less credible results	Rigorous valuation of testing and training data

FIGURE 3. Comprehensive overview of causes, implications, and ways to prevent ML failures (data-related).

lead to ML failure. For example, it is hard to judge how much diversity is enough when building ML models. Similarly, it is hard to ensure the completeness of data before building a model, which can lead to ML failure. Similarly, in some cases, data can be collected from a single domain (or the same community of people), which does not improve the learning ability of ML classifiers and can lead to ML failure. Improper division of testing and training data, and lack of evaluation for each partition can also lead to ML failure.

All of the above-cited data-related causes have severe implications in the ML field, as listed in the middle column of Fig. 3. For example, ML classifiers trained on bad data can give wrong answers/predictions right after deployment, which cannot pay dividends to stakeholders. Human error leads to establishing wrong benchmarks for subsequent research, and can lead to financial loss. Incomplete/inadequate data can lead to poor generalization by ML classifiers in real-world settings. Data with the least diversity can lead to higher misclassifications for under-represented classes. Data that lack valuation can lead to data/concept drift in real settings. Feature overlap can cause ML classifiers to underfit/overfit, which can lead to inconsistent performance. Lastly, improper division of testing/training data can lead to bias or unfair decisions, which is a hot issue in many ML/AI applications. To prevent ML failures, we suggest potential remedies for each cause in the last column of Fig. 3. Our work can pave the way to rectifying data-related shortcomings, which can lessen ML failures in real-world cases.

The model:Fig. 4 provides an overview of causes, implications, and prevention strategies of ML failures from the perspective of the model. A lot of developments have been made in classifiers, and a variety exists nowadays. However, most classifiers cannot learn beyond the data, which means they can easily be fooled by feeding them the wrong data. Enrico and Gatti [13] highlighted crucial problems in ML methods from a lack of causal explanations and evaluations behind the predictions. The higher dependence of ML classifiers on data can make them vulnerable to false classification/regression results when data are poisoned as depicted in this paper. In most classifiers, there is a serious lack of warning/notification, particularly when data are problematic. Unfortunately, most classifiers flag errors only for very basic problems such as when only one class is present in test data for binary classification, or the denominator in the accuracy formula is zero. Similarly, most classifiers cannot provide rules/formulas in making some predictions/classifications, which indicates a lack of transparency. The black-box nature of most ML models seriously impacts their adoption/acceptance in society [14].

In some cases, the testing data will leak information and the model cannot detect it, which can lead to poor reliability but better accuracy. A coding example is in Fig. 5. In this example, there are two values ($<$=50K, $>$50K) for the target class. When we include both classes in the training data, but only one class is in the test data, there is an error message. However, when we keep both classes in each set of data, but significantly downgrade the frequency of one class in the test data (only six samples), there is no error message. We believe a classifier should generate an error notification in this case as well, because the diagonal in the confusion matrix is zero. In current classifiers, such errors/controls are not captured well, which may lead to ML failure in some cases. A poor choice of hyperparameters can cause poor performance in terms of generalizations and learning ability. For example, building an RF model with inappropriate values of *ntree* and *mtry* can lead to deficient performance. Lastly, the classifier can be either too simple or too complex, and cannot handle complexities/variations in the data, leading to higher misclassifications (or wrong predictions). The deployment of an underfitted/overfitted model cannot yield desirable results in realistic scenarios, and can yield results that are unacceptable to the general public.

All the above-cited model-related causes have se-

This article has been accepted for publication in IEEE Reliability Magazine. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MRL.2024.3468213

Causes	Implications	Prevention/remedy
Higher dependence on data	Wrong result with slightly new data	Enhancement of inductive bias in ML classifiers
Lack of data monitoring	Garbage in garbage out issues	Influence analysis of samples/(sub)populations
Lack of transparency	Adversarial attacks and unfair training	Explainability integration and error messages
Lack of fair learning	Lack of trust/acceptance of ML results	Cross-validation and adversarial examples tests
Testing data leakage	Poor adaptability to new datasets	Early partitioning of test data and their analysis
Poor hyperparameter selection	Biased/unfair decision making	Optimization using built-in ML functions/libraries
Over-/under-fitting of the ML model	Legal implications and certification issue	Train models with more data or prune bad data

FIGURE 4. Comprehensive overview of causes, implications, and ways to prevent ML failures (model-related).

```
         y_pred
           <=50K  >50K
   <=50K  11396     1
>
> #Accuracy analysis from the confusion matrix.
> test_accuracy = (cm_rf[1,1] + cm_rf[2,2])/
+    (cm_rf[1,1] + cm_rf[2,2] + cm_rf[2,1] + cm_rf[1,2])
Error in `[.default`(cm_rf, 2, 2) : subscript out of bounds
```

