IDPS Signature Classification Based on Active Learning With Partial Supervision From Network Security Experts

HIDETOSHI KAWAGUCHI, YUICHI NAKATANI, AND SHOGO OKADA, (Member, IEEE)

1 Network Innovation Center, NTT, Musashino, Tokyo 180-8585, Japan
2 Japan Advanced Institute of Science and Technology, Nomi, Ishikawa 923-1211, Japan
Corresponding author: Hidetoshi Kawaguchi (kawa.hide39@jaist.ac.jp)

This work was supported in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant 22H04860 and Grant 22H00536; and in part by the Japan Science and Technology Agency (JST) AIP Trilateral Artificial Intelligence (AI) Research, Japan, under Grant JPMJCR20G6.

ABSTRACT As intrusion detection and prevention systems (IDPSs) grow in importance, the cost of managing signatures, which are pattern files of malicious communications, continues to increase. To ensure the optimal operation of an IDPS, network security experts need to classify the signatures generated over time according to their importance (low, medium, and high). Although machine learning approaches can be used to automatically classify signatures instead of human experts, there are several challenges to applying them, including (a) high annotation costs, (b) security incidents caused by classification errors, and (c) classification accuracy decreases due to domain shifts. To overcome these challenges, we propose a system based on active learning that collaborates with experts to periodically classify received signatures. The signatures are sorted by uncertainty sampling; some are transferred to experts, and the rest are automatically classified. The experts classify the transferred signatures and add them to the training dataset, and the classification model is retrained. After training, the new signatures that have not yet been labeled are classified. The proposed system executes this workflow each time it receives signatures. For evaluation purposes, a real dataset was collected monthly with the help of the experts. Experiments are conducted on this real dataset to evaluate the proposed system in a simulation case. An analysis is then performed by comparing several variants of the proposed system. The results show that the system with Monte Carlo dropout (MC-Dropout) performs best. We also show that this variation has two effects: it transfers more samples with “medium” importance to the experts and mitigates imbalances in the training dataset.

INDEX TERMS Machine learning, active learning, real dataset, IDPS signatures, uncertainty estimation.

I. INTRODUCTION Intrusion detection and prevention systems (IDPSs) monitor network systems and take actions such as logging, notification, and blocking when malicious communications are detected. This paper focuses on a type of IDPS that performs detection based on pattern files of malicious communications such as IDPS signatures [1], [2], [3]. The signatures are distributed periodically by the IDPS developer, similar to a subscription service.

Network security experts determine the importance level of each signature to set IDPS actions. For example, if the importance of a signature is high, the action is “blocking”; if it is low, the action is “logging”. This importance level determination process requires expertise and should be considered a significant cost of network security operations.

Automatic signature classification via machine learning, specifically supervised learning, is one of the available approaches. We built a classification model and evaluated its performance on our dataset in a previous study [4]. However, several challenges are encountered when applying classification models to the real world. (a) High annotation costs - Only a limited number of people can annotate...
signatures due to the need for expertise. (b) Security incidents caused by classification errors - Incorrect IDPS configurations due to classification errors may cause a security incident that could be fatal to the organization. (c) Classification accuracy decreases due to domain shifts - New signatures may result in decreased accuracy because the distribution of the new signatures is different from that of the trained signatures.

To overcome these challenges, we propose a classification system based on active learning that collaborates with experts to periodically classify received signatures. An overview of the proposed system is shown in Fig. 1. First, new signatures are received, and the signatures with the most uncertainty regarding the classification of their importance levels are annotated by the expert (uncertainty sampling [5]). The expert then determines the importance of the transferred signatures and labels them. This step has two effects: it acquires training data that are effective in improving classification accuracy and efficiently prevents errors by annotating reliable labels (reject option [6], [7]). These new labeled signatures are added to the training dataset, the parameters of the classification model are initialized, and the model is retrained. After training, the new signatures that have not yet been labeled are classified. The proposed system executes this flow each time it receives signatures. Based on active learning, the classification model can be periodically retrained with less labeled data, reducing classification errors by canceling the classification of uncertain signatures.

In this paper, we demonstrate the effectiveness of the proposed system through experiments conducted on a unique dataset that we developed under cooperation with experts. The dataset consists of labeled signatures collected every month for two years with the help of network security experts. To the best of our knowledge, how to automatically classify a real signature dataset that was collected over a long period of time has not yet been explored.

The proposed system uses uncertainty sampling to determine the signatures to be transferred to experts. Since the accuracy of uncertainty estimation affects the overall system performance, better methods need to be explored. This paper focuses on improving the accuracy of uncertainty sampling by utilizing deep learning-based uncertainty estimation methods [8]. Among the available uncertainty estimation methods, we confirm the performance improvement effects yielded by Monte Carlo dropout (MC-Dropout) [9] and deep ensembles [10], which are applicable for any data structure.

The contributions of this paper are as follows.

1) We present an efficient active learning system for classifying signatures that dynamically change with time.
2) With expert help, we developed a labeled signature dataset collected every month for two years and formulated an evaluation metric that fits the real-world situation.
3) We experimentally demonstrate and analyze the effectiveness of the proposed system.

Section II describes our research on reducing the operational costs of IDPSs, active learning, the reject option, and uncertainty estimation in deep learning to show this research’s value. Section III discusses the challenges of applying the signature classification models proposed in previous studies [4] to real-world applications. Section IV describes the active learning-based signature classification system. Section V describes the experimental setup: the real collected dataset, the original evaluation metric, and the comparative methods. In Section VI, we demonstrate the effectiveness of the proposed classification model through experiments. In Section VII, we discuss the proposed system
through an experimental analysis. In Section VIII, we summarize and discuss future issues.

