Automated Information Leakage Detection: A New Method Combining Machine Learning and Hypothesis Testing with an Application to Side-channel Detection in Cryptographic Protocols

Pritha Gupta[1][a], Arunselvan Ramaswamy[1][b], Jan Peter Drees[2][c], Eyke Hüllermeier[3][d], Claudia Priesterjahn[4][e] and Tibor Jager[2][f]

[1] *Department of Computer Science, Paderborn University, Warburger Str. 100, 33098 Paderborn, Germany*

[2] *Department of Computing, University of Wuppertal, Gaußstraße 20, 42119 Wuppertal, Germany*

[3] *Institute of Informatics, University of Munich (LMU), Akademiestr. 7, 80538 Munich, Germany*

[4] *achelos GmbH, Vattmannstraße 1, 33100 Paderborn, Germany*

Keywords: Information Leakage, Side-channel Attacks, Statistical Tests, Supervised Learning.

Abstract: Due to the proliferation of a large amount of publicly available data, information leakage (IL) has become a major problem. IL occurs when secret (sensitive) information of a system is inadvertently disclosed to unauthorized parties through externally observable information. Standard statistical approaches estimate the mutual information between observable (input) and secret information (output), which tends to be a difficult problem for high-dimensional input. Current approaches based on (supervised) machine learning using the accuracy of predictive models on extracted system input and output have proven to be more effective in detecting these leakages. However, these approaches are domain-specific and fail to account for imbalance in the dataset. In this paper, we present a robust autonomous approach to detecting IL, which blends machine learning and statistical techniques, to overcome these shortcomings. We propose to use Fisher's Exact Test (FET) on the evaluated confusion matrix, which inherently takes the imbalances in the dataset into account. As a use case, we consider the problem of detecting padding side-channels or ILs in systems implementing cryptographic protocols. In an extensive experimental study on detecting ILs in synthetic and real-world scenarios, our approach outperforms the state of the art.

1 INTRODUCTION

Information leakage (IL) is termed as the unintended disclosure of the sensitive information to an unauthorized person or an eavesdropper via observable system information (Hettwer et al., 2020). Detecting these ILs is crucial since they can cause electrical blackouts, theft of valuable and sensitive data like medical records and national security secrets (Hettwer et al., 2020). The task of detecting IL in a given system is called information leakage detection (ILD).

A system, intentionally or inadvertently, releases huge amounts of information publicly, which can be recorded by any outside observer called the *observable information (data)*. Precisely, IL occurs in this system if the observable information is directly or indirectly correlated to secret information (secret keys, plaintexts) of the system, which may result in compromising the security. Most current approaches based on statistics estimate the mutual information between observable and secret information to quantify IL. However, these estimates suffer from the problem of the *curse of dimensionality* of inputs (observable information) and in addition, strongly rely upon time-consuming manual analysis by domain experts (Chatzikokolakis et al., 2010). Current state-of-the-art research focuses on the development and application of machine learning for ILD since they have been proven to be more effective (Moos et al., 2021; Mushtaq et al., 2018). These approaches analyze the

[a] https://orcid.org/0000-0002-7277-4633
[b] https://orcid.org/0000-0001-7547-8111
[c] https://orcid.org/0000-0002-7982-9908
[d] https://orcid.org/0000-0002-9944-4108
[e] https://orcid.org/0000-0001-7236-9411
[f] https://orcid.org/0000-0002-3205-7699

Gupta, P., Ramaswamy, A., Drees, J., Hüllermeier, E., Priesterjahn, C. and Jager, T.
Automated Information Leakage Detection: A New Method Combining Machine Learning and Hypothesis Testing with an Application to Side-channel Detection in Cryptographic Protocols.
DOI: 10.5220/0010793000003116
In *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART 2022) - Volume 2*, pages 152-163
ISBN: 978-989-758-547-0; ISSN: 2184-433X

accuracy of the supervised learning model on the data extracted from the given system, by using the observable information as input and the secret information as the output (Moos et al., 2021). However, these approaches are domain-specific and not equipped to handle the imbalanced datasets, making them possibly miss novel ILs (false negatives) or detect non-existent leakage in the system (false positives) (Picek et al., 2018). Our approach tackles this problem by using the weighted version of supervised learning algorithms and the evaluated confusion matrix to detect the IL which inherently takes the imbalance in the dataset into account (Hashemi and Karimi, 2018).

As a use-case, we consider the problem of side-channel detection in cryptographic systems, which is an application of ILD. A cryptographic system unintentionally releases the *observable information* via many modes, such as network messages, CPU caches, power consumption, or electromagnetic radiation. These modes are exploited by the side-channel attacks (SCAs) to reveal the secret inputs (information) to an adversary, potentially rendering all implemented cryptographic protections irrelevant (Moos et al., 2021; Mushtaq et al., 2018). A system is said to contain a side-channel, if there exists a SCA which can reveal secret keys (or plain-texts), making it *vulnerable*. The existence of a side-channel in a cybersecurity system is equivalent to the occurrence of IL. In this field, the most relevant literature uses machine learning to perform SCAs, not preventing side-channels through early detection of ILs (Hettwer et al., 2020). Current machine learning-based approaches are able to detect side-channels, thus preventing SCA on the algorithmic and hardware levels and this has been presented by Zhang et al. (2020); Perianin et al. (2021); Mushtaq et al. (2018). These approaches apply the supervised-learning techniques using the observable system information as input to classify a system as vulnerable (with IL) or non-vulnerable (without IL). For generating the binary classification datasets, they extracted observable information from the secured systems as input, label them as 0 (non-vulnerable) and then introduce known ILs in these systems and label them 1 (vulnerable). This process makes these approaches domain-specific and misses novel side-channels (Perianin et al., 2021).

Current state-of-the-art automated learning-based approaches improve this by analyzing the accuracy of supervised-learning models on the binary classification data extracted from the given system, such that observable information is used as input and partial (sensitive) information is used as output (Moos et al., 2021; Drees et al., 2021). These approaches are restricted to detecting side-channels accurately, only if the extracted data is balanced, not noisy, and also produces a large number of false positives. The problem of imbalanced, noisy system datasets is very common in real-life scenarios (Zhang et al., 2020).

Our Contributions. We propose a novel approach that provides a general solution for detecting IL by testing the learnability of the binary classifiers on the extracted binary classification data from the system. To account for imbalance in the dataset we use weighted versions of the binary classifiers and test the evaluated confusion matrices using Fisher's Exact Test (FET) (Hashemi and Karimi, 2018). The FET inherently takes the imbalance into account by indirectly using the Mathews Correlation Coefficient evaluation measure, which is zero if the predictions are obtained by guessing the label (random guessing) or predicting the majority label (majority voting classifier) (Chicco et al., 2021). To account for the noise, we define an ensemble of binary classifiers which includes a Deep multi-layer perceptron and aggregate their FET results (p-values) to get the final verdict on the IL in a system. We show that our approach is more efficient (detection time) and accurate in detecting side-channels in real-world cryptographic OpenSSL TLS protocol implementations and ILs in synthetic scenarios as compared to the current state-of-the-art.