(a)

```
         y_pred
           <=50K  >50K
   <=50K  11390     1
   >50K       6     0
>
> #Accuracy analysis from the confusion matrix.
> test_accuracy = (cm_rf[1,1] + cm_rf[2,2])/
+    (cm_rf[1,1] + cm_rf[2,2] + cm_rf[2,1] + cm_rf[1,2])
> print(test_accuracy)
[1] 0.9993858
```

(b)

FIGURE 5. The code in (a) issues an error notification, but the code in (b) does not.

vere implications for the ML field, as listed in the middle column in Fig. 4. For example, most models cannot learn beyond the extent of the data, and thus, yield wrong results from unseen data. A lack of data monitoring can result in garbage in, garbage out situations[3] because ML models only want the data, right or wrong. The black-box nature of most ML classifiers makes them vulnerable to attack and poor results. Unfair learning can decrease public trust in ML-generated results, and therefore, most AI applications still require humans in the loop. The dependence between testing and training data can make ML models work well only on data already seen, and the classifier can fail right after deployment in real-world settings. The selection of poor hyperparameters can lead to improper results. An underfitted/overfitted model cannot get certification, which can lead to financial losses for stakeholders. To prevent ML failures, in the last column of Fig. 4, we suggest potential remedies for each cause. Our work can pave the way to rectifying model-related shortcomings, which can reduce ML failures.

Evaluation: Fig. 6 provides an overview of causes, implications, and prevention strategies of ML failures from the perspective of evaluation. Poor evaluations and average reporting of results can under-represent some rare examples/subpopulations, which can lead to severe ML failure[4]. Failing to use truly held-out data while evaluating classifiers can yield better results, but it cannot generalize well to unseen data. Similarly, the evaluation of classifiers with test data that have a higher dependence on the training data or on balanced proportions in the classes can lead to ML failure. The usage of improper evaluation metrics (e.g., accuracy instead of balanced accuracy in an imbalanced data scenario) can lead to ML failures. In addition, the evaluation of ML classifiers with limited metrics cannot truly capture the behaviors of ML classifiers from all perspectives, and can lead to failures when facing unexpected scenarios/data. Unfortunately, most researchers are concerned with overall accuracy without paying attention to the structure of the confusion matrix, which can cause ML failure if some classes remain unlearned. In addition, most classifiers do not flag an error from having no true positives or true negatives, leading to poor reliability in safety-critical applications. Lastly, improper selection of benchmark scores can seriously undermine the credibility of ML results, and most classifiers cannot meet the expectations of the general public. The use of validation data that are not representative of the deployment setting, and overlooking annotation errors in the test data can lead to ML failure.

All of the above evaluation-related causes have severe implications in ML, as listed in the middle column of the first row of Fig. 6. For instance, biased selections in some subsets of data for use as testing data to prove the significance of classifiers in academic labs can lead to unreliable and inconsistent results in real-world settings, which can lead to poor adoption of ML. Improper evaluation of classifiers can result in million-dollar fines as well as legal costs. Using only a few met-

[3]https://dl.acm.org/doi/abs/10.1145/3446776

[4]https://cacm.acm.org/news/263929-sloppy-use-of-machine-learning-is-causing-a-reproducibility-crisis-in-science/fulltext

Causes	Implications	Prevention/remedy
Selection bias in testing data	Unreliable and inconsistent results	Random data splits and novelty addition to data
Improper evaluation metrics	Trust/acceptance and financial losses	Selection of related metrics considering domain
Too few evaluation metrics	Harm to some specific communities	Selection of metrics considering the data modality
Poor analysis of confusion matrix	Irresponsible use and governance of ML	Analysis of TN & TP, and error/warning messages
Selection bias in benchmark score	Poor reproducibility and regulatory issue	Valid benchmark selection from credible sources

FIGURE 6. Comprehensive overview of causes, implications, and ways to prevent ML failures (evaluation-related).

rics for evaluation can harm some marginalized groups if the model is not evaluated with diverse metrics. By not analyzing confusion matrixes, data flows, and training procedures, ML classifiers cannot be governed properly. The use of poorly evaluated classifiers cannot benefit society or stakeholders. Lastly, poor selection of benchmarks can create the reproducibility problem, which is a leading cause of poor AI deployments around the globe[5]. To prevent ML failures, we suggest potential remedies for each cause in the last column of Fig. 6. Our work can pave the way to rectifying evaluation-related shortcomings, which can lessen ML failures in real-world cases.

Discussion and Prospects

The use of ML is expanding with each passing day; many commercial applications exist that assist people in various ways. However, along with the rapid proliferation of ML applications, there is a significant rise in ML failures leading to unintended consequences. To prevent negative consequences from ML/AI, multiple global/local initiatives have recently been taken by many countries. Notable initiatives include Global Partnership on AI (GPAI)[6], the EU AI Act[7], the DEEL project[8], ethics of AI by UNESCO[9], responsible AI practices by Google[10], and the AI Governance Alliance of the World Economic Forum[11], to name just a few. Most of these have been initiated to foster the responsible use of AI while minimizing harm to people.