II. RELATED WORKS
IDPSs and their operations are regarded as important tasks for defending network systems from cyberattacks, and research on reducing the operational costs of IDPSs (Section II-A) is actively conducted. Active learning is also positioned as a research area of machine learning and has recently been studied as an approach for reducing the amount of training data required for deep neural networks (Section II-B). Signature classification, as in the medical field, is subject to significant damage when classification errors occur, and the reject option function has been studied to overcome this challenge (Section II-C). Uncertainty estimation is another popular area of research in deep learning, which is being put to practical use due to its high prediction accuracy (Section II-D). This section reviews these studies and describes the position of this research.

A. REDUCING THE COSTS OF IDPS OPERATIONS
While it is crucial to improve the detection capability of an IDPS, it is also essential to operate the IDPS efficiently in network security scenarios. The most burdensome parts of IDPS operations are responding to alerts caused by false positives and managing signatures.

Many alerts are actually due to false positives in the IDPS, and the users of the IDPS are forced to deal with them daily. Some studies have analyzed alerts from IDPSs and reduced the number of alerts produced. Alsubhi et al. proposed a fuzzy theory system to estimate the priority levels of alerts [11]. Tadeusz proposed a system that incorporates machine learning to reduce false alerts [12].

Some studies have been conducted to organize and discard signatures that are generated daily. Stakhanova et al. proposed an analytical model for finding conflicting signatures [13]. In their model, signatures are represented as nondeterministic automata, and signature overlaps are detected based on automata equivalence. For the same purpose, Massicotte and Labiche proposed another approach based on set and automata theories [14].

Our research is categorized as signature management research. We built the first machine learning model for automatic signature classification in our previous work [4]. However, the verification in that paper was limited to simple cross-validation, and this paper examines how to actually utilize the model.

B. ACTIVE LEARNING
Active learning is a framework that aims to complete the training task with minimal annotation costs [5], [15]. In active learning, predictive models are trained by selecting the samples from an unlabeled dataset that are most likely to be useful for training and having an annotator (also called an oracle) label them while building the training data. Labeling is costly in fields that require expertise, so researched it is being actively conducted to overcome this challenge. Active learning is applied to medical image processing [16], [17], [18], [19], [20], clinical text classification [21], [22], machine translation [23], [24], [25], [26], [27], chemical scenarios [28], [29], [30], and patent classification [31].

Our proposed system is a natural integration of this active learning paradigm into an expert’s periodic signature classification task. This means that the cost of annotation is reduced for experts. The most popular main fields of active learning research are image and natural language processing. Other data types are relatively less explored, and to the best of our knowledge, there are no examples of applications of active learning to signature data structures.

C. REJECT OPTION
We interpret the transfer of unlabeled signatures to experts in active learning as a reject option [6], [7]. The reject option is a function that cancels classification based on a quantified uncertainty threshold decision. It has been investigated primarily in areas with high classification failure costs, such as medical fields [32], [33], [34].

Signature classification has the exact same high misclassification cost as that of the medical field. This is because misconfiguring a IDPS may cause malignant communications to be missed. Therefore, the reject option (transferring signatures to an expert through an acquisition function) is also valuable for signature classification.

D. UNCERTAINTY ESTIMATION IN DEEP LEARNING
When using uncertainty sampling [5] as the acquisition function for active learning, the accuracy of the estimated probabilities output by the classification model is important. If the accuracy of the estimated probabilities is low, the uncertainty cannot be properly estimated, and better training samples cannot be selected.

An interesting topic in deep learning that has recently been applied in many fields is the calibration of deep neural networks, which is the process of correcting the predicted probabilities estimated by a deep neural network to the actual probabilities [8]. This is expected to be synergistic with uncertainty sampling, which requires the proper estimation probabilities to be determined. Since Guo et al. reported that large deep neural networks tend to be overconfident, further research has been conducted [35]. Calibration approaches include post hoc methods [35], [36], [37], [38], [39] that modify estimations after the fact and regularization methods [40], [41], [42], [43], [44], [45] that modify the objective function and augment the data.

Many methods have been proposed to express the uncertainty of deep neural networks, and these techniques are also effective for calibration. The calibration performance of deep ensembles [10] has been empirically shown to exceed that of post hoc methods and Bayesian neural networks [46], [47], [48]. These are the mainstream Bayesian methods [9], [49], [50], [51], [52], [53], [54], [55], [56], but the power of deep ensembles as non-Bayesian approaches has also been
demonstrated [10]. Such methods also have calibration capabilities and are expected to be highly compatible with uncertainty sampling since the starting point better represents uncertainty.

Our proposed system is based on active learning and uses uncertainty sampling as the acquisition function. This paper also verifies the performance of the proposed system by incorporating uncertainty estimation methods, such as MC-Dropout and deep ensembles. These approaches have become particularly representative among uncertainty estimation methods, partly due to their ease of implementation, and further work on such methods is still being conducted today. In particular, [57] reported that they could be made better by combining active learning and deep ensembles.

III. PROBLEM SETUP

This section describes three challenges encountered when applying the IDPS signature classification model [4] to real-world applications.

A. ONLINE SIGNATURE CLASSIFICATION

Automatic classification with machine learning techniques helps experts determine the importance of signatures. In our previous study, we constructed such a classification model with machine learning and verified its performance on our dataset [4]. However, three challenges are encountered when applying this classification model to the real world: (a) high annotation costs, (b) security incidents caused by classification errors, and (c) classification accuracy decreases due to domain shifts. The details of each are as follows.

(a) High annotation costs: Only experts with knowledge and experience can perform signature classification. In other words, collecting labeled signatures for training is not easy. It is necessary to train the classification model on a limited dataset.