2 INFORMATION LEAKAGE: PROBLEM FORMULATION

In this section, we formalize the condition for occurrence information leakage (IL) in a given system using the binary classification problem described in the Appendix. We also define the information leakage detection (ILD) task using a mapping, which classifies the given system as vulnerable and non-vulnerable.

2.1 Information Leakage

IL occurs in a system (extracted data $\mathcal{D}$) when *observable information (data)* $\mathcal{X}$ is directly or indirectly correlated to secret information (secret keys, plain-texts) $\mathcal{Y}$ of the system, i.e., there is some information present in the $\mathcal{X}$ that can be used to derive the label y. The observable data is used as input ($\mathcal{X}$) and the secret information as the output ($\mathcal{Y}$) for the binary classification algorithms, which produces a mapping $\hat{g}$ between them, produced by Equation (5) defined in the Appendix.

In the following, we suggest using the *Bayes predictor* g^b, to check for dependencies (correlation) between $\mathcal{X}$ and $\mathcal{Y}$, thus for IL in a system. The (pointwise) *Bayes predictor* g^b minimizes the expected 0-1 loss (L_{01}) of the prediction $\hat{y}$ for given input $\boldsymbol{x}$:

$$\begin{aligned} g^b(\boldsymbol{x}) &= \arg\min_{\hat{y}\in\mathcal{Y}} \textstyle\sum_{y\in\mathcal{Y}} L_{01}(\hat{y},y)\, p(y|\boldsymbol{x}) \\ &= \arg\min_{\hat{y}\in\mathcal{Y}} E_y[L_{01}(\hat{y},y)|\boldsymbol{x}], \end{aligned} \tag{1}$$

where $E_y[L_{01}]$ is the expected 0-1 loss with respect to $y \in \mathcal{Y}$ and $p(y|\boldsymbol{x})$ is the conditional probability of the class y given an instance $\boldsymbol{x}$. If $\mathcal{X}$ and $\mathcal{Y}$ are independent of each other then, $p(y|\boldsymbol{x}) = p(y)$, such that $p(y)$ corresponds to the prior distribution of y, then g^b is defined as:

$$g^b(\boldsymbol{x}) = \arg\max_{y\in\mathcal{Y}} p(y) \tag{2}$$

For every point $\boldsymbol{x} \in \mathcal{X}$, g^b predicts label 0 if $p(1) < p(0)$ and label 1 if $p(1) > p(0)$, which implies the *Bayes predictor* is a majority voting classifier, when $p(y|\boldsymbol{x}) = p(y)$. Hence, if g^b produces an 0-1 loss less than the 0-1 loss of majority voting classifier, then there exists a dependency between $\mathcal{X}$ and $\mathcal{Y}$. For a known distribution $p(y|\boldsymbol{x})$, if g^b produces a loss (significantly) lower than that of a majority voting classifier, we imply that IL occurs in the system, else not. In reality, only $\mathcal{D}$ is available, so in place of Bayes Predictor, we use the *empirical risk minimizer* $\hat{g}$ produced by minimizing Equation (5). Using this, we quantify the IL in a system as the difference between average 0-1 loss ($1 - \mathrm{m}_{ACC}$) for $\hat{g}$ and majority voting classifier. If this difference is significant enough, then we conclude that IL occurs in the given system (that generates $\mathcal{D}$). This condition is the basis for our ILD approaches proposed in Section 3.

2.2 Information Leakage Function

The problem of ILD is reduced to analyzing the learnability of these binary classifiers on the given dataset. In a nutshell, we test this by hypothesizing that if the mapping produced by the supervised learning algorithm, accurately predicts the outputs using the inputs, then the correlation between the input and output is high, which implies the existence of IL in the system.

The task of an ILD approach is to assign a label to the extracted dataset $\mathcal{D}$ from a system, such that 0 indicates occurrence and 1 indicates the absence of IL in the system. Let $\mathcal{D}$ be the binary classification data extracted from the system, such that the observable information is represented by inputs $\mathcal{X} \subset \mathbb{R}^d$ and secret-information by outputs $\mathcal{Y} = \{0,1\}$. Given a dataset $\mathcal{D}$ of size N, the task of detecting IL boils down to associating $\mathcal{D}$ with a label in $\{0,1\}$, where 0 suggests "no information leakage" and 1 suggests "information leakage". Thus, we are interested in the function I defined as:

$$I : \bigcup_{N\in\mathbb{N}} (\mathcal{X}\times\mathcal{Y})^N \to \{0,1\}, \tag{3}$$

which takes a dataset $\mathcal{D}$ (extracted from the system) of any size as input and returns an assessment of the possible existence of IL in the given system as an output. We denote the mapping $\hat{I}$ as the predicted IL function produced by an ILD approach.

IL-Dataset. Let $\mathcal{L} = \{(\mathcal{D}_i, z_i)\}_{i=1}^{N_I}$ be the IL-Dataset, such that $N_I \in \mathbb{N}, z_i \in \{0,1\}, \forall i \in [N_I]$ Let $\boldsymbol{z} = (z_1, \ldots, z_{N_I})$ be the ground-truth vector, generated by the I, such that $z_i = I(\mathcal{D}_i), \forall i \in [N_I]$. Let $\hat{\boldsymbol{z}} = (\hat{z}_1, \ldots, \hat{z}_{N_I})$ be the corresponding prediction vector, such that $\hat{z}_i = \hat{I}(\mathcal{D}_i), \forall i \in [N_I]$. Since the ILD task produces binary decisions, their accuracy is measured using the binary classification evaluation metrics, e.g. accuracy for the ground-truth $\boldsymbol{z}$ and predictions $\hat{\boldsymbol{z}}$ is calculated using $\mathrm{m}_{ACC}(\boldsymbol{z}, \hat{\boldsymbol{z}})$ as described in the Appendix. To avoid confusion, we will refer to $\mathcal{L}$ as IL-Dataset and each $\mathcal{D}$ as the dataset.

3 INFORMATION LEAKAGE DETECTION: OUR APPROACHES

In this section, we describe our proposed information leakage detection (ILD) approaches using binary classification and statistical tests as shown in Figure 1. In addition, we describe an aggregation method based on using a set of binary classifiers and Holm-Bonferroni Correction to make our approaches more robust.