Recently, researchers have also identified many perils in ML models that require rectification in order to yield desirable outcomes. Barbierato et al. [13] highlighted crucial problems with ML techniques in terms of a lack of causal explanations concerning achieved predictions and poor learning abilities. The authors discussed important implications of ML model shortcomings in creditworthiness scenarios. Tarawneh et al. [15] highlighted the need to stop employing the oversampling technique, which is a very dominant trend in the AI community, particularly when data are imbalanced. They stressed the need to avoid using oversampling techniques in real-world applications owing to design flaws. Christen and Schnell [16] discussed practical issues related to data governance in the AI-driven era and stressed the need to share/document data-related failures in order to foster rectification. Grodzinsky et al. [17] highlighted the harms of AI from the latest development and profit-related motives. Slota et al. [18] discussed the idea of accountable AI with the help of socio-technical approaches to limit AI failures. Besides, there have been calls in the AI community to either pause development to fix things [12] or shut down completely to fix them [19]. Considering all these developments, it is fair to say that a lot of work is needed to prevent failures in AI/ML systems.

Analysis from the perspective of reliability engineering: In reliability engineering, the ability of a system is measured to properly function/work for a specified time/condition. This is usually accomplished by arranging the components of the system in a parallel/series fashion, and then computing the probability of the system to function from the failure probabilities of its component. In some ML algorithms, particularly supervised ML, the class membership is decided based on some salient features and feature values, which are regarded as states (feature values) and components (feature) in the context of reliability engineering [20]. Since the decisions for class membership are made based on combined values of all features or a combination of a few salient feature values only, and therefore, AI system failures can be measured by computing the probability of each feature and its values. The EOOs/EOCs employed in this work target both feature and their values. Therefore, the EOOs/EOCs can be

[5]https://www.nature.com/articles/d41586-023-03817-6

[6]https://gpai.ai/

[7]https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

[8]https://www.deel.ai/

[9]https://www.unesco.org/en/artificial-intelligence/recommendation-ethics

[10]https://ai.google/responsibility/responsible-ai-practices/

[11]https://initiatives.weforum.org/ai-governance-alliance/home

[12]https://spectrum.ieee.org/joy-buolamwini

classified into different categories based on their impact on ML models, and the '*k* out of *n*' model from the reliability engineering field can be adopted to estimate the probability of ML model failures/success. Adopting the reliability engineering concepts, particularly the '*k* out of *n*' model, and estimating each possible ML system state (feature values in this work) and component (feature in this work) probability at any point in time can pave the way for detecting/analyzing ML failures.

Lastly, reliability engineering is the mainstream field that has guided the reliability of complex engineering systems from the past few decades[13]. It uses a "bathtub curve" concept to model the engineering systems' lifecycle along with the three different phases/stages of failure rates: early, random, and wear-out failures. It can be adopted to model the AI systems' life cycle while identifying and diagnosing three different types of faults: failures due to algorithm design and training data, failures due to unexpected inputs/conditions, and failures due to concept or data drifts. This work considers three different types of errors/failures related to input, processing, and output, therefore, they can be modeled using the "bathtub curve" concept, and mathematical functions (a.k.a reliability functions) can be adapted for the diagnosis/classification of failures occurring at each stage/phase. The integration of these concepts/principles from the reliability engineering field can enhance the safety and innovation of ML-powered systems while lowering undesirable outcomes.

Conclusion and Future Work

This paper underscored the lack of robustness/ traceability functions in ML classifiers, which can lead to ML failure when facing poisoned datasets, provoking reliability concerns in real-world applications. Specifically, we designed and executed a data poisoning attack (e.g., reducing salient information, excluding diverse data constituents, randomly deleting some features, and distorting the true data distribution) on ML classifiers and observed that most of them cannot withstand poisoned data. Poisoned data can be curated in three ways: intentionally, maliciously, and/or due to the incompetence of data scientists. Furthermore, there is a lack of data monitoring in ML classifiers, and many data-related problems can go undetected, leading to undesirable outcomes. To this end, we comprehensively discussed the causes, implications, and prevention of ML failures. Finally, to answer the question posed in this article's title, we conclude that, unfortunately, the ML classifiers are not fully robust yet when it comes to facing poisoned (or intentionally perturbed) datasets. To address the problems, practical and end-to-end solutions like Influenciae [21] are needed to prevent ML failures from happening. There is an urgent need to develop robust ML models to cater to adversarial perturbations in data to prevent ML models from producing undesirable and erroneous outputs, despite appearing benign on the surface. In the future, we intend to analyze tools/libraries that have been recently developed to monitor data flows across ML pipelines. Lastly, we intend to further verify the robustness of the proposed concept/idea by employing advanced and custom ensemble ML classifiers on diverse datasets encompassing multiple classes.