(b) Security incidents caused by classification errors: Classification errors can lead to improper IDPS configurations, which may cause security incidents. Security incidents can significantly damage the social credibility of an organization and sometimes cause fatal damage. As in the medical field, mechanisms that reduce classification errors, such as the reject option, are needed for our field.

(c) Classification accuracy decreases due to domain shifts: Signatures are periodically generated to keep up with new cyberattacks. This causes a domain shift, and there is concern that signature classification models will not be able to effectively classify new signatures. Fig. 2 shows a simple analysis of this issue. We have developed a time-stamped signature dataset (Section V-A). In this dataset, we measured balanced accuracy (BACC) using the following two hold-out methods. (i) Ignoring timestamps (blue bars in Fig. 2). (ii) Considering timestamps; new signatures were classified by a classification model trained on old signatures (red bars in Fig. 2). If no domain shift has occurred, there should be little differences between the BACCs of these cases. However, the BACC is lower for the time series split case. In other words, domain shift occurs; the simple classification models are ineffective in real situations. We need a mechanism to ensure that a classification model can keep up with new signatures.

The signature classification task is similar to the challenges of applying machine learning in the medical field; (a) high annotation costs and (b) security incidents caused by classification errors are common challenges. Our signature classification task is characterized by the challenge concerning (c) classification accuracy decreases due to domain shifts. In the medical field, classification targets are usually observation data from the human body, which do not change significantly even if the time series changes. However, signatures are generated as new cyberattacks are created, so they change significantly.

B. FORMATION OF IDPS SIGNATURES

The target signatures are described in a notation that corresponds to the IDPS security engine named Snort. 1 A specific example2 is shown in Fig. 3, and the format of a signature is described below. The string located at the leading “alert” indicates the action of the IDPS when detected by that signature. Actions are set by experts based on importance, so features are extracted from the strings after an action is taken.

“tcp $EXTERNAL_NET any -> $HOME_NET 79” contains 5-tuple elements separated by spaces. A 5-tuple is a set of five pieces of information described in the header of an IP packet, consisting of the communication protocol, source IP address, source port number, destination IP address, and destination port number. “-gt” indicates the direction of communication. “tcp,” “$EXTERNAL_NET”, “any”, “$HOME_NET”, and “79” represent the communication protocol, source IP address, source port number, destination IP address, source port number, destination port number, and destination IP address, respectively.

1https://snort.org/
2https://www.snort.org/downloads/#rule-downloads
destination IP address, and destination port number, respectively. The above elements are required for signatures, and the following elements in parentheses are options that can be specified at the discretion of the signature designer. The options are expressed in key-value pairs but have the following characteristics.

- Keys and values are linked by ";" symbols.
- The ";" character is used as a key-value delimiter.
- There is a key-value pair that allows multiple identical keys.
- Some values, such as “no case”, have no key.

We focus on two key values, *msg* (abbreviation of message) and *reference*. The *msg* is a string that will be included in the log or alert produced when a signature is matched. The “PROTOCOL-FINGER bomb attempt” in Fig. 3 is an example of an *msg*. The *reference* contains information for referencing external information. The example “cve,1999-0106” indicates that the Common Vulnerabilities and Exposures (CVE) identifier is 11157, which can be uniquely retrieved from external sites. There are two ways to describe a *reference*. First, the name and ID of the product’s vulnerability list are entered. The vulnerability lists include nessus and Bugtraq, in addition to CVE. A CVE with ID 1999-0067 is listed as “cve, CVE-1999-0067”, and Bugtraq with ID 629 is listed as “bugtraq,629”. Second, the URL is directly described as the destination for accessing the information.

### C. FEATURE ENGINEERING

The proposal in this paper is for a system that includes the developed IDPS signature classification model for real-world applications. Note that this method is independent of the feature design process in the classification model. In this section, we present the feature design used in the evaluation step, which is the target of the proposed system. The feature design is the one proposed in our previous study [4]. An overview of the feature design is shown in Fig. 4.

The entire feature vector is created by independently extracting each of the elements in a 5-tuple, the *msg*, and the *reference* and finally combining them.

The 5-tuple is decomposed into five parts: a communication protocol, source IP address, source port number, destination IP address, and destination port number. Each of these is then converted into a numeric vector using one-hot encoding. For example, if there are three types of symbols, A, B, and C, they are converted to features as (1, 0, 0), (0, 1, 0), and (0, 0, 1), respectively.

For the *reference*, web scraping is performed to obtain information from the external information to which the *reference* refers. The *reference* is a set containing the name of the vulnerability list (CVE, Bugtraq, etc.) and its ID or URL so that the information related to the signature can be uniquely identified. For example, when referring to CVEs, information can be obtained by searching with IDs in web systems such as the National Vulnerability Database (NVD)\(^3\) and RedHat’s CVE Database. Examples of signature-related information include the software targeted by the malicious communication, which is indicated by the signature and its version information. The developers performed a web scraping process for each web system that publishes the information referred to by the *reference*. Although it is difficult to obtain information from all web systems, we can realistically code the process by focusing on web systems that are frequently referenced. In the following, the term *reference* refers to the information obtained by web scraping.

Before converting the *msg* and *reference* to a numeric vector, the following process is performed for cleansing. First, only alphabetic characters, numbers, and underscores are used; all other symbols are deleted. Next, words that are considered stop words in English [58] and words that appear only once in the training dataset are deleted.