3.1 Paired T-Test based Approach

From the machine learning perspective, information leakage (IL) occurs in a system, if a binary classification algorithm trained on the dataset extracted from the system produces an accurate mapping between input (observable information) and output (secret information). This accuracy should be significantly better than that of the *Bayes predictor* evaluated on the dataset extracted from a secure system. For such systems, the inputs and outputs are independent of each other and *Bayes predictor* becomes majority voting classifier as per Equation (2) defined in the Appendix.

This provides us motivation to use paired statistical tests between the performance estimates, i.e., K accuracies obtained from K-Fold cross-validation (KFCV) of the binary classifier and majority voting classifier. These tests examine the probability (p-value) of observing the statistically significant difference between the paired samples (accuracies of majority voting classifier and binary classifier) (Demšar, 2006). The p-value is the probability of obtaining

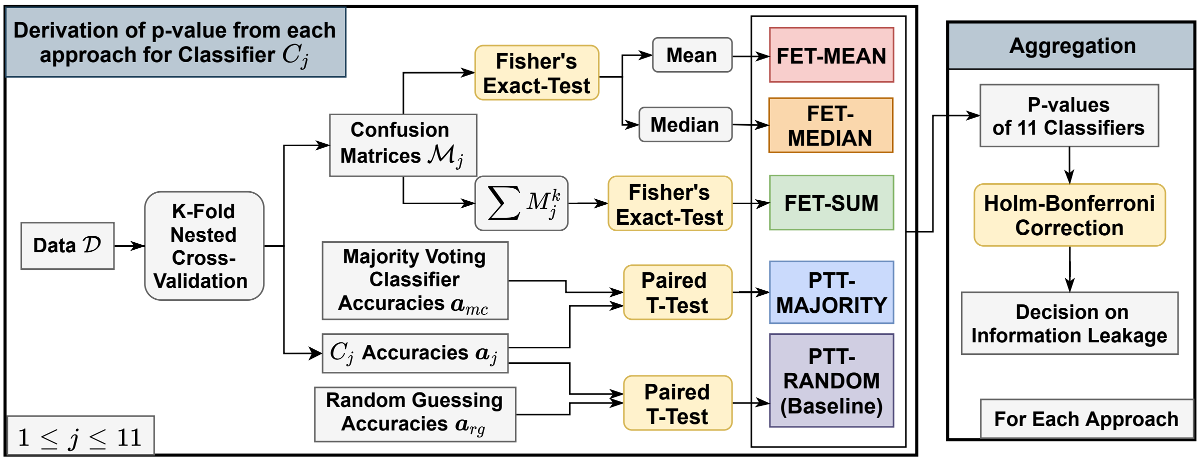
Derivation of p-value from each approach for Classifier C_j
Data $\mathcal{D}$
K-Fold Nested Cross-Validation
Confusion Matrices $\mathcal{M}_j$
Fisher's Exact-Test
Mean
Median
$\sum M_j^k$
Fisher's Exact-Test
Majority Voting Classifier Accuracies a_{mc}
C_j Accuracies a_j
Random Guessing Accuracies a_{rg}
Paired T-Test
Paired T-Test
FET-MEAN
FET-MEDIAN
FET-SUM
PTT-MAJORITY
PTT-RANDOM (Baseline)
$1 \leq j \leq 11$
Aggregation
P-values of 11 Classifiers
Holm-Bonferroni Correction
Decision on Information Leakage
For Each Approach

Figure 1: Approaches for Information Leakage Detection.

test results (mean of the difference between the accuracies) at least as extreme as the observation, assuming that the null hypothesis (H_0) is true (Demšar, 2006). The null hypothesis H_0 states that the accuracies are drawn from the same distribution (no difference in performance) or the average difference between the paired samples drawn from the two populations is zero(~ 0) (Demšar, 2006).

Out of many paired statistical tests, the most commonly used paired tests are the Paired T-Test (PTT) and the Wilcoxon-Signed Rank test (Demšar, 2006). These tests assume that each accuracy estimate is independent of each other and using KFCV violates the assumption of independence (the training data across each fold overlaps). Nadeau and Bengio (2003) showed that the violation of independence in the paired tests leads to overestimating the t statistic, resulting in tests being optimistically biased. We consequently choose to use their corrected version of the PTT, because it accounts for the dependency in KFCV, as explained in the Appendix.

The shortcoming of PTT is their asymptotic nature and the assumption that the samples (difference between the accuracies) are normally distributed, which results in optimistically biased p-values. This approach is based on accuracy, which is an extremely misleading metric for imbalanced classification datasets (Powers, 2011; Picek et al., 2018).

3.2 Fisher's Exact Test based Approach

To address the problem of class-imbalance and incorrect p-value estimation, we propose to use Fisher's Exact Test (FET) on the evaluated K confusion matrices (using KFCV) to detect IL.

IL is likely to occur if there exists a (sufficiently strong) correlation between the inputs $\boldsymbol{x}$ and the outputs y in the given data $\mathcal{D}$. The predictions produced by the classifier C_j is defined as $\hat{g}(\boldsymbol{x}) = \hat{y}$, where $\hat{g}$ is the *predicted function* as per Equation (5) defined in the Appendix. So, $\hat{y}$ is seen as single point encapsulating the complete information contained in the input $\boldsymbol{x}$. If there exists a correlation, then $\hat{y}$ will contain input information that is relevant/used for predicting the correct outputs (TP, TN). We can therefore determine the existence of IL by examining the dependency between predictions $\hat{\boldsymbol{y}}_j$ of C_j and the ground-truths $\boldsymbol{y}$.

We proposed to apply FET on the confusion matrix for calculating the probability of independence between the model predictions $\hat{\boldsymbol{y}}$ and the actual labels $\boldsymbol{y}$ (Fisher, 1922). FET is a non-parametric test that is used to calculate the probability of independence (non-dependence) between two classification methods, in this case, classification of instances according to ground-truth y and the binary classifier predictions $\hat{y}$ (Fisher, 1922). The null hypothesis H_0 states that the model predictions $\hat{y}$ and ground-truth labels y are independent, implying absence of IL. While the alternate hypothesis H_1 states that the model predictions $\hat{y}$ are (significantly) dependent on the ground-truth labels y, implying occurrence of IL.

The advantage of using FET is that the p-value is calculated using Hypergeometric distribution exactly, rather than relying on an approximation that becomes exact in the limit with sample size approaching infinity, as is the case for many other statistical tests. In addition to that, this approach directly tests the learnability of a binary classifier (without considering majority voting classifier) and is indirectly proportional to the Mathews Correlation Coefficient performance measure, which takes class-imbalance in the dataset into account (Camilli, 1995; Chicco et al., 2021). Please refer to the Appendix for details.