[13]https://oecd.ai/en/wonk/reliability-engineering-data-driven-risk-management

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (RS-2024-00340882).

REFERENCES

1. J. Parmar, S. Chouhan, V. Raychoudhury, and S. Rathore, "Open-world machine learning: applications, challenges, and opportunities," *ACM Computing Surveys*, vol. 55, no. 10, pp. 1–37, 2023.
2. E. Strickland, "Andrew ng, ai minimalist: The machine-learning pioneer says small is the new big," *IEEE Spectrum*, vol. 59, no. 4, pp. 22–50, 2022.
3. A. Majeed and S. O. Hwang, "Technical analysis of data-centric and model-centric artificial intelligence," *IT Professional*, vol. 25, no. 6, pp. 62–70, 2023.
4. J. Wang, S. Liu, and W. Zhang, "Visual analytics for machine learning: A data perspective survey," *IEEE Transactions on Visualization and Computer Graphics*, 2024.
5. P. Giudici, M. Centurelli, and S. Turchetta, "Artificial intelligence risk measurement," *Expert Systems with Applications*, vol. 235, p. 121220, 2024.
6. A. Majeed and S. O. Hwang, "Ctgan-mos: Conditional generative adversarial network based minority-class-augmented oversampling scheme for imbalanced problems," *IEEE Access*, 2023.
7. L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.
8. S. S. Chanda and D. N. Banerjee, "Omission and commission errors underlying ai failures," *AI & society*, pp. 1–24, 2022.
9. R. Barati, R. Safabakhsh, and M. Rahmati, "On continuity of robust and accurate classifiers," *arXiv preprint arXiv:2309.17048*, 2023.
10. M. A. Lones, "How to avoid machine learning pitfalls: a guide for academic researchers," *arXiv preprint arXiv:2108.02497*, 2021.

11. A. Serban, K. van der Blom, H. Hoos, and J. Visser, "Software engineering practices for machine learning—adoption, effects, and team assessment," *Journal of Systems and Software*, vol. 209, p. 111907, 2024.
12. G. Press, "Andrew ng launches a campaign for data-centric ai," *Forbes, Jun*, vol. 16, p. 2021, 2021.
13. E. Barbierato and A. Gatti, "The challenges of machine learning: A critical review," *Electronics*, vol. 13, no. 2, p. 416, 2024.
14. A. Rawal, J. McCoy, D. B. Rawat, B. M. Sadler, and R. S. Amant, "Recent advances in trustworthy explainable artificial intelligence: Status, challenges, and perspectives," *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 6, pp. 852–866, 2021.
15. A. S. Tarawneh, A. B. Hassanat, G. A. Altarawneh, and A. Almuhaimeed, "Stop oversampling for class imbalance learning: A review," *IEEE Access*, vol. 10, pp. 47 643–47 660, 2022.
16. P. Christen and R. Schnell, "When data science goes wrong: How misconceptions about data capture and processing causes wrong conclusions," 2024.
17. F. S. Grodzinsky, M. J. Wolf, and K. W. Miller, "Ethical issues from emerging ai applications: Harms are happening," *Computer*, vol. 57, no. 2, pp. 44–52, 2024.
18. S. C. Slota, K. R. Fleischmann, S. Greenberg, N. Verma, B. Cummings, L. Li, and C. Shenefiel, "Many hands make many fingers to point: challenges in creating accountable ai," *AI & SOCIETY*, pp. 1–13.
19. E. Yudkowsky, "Pausing ai developments isn't enough. we need to shut it all down," *Time Magazine*, vol. 29, 2023.
20. Z. Ursani and D. W. Corne, "Use of reliability engineering concepts in machine learning for classification," in *2017 IEEE 4th International Conference on Soft Computing & Machine Intelligence (ISCMI)*. IEEE, 2017, pp. 30–34.
21. A. M. Picard, L. Hervier, T. Fel, and D. Vigouroux, "Influenciæ: A library for tracing the influence back to the data-points," pp. ffhal–04 284 178, 2023.

Abdul Majeed is an Assistant Professor in the Department of Computer Engineering, Gachon University, Korea. He received his Ph.D. in Computer Information Systems & Networks from the Korea Aerospace University (KAU), Korea. Contact him at ab09@gachon.ac.kr

Seong Oun Hwang is a Professor in the Department of Computer Engineering, Gachon University, Korea. He received his Ph.D. in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Korea. He is a senior member of IEEE. Contact him at sohwang@gachon.ac.kr