The cleansed *msg* and *reference* are separately converted into feature vectors by term frequency-inverse document frequency (TF-IDF) [59], [60], [61]. The TF-IDF process converts a document into a numeric vector consisting of TFs multiplied by the IDF vectors. A TF represents the number of occurrences in the given document, and the IDF is the rarity of the word. Assuming that *d* is the identifier for the document (the *msg* or *reference* in the signature) and *w* is the identifier for the word, the TF-IDF is as follows.

\[
tf-idf(w, \ d) = tf(w, \ d) \cdot idf(w) \tag{1}
\]

\(tf(w, \ d)\) is the number of occurrences (an integer greater than or equal to zero) of the word *w* that are contained in document *d*. \(idf(w)\) is as follows.

\[
idf(w) = \log \frac{N_L + 1}{df_L(w) + 1} + 1 \tag{2}
\]

\(N_L\) is the number of documents in the training dataset, and \(df_L(w)\) is the number of words *w* contained in the \(N_L\) documents. In this paper, TF-IDF treats all words as unigrams. It also performs L2 normalization and min-max scaling with a minimum value of zero and a maximum value of one.

### IV. SIGNATURE CLASSIFICATION WITH ACTIVE LEARNING

In this section, we propose an active learning system (Fig. 1) that uses the IDPS signature classification model developed in [4] as one of the elements for overcoming the three challenges mentioned above: (a) high annotation costs,

---

\(^3\)https://nvd.nist.gov/vuln

\(^4\)https://access.redhat.com/security/security-updates/#/cve (2021/4/16)
(b) security incidents caused by classification errors, and
(c) classification accuracy decreases due to domain shifts.

A. PROCEDURES FOR APPLYING ACTIVE LEARNING

We present notations showing the processing steps of the proposed active learning-based system. The key to understanding the procedure is how to build the training dataset and the set of signatures to be classified for each discrete time step \( t \in \{0, 1, \ldots, T\} \). Let \( s(t) \in \mathbb{N} \) be the number of annotations set by the expert at each time step \( t \). \( s(t) \) is a hyperparameter that the system user sets while considering the annotation cost. At each time step \( t \), the top \( s(t) \) signatures resulting from uncertainty sampling are annotated and added to the training dataset. After the classification model is retrained, the signatures that were not annotated are classified.

Let \( X(t) \) be the space of the signatures at time \( t \). The set of signatures sampled from \( X(t) \) is \( \mathcal{X}(t) = \{x_i\}_{i=1}^{N(t)} \). \( N(t) \in \mathbb{N} \) is the number of signatures given at time \( t \), e.g., the number of signatures distributed by the IDPS developer. The space of the importance labels assigned to the signatures is constant regardless of the time series, with \( y \in \mathbb{Y} = \{1, \ldots, C\} \). The indices of the signatures manually annotated by experts at time \( t \) are \( I^{(t)}_{\text{train}} \subset \{1, \ldots, N(t)\} \), and the indices of the signatures automatically classified by the machine learning model are \( I^{(t)}_{\text{test}} \subset \{1, \ldots, N(t)\} \). Let \( A^{(t)} = \{x_i, y_i\}_{i=1}^{N(t)} \) be the dataset labeled by the experts at time \( t \). Let \( D^{(t)}_{\text{train}} = \{x_i, y_i\}_{i=1}^{N(t)} \) be the initial training dataset. The training dataset at time \( t \) is constructed by accumulating a labeled dataset \( \mathcal{A}^{(t)} \) as follows.

\[
\mathcal{D}^{(t)} = D^{(0)}_{\text{train}} \cup \mathcal{A}^{(1)} \cup \ldots \cup \mathcal{A}^{(t-1)} \cup \mathcal{A}^{(t)}
\]

Algorithm 1: The Procedure of the Classification Model at \( t \)

1. Given a dataset \( \mathcal{X}^{(t)} = \{x_i^{(t)}\}_{i=1}^{N(t)} \) and \( s(t) \).
2. \( y(t) = \max_{i \in I^{(t)}_{\text{train}}} \{y(x_i^{(t)}) : x_i^{(t)} \in \mathcal{X}^{(t)}\} \)
3. \( I^{(t)}_{\text{train}} = \{i \in \{1, \ldots, N(t)\} \cap \{y(x_i^{(t)}) \geq y(t)\}\}
4. The experts annotate \( \mathcal{A}^{(t)} \) of \( D^{(t)}_{\text{train}} \) to develop a labeled dataset \( \mathcal{A}^{(t)} \).
5. \( D^{(t)}_{\text{train}} = D^{(t-1)}_{\text{train}} \cup \mathcal{A}^{(t)} \)
6. Train the classification model’s parameter \( w(t) \) with \( D^{(t)}_{\text{train}} \).
7. \( I^{(t)}_{\text{test}} = \{i \in \{1, \ldots, N(t)\} \cap \{y(x_i^{(t)}) < y(t)\}\}
8. \( A^{(t)} = \{x_i, y_i\}_{i \in I^{(t)}_{\text{test}}} \)
9. Predict the classes in \( A^{(t)} \) with the classification model trained in the above step.

Let \( w(t) \) be the parameter of the classifier trained from \( D^{(t)}_{\text{train}} \). Let \( P(y|x, w(t)) \) be the membership probability of sample \( x \) as predicted by the classifier trained on \( D^{(t)}_{\text{train}} \). The output label of the classifier is calculated as

\[
\hat{y} = \arg \max_{y \in \mathbb{Y}} P(y|x, w(t)).
\]

The procedure of the system at step \( t \) is summarized in Algorithm 1. For effective active learning, it is important to design an acquisition function that selects annotation targets from unlabeled data. Our proposed system uses an uncertainty sampling strategy. The uncertainty estimation function \( \nu : X(t) \rightarrow \mathbb{R} \) is used for uncertainty sampling. For the classification model to consistently achieve high accuracy, it is crucial to design the function \( \nu \). The function \( \nu \) should output higher values for signatures with more uncertain importance labels and lower values for more certain samples. The simplest example of \( \nu \) is entropy. Entropy is calculated as follows:

\[
u_H(x) = - \sum_{c \in \mathbb{Y}} P(y = c|x, w(t)) \log(P(y = c|x, w(t))).
\]