3.3 On Robustness

For making our ILD approaches more robust, we define a set of 11 binary classifiers $\mathcal{C}$ that are evaluated on the extracted dataset $\mathcal{D}$ of the given system.

The motivation of using an ensemble of binary classifiers, rather than just one binary classifier is that each binary classifier restricts their hypothesis space $\mathcal{H}$, based on the assumptions imposed on h. In addition, the statistical tests are asymmetric in nature, i.e. they can only be used to reject H_0 which implies that we can prove the existence of IL but not its absence. To work around both restrictions, we use a set of multiple binary classifiers ($\mathcal{C}, |\mathcal{C}| = 11$) to detect IL more reliably inculcating greater trust in the absence of IL, if all classifiers fail to find an accurate matching.

The ***set of binary classifiers*** $\mathcal{C}$ ($|\mathcal{C}| = 11$), includes simple and commonly used linear classifiers, which assume linear dependencies between the input and output, such as Perceptron, Logistic Regression, and Ridge Classifier. $\mathcal{C}$ also includes Support Vector Machine, which classifies the non-linearly separable data using a *kernel trick*, i.e. its *hypothesis space* also contains non-linear functions. $\mathcal{C}$ also includes Decision Tree and Extra Tree, which learn a set of rules using a tree for classification (Geurts et al., 2006). To learn more complex dependencies with very high accuracy, we also include *ensemble* based binary classifiers in $\mathcal{C}$. The *ensemble* based approaches train multiple binary classifiers (*base learners*) on the given dataset and the final prediction is obtained by aggregating the predictions of each *base learner*. The two most popular approaches proposed to build a diverse ensemble of learned *base learners* are *bagging* and *boosting* (Kotsiantis et al., 2006). To achieve diversification, *bagging* generates sub-sampled datasets from the given training dataset and *boosting* successively trains a set of weak learners (Decision Stumps) and at each round, more weight is given to the previously misclassified instances (Kotsiantis et al., 2006). The bagging-based approaches included in $\mathcal{C}$ are Extra Trees (mean aggregation) and Random Forest (majority voting aggregation). We choose Ada Boost and Gradient Boosting from the available boosting-based approaches (Kotsiantis et al., 2006). We also include a Deep multi-layer perceptron, as they are the universal approximators, i.e. in theory they can approximate any continuous function between input and the output (Cybenko, 1989).

For evaluation of each binary classifier C_j, we use nested KFCV with hyperparameter optimization to get K unbiased estimates of accuracies and confusion matrices (Bengio and Grandvalet, 2004). As shown in Figure 1, for each binary classifier $C_j \in \mathcal{C}$, $\boldsymbol{a}_j = (a_{j1}, \ldots, a_{jK})$ denotes the K accuracies and $\mathcal{M}_j = \{M_j^k\}_{k=0}^K$ denotes set of K confusion matrices.

PTT-MAJORITY. is our proposed PTT based approach, which compares the accuracy of majority voting classifier ($\boldsymbol{a}_{mc}$) with that of the binary classifier C_j ($\boldsymbol{a}_j$). We denote the baseline approach proposed in Drees et al. (2021) as **PTT-RANDOM**, which uses PTT to test if the binary classifier C_j ($\boldsymbol{a}_j$) performs significantly better than random guessing ($\boldsymbol{a}_{rg}$).

We propose 3 FET-based approaches **FET-SUM**, **FET-MEAN** and **FET-MEDIAN**. We require additional aggregation techniques to get a final FET p-value, since we obtain K confusion matrices $\mathcal{M}_j$ for each binary classifier C_j. In KFCV the test dataset does not overlap for different folds, which implies that each confusion matrix M_j^k could be seen as an independent estimate. Since, FET is only applicable for 2×2 matrices containing natural numbers, we aggregate the confusion matrices using *sum* ($\sum_{k=1}^K M_j^k$ for classifier C_j) and apply FET to obtain the final p-value. We refer to this approach as **FET-SUM**. The second technique is to apply the FET on each confusion matrix $M_j^k \in \mathcal{M}_j$ to acquire K p-values and then aggregate them. Bhattacharya and Habtzghi (2002) showed that the *median* aggregation operator provides best estimation of the true p-value. We aggregate K p-values using the *median* operator and refer to this approach as **FET-MEDIAN** and We also propose to use the arithmetic *mean* operator to get the final p-value and refer to the approach as **FET-MEAN**.

We apply these approaches to each classifier $C_j \in \mathcal{C}$ and produce $|\mathcal{C}| = 11$ p-values as shown in Figure 1. To detect IL, we aggregate these p-values using the Holm-Bonferroni correction as described in the Appendix. Using this correction yields the value m, which denotes the number of binary classifiers for which the null hypothesis H_0 was rejected.

IL Detection. The sufficient condition for the existence of IL in the system is that even if one binary classifier is able to learn an accurate mapping between input and output, i.e. if $m = 1$ then IL occurs in the system. Using different types of binary classifiers and $m = 1$ makes our approach more general and can detect a diverse class of ILs in a system.

4 EMPIRICAL EVALUATION

In this section, we provide an extensive evaluation of our proposed approaches, detecting the information leakage (IL) in synthetic and real-world scenarios. In

Table 1: Overview of the IL-Datasets used for the experiments.

Scenario	Fixed Parameter	IL-Dataset $\mathcal{L}$ configuration			Binary Classification Dataset $\mathcal{D}$ configuration			
		# Systems $\lvert\mathcal{L}\rvert$	# $z=0$	# $z=1$	$\lvert\mathcal{D}\rvert$	# $y=0$	# $y=1$	# Features
Efficiency	$r=0.1$	20	10	10	$200\times K$	$180\times K$	$20\times K$	10
K for KFCV,	$r=0.3$	20	10	10	$200\times K$	$140\times K$	$60\times K$	10
$3\le K\le 30$	$r=0.5$	20	10	10	$200\times K$	$100\times K$	$100\times K$	10
Generalization	$K=10$	20	10	10	2000	$2000\times(1-r)$	$2000\times r$	10
Class-Imbalance parameter r	$K=20$	20	10	10	4000	$4000\times(1-r)$	$4000\times r$	10
$0.01\le r\le 0.51$	$K=30$	20	10	10	6000	$6000\times(1-r)$	$600\times r$	10
Configuration of the OpenSSL IL-Datasets								
OpenSSL0.9.7a (Vulnerable)	$r=0.1$	20	-	10	11124	10012	1112	88
OpenSSL0.9.7b (Non-Vulnerable)	$r=0.1$		10	-	11078	9971	1107	88
OpenSSL0.9.7a (Vulnerable)	$r=0.3$	20	-	10	14302	10012	4290	88
OpenSSL0.9.7b (Non-Vulnerable)	$r=0.3$		10	-	14244	9971	4273	88
OpenSSL0.9.7a (Vulnerable)	$r=0.5$	20	-	10	19991	10012	9979	88
OpenSSL0.9.7b (Non-Vulnerable)	$r=0.5$		10	-	19995	9971	10024	88

particular, we show that our approach outperforms the state-of-the-art, presented in Drees et al. (2021); Moos et al. (2021), with respect to detection accuracy, efficiency, and generalization capability with respect to class-imbalance in the datasets. We also describe the IL-Datasets used to illustrate our ideas.

4.1 Dataset Descriptions

As stated earlier, we need to generate a binary classification dataset from the system under consideration for IL detection. In this section, we describe the generation process for these datasets from synthetic and real-world systems. Recall from Section 3.3, that K refers to the number of folds of nested K-Fold cross-validation (KFCV) used for evaluation of binary classifiers. *Class-imbalance parameter* is the proportion of positive instances in a given dataset $\mathcal{D}$, defined as: $r = \frac{|\{(\boldsymbol{x}_i,y_i)\in\mathcal{D} \mid y_i=1\}|}{|\mathcal{D}|}$.

4.1.1 Synthetic Dataset Generation

For simulating a realistic leakage detection scenario, we generate synthetic binary classification datasets $\mathcal{D}$ from *vulnerable* and *non-vulnerable* systems, using Algorithm 1.

For each system, the inputs $\boldsymbol{x}$ of the datasets are produced using the d-dimensional ($d = 10$) multivariant normal distribution. To imitate a system containing IL (vulnerable), the corresponding labels (output) are produced using a function $y = f(\boldsymbol{x})$, which makes the output dependent on the inputs. For imitating system which does not contain IL (non-vulnerable), the corresponding labels (output) are produced using Bernoulli distribution, such that $y \sim \text{Bernoulli}(p = r, q = 1-r)$ (r: class-imbalance parameter), which makes the output (labels) independent of the inputs. Algorithm 1 generates balanced IL-Datasets, containing 10 datasets extracted from the vulnerable systems and 10 from the non-vulnerable systems, such that each $\mathcal{D}$ contains $200\times K$ instances with dimensionality 10, out of which $r\times 200\times K$ are labeled as 1. This makes sure that the number of instances used for evaluation of a binary classifier ($|\mathcal{D}|/K = 200$) is the same across different scenarios, making accuracies and confusion matrices estimates fair ($TP+FP+TN+FN = 200$). We implement Algorithm 1 by modifying the `make_classification` function by scikit-learn (Pedregosa et al., 2011).

Algorithm 1: Generate IL-Dataset $\mathcal{L}$ for given K, r.

```
1: Define $\mathcal{L} = \{\}$, $N = K\times 200$, $N_L$.
2: Sample weight-vector $\boldsymbol{\beta} \sim N(1,\sigma)$, $\sigma \sim [0,2]$
3: Define $\boldsymbol{\mu}$ as vertices of $d$-dimensional hypercube.
4: for $j \in \{2\times j-1\}_{j=1}^{N_L/2}$ do
5:     Draw i.i.d. samples $\boldsymbol{x}_i \sim N(\boldsymbol{\mu}, \boldsymbol{I}_d), \forall i \in [N]$
6:     Define $\mathcal{D}_j = \{\}$, $\mathcal{D}_{j+1} = \{\}$. $\{\mathcal{D}_j$: With IL, $\mathcal{D}_{j+1}$: No IL$\}$
7:     for $(i = 1; i <= N; i{+}{+})$ do
8:         Calculate score $s_i = \text{sigmoid}(\boldsymbol{x}_i \cdot \boldsymbol{\beta})$
9:         Label $y_i$: $y_i = [\![s_i < r]\!]$.
10:        $\mathcal{D}_j = \mathcal{D}_j \cup \{(\boldsymbol{x}_i, y_i)\}$ {Add instance}
11:    end for
12:    for $(i = 1; i <= N; i{+}{+})$ do
13:        Label $y_i$: $y_i \sim \text{Bernoulli}(p = r, q = 1-r)$.
14:        $\mathcal{D}_{j+1} = \mathcal{D}_{j+1} \cup \{(\boldsymbol{x}_i, y_i)\}$ {Add instance}
15:    end for
16:    $\mathcal{L} \cup \{(\mathcal{D}_j, 1), (\mathcal{D}_{j+1}, 0)\}$ {Add datasets}
17: end for
18: return $\mathcal{L}$
```

The main goal of our empirical evaluation is to analyze how our proposed information leakage detection (ILD) approaches perform compared to baselines in regards to *efficiency (detection time)* and *generalization* capability with respect to the class-imbalance parameter r .The total time taken by the

ILD approaches is linear w.r.t K ($|\mathcal{D}| = K \times 200$), i.e. $O(K)$. For analyzing the *efficiency*, we generate 28 IL-Datasets with each dataset $\mathcal{D}$ of size $|\mathcal{D}| = K * 200$, for value of K ranging from 3 to 30 and fixed class-imbalance $r = 0.1, 0.3, 0.5$, as detailed in Table 1. To gauge the *generalization* capability, we generate 25 IL-Datasets with each dataset $\mathcal{D}$ of size $|\mathcal{D}| = K * 200$ for fixed $K = 10, 20, 30$ with class-imbalance r varying between 0.01 and 0.51, as detailed in Table 1.

4.1.2 OpenSSL Dataset Generation

The real-world classification datasets are generated from the network traffic of 2 OpenSSL TLS servers, one of which is vulnerable (contains IL) and other being non-vulnerable (secure, does not contain IL).

In the use-case of side-channel detection, such datasets are already being used, so we can generate them using the automatic side-channel analysis tool[1] presented by (Drees et al., 2021). The tool uses a modified TLS client to send requests with manipulated padding to a TLS server. In this setting, IL occurs when an attacker can deduce the manipulation in the request simply by observing the server's reaction to the message. Therefore, the server's reaction is recorded as a network trace by the tool and exported to a labeled dataset suitable for classifier training. We need to determine which TLS server to use for the experiment, and the most widely used TLS server, OpenSSL, comes to mind. According to the OpenSSL changelog[2], a fix for the Klíma-Pokorny-Rosa (Klíma et al., 2003) bad version attack was applied on the OpenSSL TLS implementation in version 0.9.7b. Consequently, version 0.9.7a contained IL in the form of a bad version side-channel, while version 0.9.7b does not contain IL. This offers the opportunity to gather suitable datasets from these servers. To generate the dataset, we configure the modified TLS client to manipulate the TLS version bytes contained in the pre-master secret it sends to the OpenSSL server. For each handshake, the client flips a coin to either keep the correct TLS version in place or replace it with the non-existing "bad" version `0x42 0x42`.