Let \( \nu(x) \) be a function that returns the \( n \)-th highest value among the given values. At time \( t \), the proposed system transfers the top \( s(t) \) uncertain signatures quantified by the function \( \nu \) to the experts for annotation. The uncertainty threshold for selecting such signatures is as follows:

\[
y(t) = \max_{i \in I^{(t)}_{\text{train}}} \{y(x_i^{(t)}) : x_i^{(t)} \in \mathcal{X}^{(t)}\}
\]

The indices \( I^{(t)}_{\text{train}} \) of the samples transferred to the experts at time \( t \) and the index set \( I^{(t)}_{\text{test}} \) of the samples classified by the machine learning model are as follows:

\[
I^{(t)}_{\text{train}} = \{i | i \in \{1, \ldots, N(t)\} \cap \nu(x_i^{(t)}) \geq y(t)\}
I^{(t)}_{\text{test}} = \{i | i \in \{1, \ldots, N(t)\} \cap \nu(x_i^{(t)}) < y(t)\}
\]

The experts annotate the signatures whose indices are contained in \( I^{(t)}_{\text{train}} \) to form \( \mathcal{A}^{(t)} \). The training dataset \( \mathcal{D}^{(t-1)}_{\text{train}} \) are combined into \( \mathcal{D}^{(t)}_{\text{train}} \). Then, \( \mathcal{D}^{(t)}_{\text{train}} \) is used to train \( w(t) \). After performing training, the target signature dataset
The key idea of our proposed system is to overcome the multiple challenges that arise when applying signature classification models within an active learning framework to real-world situations. We explain why the proposed system can overcome the challenges listed above. (a) In the active learning process, which is the basis of the system, the system picks up data that are considered helpful for training and trains on a smaller set of labeled data by asking humans to label the data. (b) Transferring signatures to experts in the active learning acquisition function corresponds to a reject option, a function that avoids classification errors by canceling uncertain classifications. (c) The classification model is also naturally updated sequentially to keep up with the periodically generated signatures.

**B. UNCERTAINTY ESTIMATION PERFORMANCE IMPROVEMENT**

We evaluate the performance of the proposed system when MC-Dropout [9] or a deep ensemble [10] is used for the classification model to improve the system’s performance in Section VII. MC-Dropout is a method in which dropout [62], generally used during training, is also used during the inference process. The final output is the average of the probability vectors calculated from multiple feedforward steps. A deep ensemble is a classifier consisting of multiple neural networks. [57] conducted a study that examined the combination of active learning and uncertainty estimation. In the field of image recognition, this study experimentally showed that while MC-Dropout was adequate, the deep ensemble gave better results. However, in the experiments described below, we show that MC-Dropout performs better than the deep ensemble on the dataset used in this paper.

Both MC-Dropout and deep ensembles require multiple feedforward steps during inference. Assuming that the common variable \( K \) represents the number of feedforward steps of each method, the prediction probabilities are as follows.

\[
P(y|x, w(t)) = \frac{1}{K} \sum_{k=1}^{K} P(y|x, w_k)
\]

In the case of MC-Dropout, \( w_k \) is the parameter of the neural network that is changed by the dropout operation at the \( k \)-th feedforward step. In the case of a deep ensemble, \( w_k \) is the parameter of the \( k \)-th neural network.

MC-Dropout and the deep ensemble can use Bayesian active learning by disagreement (BALD) [63] as the uncertainty estimation function in addition to entropy. Entropy and BALD are the two most popular acquisition functions in active learning-based uncertainty sampling. BALD determines the mutual information content between data points and weights \( w_k \). This measure is the entropy of the probability vector output by the classification model minus the average conditional entropy for a given weight. The BALD function is as follows.

\[
u_{BALD}(x) = u_H(x) - \lambda(x)
\]

\( \lambda(x) \) is the average conditional entropy for a given weight, which is given as follows.

\[
\lambda(x) = \frac{1}{K} \sum_{k=1}^{K} \sum_{c=1}^{C} -P(y = c|x, w_k) \log P(y = c|x, w_k)
\]

**V. EXPERIMENTAL SETTINGS**

To evaluate the proposed system, we develop a real dataset consisting of time-stamped and labeled signatures with the help of experts. We also define our evaluation metric that matches the problem of classifying signatures in the real world with the imbalance between classes.

**A. DATA COLLECTION**

Table 1 shows the distribution of the classes by time step \( t \). Signatures are classified into one of three importance levels, low, medium, or high, and assigned a time step \( t \). The signatures and their labels were collected on a monthly basis for two years. These signatures are distributed periodically by an IDPS developer. Some signatures can be automatically classified using if-then rules, but these signatures are thinned out in advance. In other words, the dataset consists only of signatures that experts have manually labeled based on their knowledge and experience.

Signatures are labeled by importance, but all classes are treated equally in this study. Suppose that “high” importance corresponds to the IDPS setting “block” and that “low” importance corresponds to “logging”. Classifying a
signature labeled “high” as “low” would allow communications that should be blocked to pass through. Naturally, experts would prefer not to do this, as it could cause a security incident. Similarly, mistaking “low” for “high” should also be avoided. This mistake can block communications that should otherwise be allowed to pass through unimpeded. This prevents the network from providing proper communications to its users. Which of the two error types is more important depends on the operational policy of the experts. Our research is conducted from the standpoint of not emphasizing any particular policy.