In the resulting dataset, class label $y = 0$ is used for handshakes with correct TLS version and class label $y = 1$ for handshakes with "bad" (incorrect) TLS version. This handshake process is then repeated 20 000 times, with each handshake being extracted into a single instance in the dataset. This approach produces datasets with approximately 10 000 instances per class and produces almost balanced datasets, we refer to them as $r = 0.5$ in our experiments. In addition, we also generate datasets with class-imbalance $r = 0.3$ and $r = 0.1$, containing around 14 000 handshakes and 11 000 handshakes respectively. We generate IL-Datasets containing 10 datasets from 0.9.7a (with IL) and 10 datasets from 0.9.7b (without IL) as shown in Table 1. All real-valued features of the TLS and TCP layers in the messages sent by the server as a reaction to the manipulated TLS message are part of an instance in the dataset. This results in high dimensional datasets, containing 88 features and only a handful of which are actually correlated to the output.

[1]https://github.com/ITSC-Group/autosca-tool

[2]https://www.openssl.org/news/changelog.html

4.2 Implementation Details

The main goal of our empirical evaluation is to analyze how our proposed ILD approaches perform in comparison to the baselines in regards to *efficiency (detection time)*, *generalization* capability with respect to the class-imbalance parameter r, and overall ILD accuracy. Table 1 describes the IL-Datasets used for these experimental scenarios.

As a baseline, we refer to the methodology described in Drees et al. (2021) as PTT-RANDOM. Recall that this approach is similar to PTT-MAJORITY, with the difference that it uses random guessing rather than the majority voting classifier and a set of binary classifiers (ensemble) without Deep multi-layer perceptron. For a fair comparison, we use our defined set of binary classifiers described in Section 3.3. We also consider DL-LA as another baseline, which trains a Deep multi-layer perceptron on a balanced binary classification dataset and propose that if its accuracy is significantly greater than 0.5, then IL exists in the given system (Moos et al., 2021).

We apply nested KFCV with hyper-parameter optimization on $|\mathcal{C}| = 11$ binary classifiers as described in Section 3.3. To improve the accuracy of the binary classifiers for imbalanced datasets i.e., $r < 0.5$, we employ the weighted versions of binary classifiers described by Hashemi and Karimi (2018). They penalize the mis-classification of positive ($y = 1$) instances by $\frac{1}{r}$ and negative ($y = 0$) instances by $\frac{1}{1-r}$. For our experiments, the rejection criteria for statistical tests is set to 0.01, i.e. $\alpha = 0.01$, giving us 99% confident for our prediction. The binary classifiers, stratified KFCV, and evaluation measures were implemented using the scikit-learn (Pedregosa et al., 2011) and statistical tests using SciPy (Virtanen et al., 2020). The hyperparameters of each binary classifier were tuned using scikit-optimize (Head et al., 2021). The code for the experiments and the generation of plots with detailed documentation is publicly available on GitHub[3].

[3]https://github.com/prithagupta/ML-ILD

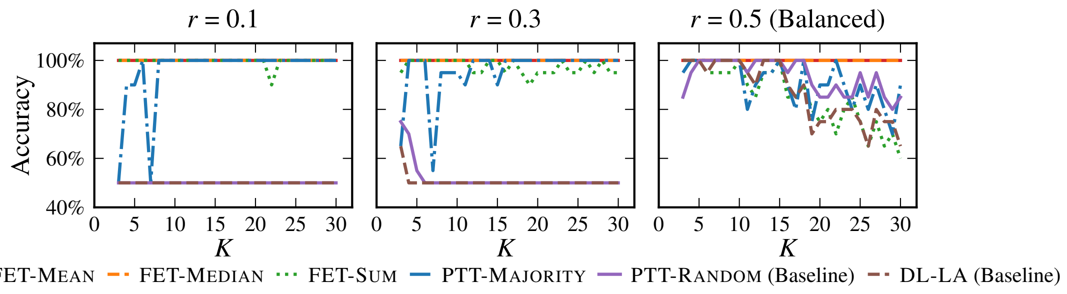

Figure 2: Accuracies of different detection approaches on Synthetic IL-Datasets evaluated using different K for KFCV.

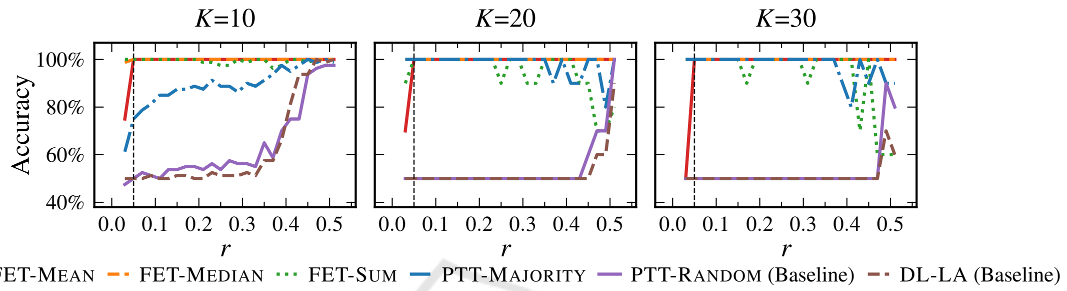

Figure 3: Accuracies of different detection approaches on Synthetic IL-Datasets containing datasets with different r.

4.3 Results

In this section, we discuss the results of the experiments outlined above. Recall that K is the number of folds used for conducting KFCV and r is the proportion of positive instances in the dataset $\mathcal{D}$. Each IL-Dataset contains binary classification datasets of size $200 \times K$. In Figures 2 to 4, we compare the performances (detection accuracy) of FET-Mean, FET-Median, FET-Sum, PTT-Majority with the baselines PTT-Random and DL-LA.

Efficiency. In Figure 2, the value K (KFCV) is varied between 3 and 30 and shown on the X-axis, and the resulting accuracy of the ILD approaches along the Y-axis. Additionally, we compare the performance for different choices of class-imbalance parameter r ($r = .1$, $r = .3$, and $r = .5$ (balanced)), producing three individual plots shown side-by-side.