B. AN EVALUATION METRIC
In this section, we describe our evaluation metric. The typical top-1 classification accuracy and BACC measures, which do not consider the timing of data generation, are not appropriate for measuring the degree to which the problem in this paper is solved. We define our metric, which takes the following two points into account. The system prompts experts to label some signatures, retains them, and classifies the rest of the signatures, repeating the sequence in discrete time order. The class distribution imbalance is shown in Table 1.

The simulations evaluate the proposed system on a set of expert-labeled datasets \( D = \{ D^{(0)}, D^{(1)}, \ldots, D^{(i)}, \ldots, D^{(T)} \} \), \( D^{(i)} = \{ x_i^{(t)}, y_i^{(t)} \}_{i=1}^{N^{(i)}} \) is a dataset consisting of the signatures generated at time \( t \) and their labels. Let \( X^{(i)} = \{ x_i^{(t)} | x_i^{(t)} \in X^{(i)} \} \) be a subset of \( X^{(i)} \) that consists only of signatures labeled \( y \).

We define the co-BACC (CO-BACC) as an evaluation metric for this problem as follows:

\[
\text{CO-BACC}(t; s^{(i)}) = \frac{1}{|Y|} \sum_{y \in Y} \frac{\sum_{i'=1}^{i} \sum_{x \in X_{y}^{(i')}} \beta(x, y, t)}{\sum_{i'=1}^{i} |X_{y}^{(i')}}
\]

\[
\beta(x, y, t) = \min(1(y = y) + 1(u(x) > y^{(i)}), 1)
\]

The CO-BACC is a metric to be maximized, and it is calculated given \( s^{(i)} \), the number of signatures transferred to the experts at each time step \( t \). Through \( t = 1, \ldots, T \), we assume that the signatures transferred to the experts are correct. This evaluation metric was inspired by the accuracy-rejection curve (ARC) used in the reject option [64]. The ARC allows us to visualize the tradeoff between the rejection rate and the classification accuracy when the rejected samples are considered correct. The classification accuracy of the ARC does not account for imbalance, but our CO-BACC is developed with this idea in mind.

One experiment is a simulation in which an expert and a classification model collaborate to classify signatures while \( t = 1, \ldots, 23 \). The CO-BACC is then calculated and plotted at each time step \( t \). Let \( r \) be the acquisition rate for each step \( t = 1, \ldots, 23 \). We perform a simulation for each \( r = 10\%, 11\%, \ldots, 50\% \) and calculate the corresponding CO-BACC. If the number of samples to be acquired is not divisible, the decimal point is rounded down. For example, in the case of \( r = 30\% \), \( s^{(1)} = \lfloor 283 \times 30\% \rfloor = 84 \). The experiment is run 50 times with the same parameters, and the average value is used as the result.

C. COMPARATIVE METHODS
The classification model in this experiment is implemented based on a neural network as a machine learning model. The architecture of the neural network is a three-layer multilayer perceptron (MLP) with the following hyperparameters. A three-layer structure with an intermediate layer contained 100 nodes is trained using the error backpropagation method. The activation function for all nodes is a rectified linear unit (ReLU; ramp function). L2 regularization is used to suppress overlearning. The regularization parameter is set to 0.0001. The optimization method is adaptive moment estimation (Adam) with default parameters (\( \alpha = 0.0001, \beta_1 = 0.9, \beta_2 = 0.99, \text{and } \epsilon = 10^{-8} \) [65]. The neural network training process is terminated when the loss value for the training data is less than 0.0001 from the minimum value at least 10 times (early stopping). The maximum number of epochs is set to 200.

In our experiment, the following six systems are implemented and compared to confirm the effectiveness of the proposed system.

- **MLP-Random**: A neural network is used alone as the classification model, and random sampling is applied as the uncertainty estimation function.
- **MLP-Entropy**: A neural network is used alone as the classification model, and entropy is used as the uncertainty estimation function.
- **DE-Entropy**: A deep ensemble is employed as the classification model, and entropy is used as the uncertainty estimation function.
- **DE-BALD**: A deep ensemble is employed as the classification model, and BALD is used as the uncertainty estimation function.
- **MCD-Entropy**: MC-Dropout is used when inferring neural networks, and entropy is used as the uncertainty estimation function.
- **MCD-BALD**: MC-Dropout is used when inferring neural networks, and BALD is used as the uncertainty estimation function.

The deep ensemble has 100 members. All hyperparameters are common among them, and the only randomness concerns the initial values of the weights and the choices of minibatches. The MC-Dropout probability is set to 0.5, and the feedforward step is performed 100 times.

VI. EXPERIMENTAL RESULTS
In this section, we conduct experiments with the abovementioned datasets, evaluation metrics, and six systems and make comparisons. The first two simplest systems (MLP-Random and MLP-Entropy) are compared. Next, we validate the proposed improvements yielded by introducing the uncertainty estimation method to deep learning (DE-Entropy, DE-BALD, MCD-Entropy and MCD-BALD).
A. COMPARING MLP-RANDOM AND MLP-ENTROPY

From left to right, Fig. 5 shows the results obtained in the experiment with acquisition rates of 10%, 30% and 50%. The horizontal axis is $t$, and the vertical axis shows the CO-BACC at the corresponding step $t$. At all acquisition rates, MLP-Entropy outperforms MLP-Random in most of the steps $t$. In particular, MLP-Entropy outperforms MLP-Random even when the amount of training data is small and $t$ is small. MLP-Entropy can efficiently transfer samples to experts that are either appropriate for the training data or prone to errors.

Fig. 6 shows the final CO-BACC values obtained for all acquisition rates tested. The acquisition rate leads to the number of samples submitted to the expert at each step; i.e., it represents the size of the cost paid by the expert. It is an essential aspect of whether the proposed system works at any of the acquisition rates. In particular, MLP-Entropy outperforms MLP-Random even when the amount of training data is small and $t$ is small. MLP-Entropy can efficiently transfer samples to experts that are either appropriate for the training data or prone to errors.

In summary, MLP-Entropy works well in online signature classification tasks. Performing active learning with a simple neural network and the most primitive uncertainty sampling function as the acquisition function is effective for real-world situations.

B. VERIFICATION OF THE PROPOSED IMPROVEMENTS

We incorporate MC-Dropout and the deep ensemble into the proposed system to verify the resulting performance improvements. Fig. 7 shows the experimental results in the same style as Fig. 5 but with a different system described. Fig. 8 also shows the final CO-BACCs, as in Fig. 6. It can be seen that the performance of the combined MCD systems, especially MCD-BALD, is generally higher than that of the combined DE systems and MLP-Entropy. MCD-BALD achieves a CO-BACC increase under relatively low numbers of transfers to experts with acquisition rates of 10% and 30% and at earlier stages in the time series. Eventually, the CO-BACCs of many systems converge to similar values, but MCD-BALD appears to be slightly dominant (Fig. 8).

In [57], image recognition benchmarks were used, and deep ensembles performed better than MC-Dropout. In our experiments, however, MC-Dropout is preferred. The image recognition and signature classification tasks differ significantly in the architecture of the neural network used for the classification model. The former uses a large neural network such as a convolutional neural network (CNN) to capture the complexity of images, while the latter uses a simple MLP. Inference with MC-Dropout can be interpreted as inference with different neural networks, i.e., pseudo-deep ensembles. Although diversity among the neural network members is considered essential for deep ensembles [66], [67], MC-Dropout exhibits more diversity under the conditions of this experiment. This may be due to the positive effect of MC-Dropout on uncertainty sampling for relatively small neural networks. We suspect that the simple MLP, which classifies signatures, has fewer parameters than the CNN and thus cannot exhibit the diversity that results from random initial values. MC-Dropout performs...
VII. DISCUSSION
The analysis focuses on MC-Dropout, which is the best-performing method. All values observed in the analysis are averages obtained over 50 trials.

A. CLASS DISTRIBUTION OF THE ACQUIRED SAMPLES
Since the experimental results show that MCD-BALD is superior, we analyze its behavior. First, we check the distribution of the importance levels of the acquired samples through simulations. We compare the MLP-Entropy values, MCD-BALD values and expected values (Random) for \( r = 10\%, 30\%, 50\% \). The results are shown in Table 2. MCD-BALD acquires more samples with “medium” labels than the other approaches. The “medium” sample is the second minority class and is harder to predict as “medium” than the majority class (“low”). The acquisition of these samples and their forwarding to the experts may reduce the number of errors. The dataset has three importance labels: “low”, “medium”, and “high”, which are ordered. The “medium” label is in the middle of these three labels and may be uncertain for the classification model since its importance is uncertain even for experts.

B. TIME SERIES IN THE DISTRIBUTION
In the training dataset
Imbalance within the training dataset is an undesirable property that increases the concern that the classification model learns to neglect minority classes. The class distribution in the training dataset for each time step \( t \) reveals that the proposed system mitigates the imbalance issue. Fig. 9 shows the training dataset distributions observed over time for \( r = 10\%, 30\%, 50\% \). The vertical axis indicates the percentage of signatures with “medium” or “high” importance in the dataset, and the horizontal axis indicates the time step. The green lines indicate the expected values of the class distribution when the signatures are transferred randomly. MLP-Entropy (blue lines) and MCD-BALD (red lines) also have roughly higher percentages of minority samples than the random acquisition strategy. This trend is particularly pronounced in the latter half of the period, indicating that the imbalance issue is rapidly dissipating.
VIII. CONCLUSION
To classify signatures generated over time, we proposed a system in which a human and a machine learning model cooperate by applying active learning. To verify the performance of the proposed system, a real dataset was collected over two years with the help of experts. The real dataset has an imbalanced class distribution. We defined the CO-BACC as a performance measure that considers the class imbalance issue, the time series of the given signatures, and the behavior of the experts asked to make classification decisions. Experiments conducted on this real dataset showed that the attained CO-BACC was higher with active learning-based uncertainty sampling than with random signature sampling. Furthermore, it was confirmed that incorporating MC-Dropout, a deep learning-based uncertainty estimation method, into the proposed system further yielded improved performance. The analysis showed that incorporating MC-Dropout resulted in the transfer of more signatures with “medium” importance to the experts and a gradual alleviation of the imbalanced nature of the training data.

In this paper, we applied active learning to the signature classification task, and the system is independent of the structure of the data. There are other tasks in network security operations where data are generated periodically and classified by experts. For example, software vulnerability information, such as CVE, is released periodically, and experts may decide whether to classify this information as necessary. The proposed system can be applied to such tasks and can be widely applied to other tasks as well. We hope that the ideas and validation results in this paper will help solve signature classification problems as well as other tasks.