Overall, we observe that the performance of FET-Mean and FET-Median ($\sim 100\%$) does not change with the value of K (number of estimates) for all three IL-Datasets, while FET-Sum is very unstable for balanced dataset $r = 0.5$ and slightly unstable for imbalanced datasets. The PTT-Majority approach outperforms the baselines, but there is no possible fixed value of K for which the detection accuracy is high regardless of the imbalance. A good choice to prevent unstable and inaccurate results would require a small value $K < 7$ for balanced datasets ($r = 0.5$) and a large value $K > 7$ for imbalanced datasets $r = 0.1, 0.3$, which is an issue in applications where the imbalance is not known in advance. The reason behind this could be that Paired T-Test (PTT) produces incorrect and optimistically biased p-values due to the low deviation in estimated accuracies of the majority voting classifier, as explained in the Appendix. Another interesting observation is that DL-LA performs similarly to PTT-Random, which could be because a Deep multi-layer perceptron can theoretically approximate any continuous function between input and output (Cybenko, 1989). Overall, we observed that Fisher's Exact Test (FET)-based approaches require a lower number of estimates K, thus being more efficient in detecting ILs.

Generalization w.r.t. r. In Figure 3, the value r (class-imbalance) is varied between 0.05 and 0.5 and shown on the X-axis, and the resulting accuracy of the ILD approaches along the Y-axis. Additionally, we compare the performance for different choices of K ($K = 10$, $K = 20$, and $K = 30$), producing three individual plots shown side-by-side.

The FET-based approaches perform very well for all datasets $r \geq 0.05$, even if the number of estimates is as low as $K = 10$. As before, PTT-Majority achieves a high detection accuracy for larger numbers of estimates $K = 20, K = 30$ only with high imbalance $0.05 \leq r < 0.3$, with deteriorating accuracy for $r \geq 0.3$. The baselines are not able to detect ILs in imbalanced datasets, as most of the binary classifiers easily achieve an accuracy of $1 - r$ (same as major-

Table 2: Results on the OpenSSL datasets for every approach using $K = 20$ and $m = 1$. The best entry is marked in bold.

	Class-Imbalance $r = 0.1$				Class-Imbalance $r = 0.3$				Class-Imbalance $r = 0.5$ (Balanced)			
Approach	FPR	FNR	ACCURACY	F1-SCORE	FPR	FNR	ACCURACY	F1-SCORE	FPR	FNR	ACCURACY	F1-SCORE
FET-MEAN	**0.0021**	**0.0**	**0.999**	**0.999**	**0.0182**	**0.0**	**0.9909**	**0.9911**	**0.0164**	**0.0**	**0.9918**	**0.9919**
FET-MEDIAN	**0.0021**	**0.0**	**0.999**	**0.999**	0.0455	0.0	0.9773	0.9778	0.0327	0.0	0.9836	0.9839
FET-SUM	0.66	0.0	0.67	0.7519	0.84	0.0	0.58	0.7043	0.9564	0.0	0.5218	0.6766
PTT-MAJORITY	0.0473	0.0	0.9764	0.9769	0.3309	0.0	0.8345	0.8585	0.3145	0.0	0.8427	0.8674
PTT-RANDOM (Baseline)	1.0	0.0	0.5	0.6667	1.0	0.0	0.5	0.6667	0.1836	0.0	0.9082	0.9179
DL-LA (Baseline)	1.0	0.0	0.5	0.6667	1.0	0.0	0.5	0.6667	0.4727	0.0	0.7636	0.8111

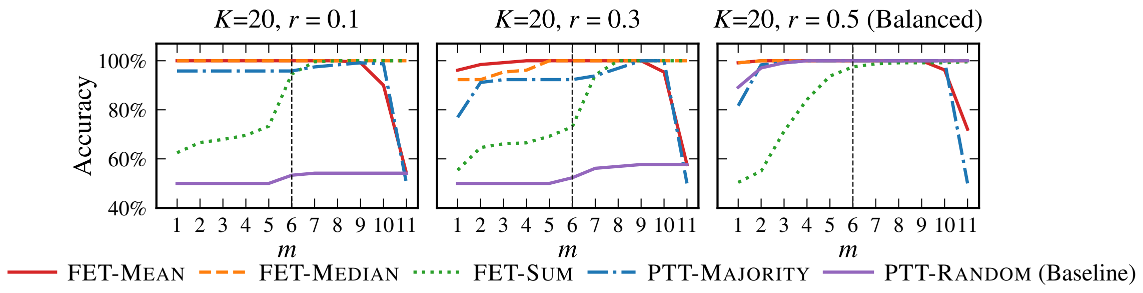

Figure 4: Accuracies of the detection approaches on OpenSSL Datasets with varying Holm-Bonferroni cut-off parameter m.

ity voting classifier) for them, always outperforming random guessing (0.5) even when there is no IL.

OpenSSL Dataset. In Table 2, we summarize the overall performance in terms of FPR, FNR, ACCURACY, and F1-SCORE of ILD approaches on the real datasets, fixing $K = 20$ based on the previous results. Overall FET-MEAN and FET-MEDIAN outperform other approaches in detecting side-channel in the OpenSSL case study. As expected the baselines work well for balanced datasets while failing to accurately detect side-channels for imbalanced datasets. The PTT-MAJORITY approach works well for imbalanced datasets but produces false positives for balanced datasets. The FET-SUM approach overestimates ILs in the servers, producing false positives for both balanced and imbalanced datasets.

In Figure 4 we also explore the performance of ILD approaches for different values of the Holm-Bonferroni parameter m on these datasets. These results exclude the DL-LA approach, as it does not use the Holm-Bonferroni correction. For lower values of m, some ILD approaches produce a large number of false positives, because the tests are underestimating the p-values. Conversely, for larger values of m, the ILD approaches produce many false negatives, because not all binary classifiers are able to learn an accurate enough mapping to warrant rejection of the null hypothesis. Based on the requirements at hand, it will therefore be necessary to tune m to achieve optimal performance. PTT-MAJORITY and FET-SUM are not reliable for estimating the correct p-values and produce large false positives, especially for $m < 4$. The overall performance of FET-MEDIAN and FET-MEAN is consistently very high (between 99% and 100%) for all choices of $m < 10$. FET-MEDIAN is even more versatile than FET-MEAN, almost always achieving an accuracy of 100%, because the median aggregation results in more accurate p-values (Bhattacharya and Habtzghi, 2002).

5 CONCLUSION

We presented a novel machine learning-based framework to detect the possibility of information leakage (IL) in a given system. For this, we first generated an appropriate (binary) classification dataset using its observable and secret system data. We then trained an ensemble of classification models using the dataset. We deduce the existence of IL when either the ensemble performance is significantly better than majority voting or using the more complex Fisher's Exact Test (FET) from statistics.

The major advantages of the presented approach over previous ones are: (a) it accounts for imbalances in datasets (b) it has a very low false positive rate (c) it is robust to noise in the generated dataset (d) it is time-efficient and still outperforms the state-of-the-art machine learning-based approaches. These advantages are partly due to our IL inference using the non-parametric FET, applied on the confusion matrix of the learning models, rather than direct IL inference using learning accuracies. The robustness, in particular, is a consequence of using the Holm-Bonferroni correction technique. We presented extensive empirical evidence for our claims and compared our approach to other baseline approaches, including a deep-learning-based one.