REFERENCES
[1] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, “Intrusion detection by machine learning: A review,” Exp. Syst. Appl., vol. 36, no. 10, pp. 11994–12000, 2009.
[2] A. L. Buczak and E. Guven, “A survey of data mining and machine learning methods for cyber security intrusion detection,” IEEE Commun. Surveys Tuts, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.
[3] S. Das and M. J. Nene, “A survey on types of machine learning techniques in intrusion prevention systems,” in Proc. Int. Conf. Wireless Commun., Signal Process. Netw. (WiSPNET), Mar. 2017, pp. 2296–2299.
[4] H. Kawaguchi, Y. Nakatani, and S. Okada, “IDPS signature classification with a reject option and the incorporation of expert knowledge,” 2022, arXiv:2207.10797.
[5] B. Settles, “Active learning literature survey,” Comput. Sci. Univ. Wisconsin–Madison, Madison, WI, USA, Tech. Rep. 1648, 2009.
[6] C. K. Chow, “An optimum character recognition system using decision functions,” IRE Trans. Electron. Comput., vol. EC-6, no. 4, pp. 247–254, Dec. 1957.
[7] C. Chow, “On optimum recognition error and reject tradeoffs,” IEEE Trans. Inf. Theory, vol. IT-16, no. 1, pp. 41–46, Jan. 1970.
[8] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, M. Shahzad, W. Yang, R. Bamler, and X. X. Zhu, “A survey of uncertainty in deep neural networks,” 2021, arXiv:2107.03342.
[9] Y. Gal and Z. Ghahramani, “Dropout as a Bayesian approximation: Representing model uncertainty in deep learning,” in Proc. Int. Conf. Mach. Learn. (ICML), 2016, pp. 1050–1059.
[10] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017, pp. 6402–6413.
[11] K. Alsubhi, E. Al-Shaer, and R. Boutaba, “Alert prioritization in intrusion detection systems,” in Proc. IEEE Netw. Oper. Manage. Symp. (NOMS), Apr. 2008, pp. 33–40.
[12] T. Piatrzeszek, “Using adaptive active classification to reduce false positives in intrusion detection,” in Recent Advances in Intrusion Detection (Lecture Notes in Computer Science), E. Jonsson, A. Valdes, M. Almgren, Eds. Berlin, Germany: Springer, 2004, pp. 102–124.
[13] N. Stakhonava and A. A. Ghorbani, “Managing intrusion detection rule sets,” in Proc. 3rd Eur. Workshop Syst. Secur. (EUROSEC), Berlin, Germany: Springer, 2004, pp. 102–124.
[14] F. Massicotte and Y. Labiche, “An analysis of signature overlaps in intrusion detection systems,” in Proc. IEEE IFIP 41st Int. Conf. Dependable Syst. Netw. (DSN), Jun. 2011, pp. 109–120.
[15] P. Ren, Y. Xiao, X. Chang, P.-Y. Chang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, “A survey of deep active learning,” ACM Comput. Surv., vol. 54, no. 9, pp. 1–40, Oct. 2021.
[16] S. C. H. Ho, R. Jin, J. Zhu, and M. R. Lyu, “Batch mode active learning and its application to medical image classification,” in Proc. 23rd Int. Conf. Mach. Learn. (ICML), New York, NY, USA, 2006, pp. 417–424.
[17] L. Yang, Y. Zhang, J. Chen, S. Zhang, and D. Chen, “Suggestive annotation: A deep active learning framework for biomedical image segmentation,” in Medical Image Computing and Computer Assisted Intervention—MICCAI (Lecture Notes in Computer Science), Cham, Switzerland: Springer, Sep. 2017, pp. 399–407.
[18] J. Nam, S. Park, E. J. Hwang, J. Lee, K.-N. Jin, K. Lim, T. Yu, J. Sohn, S. Hwang, J. M. Goo, and C. M. Park, “Development and validation of deep learning-based automatic detection algorithm for malignant pulmonary nodules on chest radiographs,” Radiology, vol. 290, Sep. 2018, Art. no. 180237.
[19] J. Wang, Z. Chen, L. Wang, and Q. Zhou, “An active learning with two-step query for medical image segmentation,” in Proc. Int. Conf. Med. Imag. Phys. Eng. (ICMIPE), Nov. 2019, pp. 1–5.
[65] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. ICLR, Y. Bengio and Y. LeCun, Eds. 2015, pp. 1–15.

[66] F. Wenzel, J. Snoek, D. Tran, and R. Jenatton, “Hyperparameter ensembles for robustness and uncertainty quantification,” in Proc. NeurIPS, H. Larochelle, R. Marc’Aurelio, R. Hadsell, M.-F. Balcan, and H.-T. Lin, Eds. 2020, pp. 6514–6527.

[67] S. Zaidi, A. Zela, T. Elsken, C. C. Holmes, F. Hutter, and Y. Teh, “Neural ensemble search for uncertainty estimation and dataset shift,” in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2021, pp. 7898–7911.

HIDETOSHI KAWAGUCHI received the B.S. degree in informatics from the Hiroshima Institute of Technology, in 2014, and the M.S. degree in engineering from the Tokyo Institute of Technology, in 2016. He is currently pursuing the Ph.D. degree in applying machine learning to network operation systems with the Japan Advanced Institute of Science and Technology (JAIST). Since 2016, he has been engaged in researching machine learning and developing machine learning systems with NTT. He is a member of the Information Processing Society of Japan (IPSJ) and the Japanese Society for Artificial Intelligence (JSAI).

YUICHI NAKATANI received the B.S. and M.S. degrees in informatics from Kyoto University, in 2003 and 2005, respectively. Since 2005, he has been engaged in researching distributed systems and network security and developing network services with NTT. He is a member of IEICE.

SHOGO OKADA (Member, IEEE) received the Ph.D. degree from the Tokyo Institute of Technology, Japan, in 2008. He directs the Social Signal and Interaction Group, Japan Advanced Institute of Science and Technology (JAIST), Japan, where he is currently an Associate Professor. In 2008 and 2011, he joined the Tokyo Institute of Technology, Kyoto University, as an Assistant Professor. He joined the IDIAP Research Institute, Switzerland, as a Visiting Faculty Member, in 2014. His research interests include social signal processing, human dynamics, multimodal interaction, and machine learning. He is a member of the ACM.

***