In the future, we aim to extend our work to detect new and unknown side-channels. We are also interested in exploring IL detection when the generated dataset yields a multi-class classification problem with ≥ 3 classes. This necessitates extending the presented FET approach to account for multi-class classification problems. Finally, we would like to provide appropriate theoretical backing for our approaches using information theory.

ACKNOWLEDGEMENTS

We would like to thank Björn Haddenhorst, Karlson Pfanschmidt, Vitalik Melnikov, and the anonymous reviewers for their valuable and helpful suggestions. This work is supported by the Bundesministerium für Bildung und Forschung (BMBF) under the project 16KIS1190 (AutoSCA) and funded by European Research Council (ERC)-802823.

REFERENCES

Bengio, Y. and Grandvalet, Y. (2004). No unbiased estimator of the variance of k-fold cross-validation. *Journal of Machine Learning Research*, 5:1089–1105.

Bhattacharya, B. and Habtzghi, D. (2002). Median of the p value under the alternative hypothesis. *The American Statistician*, 56(3):202–206.

Camilli, G. (1995). The relationship between fisher's exact test and pearson's chi-square test: A bayesian perspective. *Psychometrika*, 60(2):305–312.

Chatzikokolakis, K., Chothia, T., and Guha, A. (2010). Statistical measurement of information leakage. In Esparza, J. and Majumdar, R., editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 390–404, Berlin, Heidelberg. Springer Berlin Heidelberg.

Chicco, D., Tötsch, N., and Jurman, G. (2021). The matthews correlation coefficient (mcc) is more reliable than balanced accuracy, bookmaker informedness, and markedness in two-class confusion matrix evaluation. *BioData Mining*, 14(1):13.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1):1–30.

Drees, J. P., Gupta, P., Hüllermeier, E., Jager, T., Konze, A., Priesterjahn, C., Ramaswamy, A., and Somorovsky, J. (2021). Automated detection of side channels in cryptographic protocols: Drown the robots! In *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security*, AISec '21, page 169–180, New York, NY, USA. Association for Computing Machinery.

Fisher, R. A. (1922). On the interpretation of χ^2 from contingency tables, and the calculation of *P*. *Journal of the Royal Statistical Society*, 85(1):87–94.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine learning*, 63(1):3–42.

Hashemi, M. and Karimi, H. (2018). Weighted machine learning. *Statistics, Optimization and Information Computing*, 6(4):497–525.

Head, T., Kumar, M., Nahrstaedt, H., Louppe, G., and Shcherbatyi, I. (2021). scikit-optimize/scikit-optimize.

Hettwer, B., Gehrer, S., and Güneysu, T. (2020). Applications of machine learning techniques in side-channel attacks: a survey. *Journal of Cryptographic Engineering*, 10(2):135–162.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70.

Klíma, V., Pokorný, O., and Rosa, T. (2003). Attacking rsa-based sessions in ssl/tls. In Walter, C. D., Koç, Ç. K., and Paar, C., editors, *Cryptographic Hardware and Embedded Systems - CHES 2003*, pages 426–440, Berlin, Heidelberg. Springer Berlin Heidelberg.

Kotsiantis, S. B., Zaharakis, I. D., and Pintelas, P. E. (2006). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26(3):159–190.

Koyejo, O., Ravikumar, P., Natarajan, N., and Dhillon, I. S. (2015). Consistent multilabel classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, page 3321–3329, Cambridge, MA, USA. MIT Press.

Moos, T., Wegener, F., and Moradi, A. (2021). DL-LA: Deep Learning Leakage Assessment: A modern roadmap for SCA evaluations. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(3):552–598.

Mushtaq, M., Akram, A., Bhatti, M. K., Chaudhry, M., Lapotre, V., and Gogniat, G. (2018). Nights-watch: A cache-based side-channel intrusion detector using hardware performance counters. In *Proceedings of the 7th International Workshop on Hardware and Architectural Support for Security and Privacy*, HASP '18, New York, NY, USA. Association for Computing Machinery.

Nadeau, C. and Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, 52(3):239–281.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Perianin, T., Carré, S., Dyseryn, V., Facon, A., and Guilley, S. (2021). End-to-end automated cache-timing attack driven by machine learning. *Journal of Cryptographic Engineering*, 11(2):135–146.

Picek, S., Heuser, A., Jovic, A., Bhasin, S., and Regazzoni, F. (2018). The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2019:209–237.

Powers, D. M. (2011). Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1):37–63.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., et al. (2020). Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature Methods*, 17(3):261–272.

Zhang, J., Zheng, M., Nan, J., Hu, H., and Yu, N. (2020). A novel evaluation metric for deep learning-based side channel analysis and its extended application to imbalanced data. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(3):73–96.

APPENDIX

Preliminaries

In this section, we formally describe the binary classification problem and evaluation metrics. Using this, we formalize the problem of information leakage (IL) in Section 2. Note that, we use these notations throughout the paper.

Binary Classification: Notation and Terminology

In binary classification, the learning algorithm is provided with a set of training data $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=0}^N \subset \mathcal{X} \times \mathcal{Y}$ of size $N \in \mathbb{N}$, where $\mathcal{X} = \mathbb{R}^d$ is the instance (input) space and $\mathcal{Y} = \{0, 1\}$ the (binary) output space. The task of the learner is to induce a hypothesis $h \in \mathcal{H}$ with low generalization error (risk)

$$R(h) = \int_{\mathcal{X} \times \mathcal{Y}} L(y, h(\boldsymbol{x}))\, d\, P(\boldsymbol{x}, y) \qquad (4)$$

where $\mathcal{H}$ is the underlying hypothesis space (set of candidate functions the learner can choose from), $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ a loss function, and P a joint probability measure modeling the underlying data-generating process. A loss function commonly used in binary classification is the 0-1 loss defined as $L_{01}(y, \hat{y}) := [\![y \neq \hat{y}]\!]$, where $[\![c]\!]$ is the indicator function returning a value 1 if condition c is true and 0 otherwise.

The measure P in (4) induces marginal probability (density) functions on $\mathcal{X}$ and $\mathcal{Y}$ as well a conditional probability of the class y given an instance $\boldsymbol{x}$, so that we can write $p(\boldsymbol{x}, y) = p(y \,|\, \boldsymbol{x}) \times p(\boldsymbol{x})$. Of course, these probabilities are not known to the learner, so that (4) cannot be minimized directly. Instead, learning is commonly accomplished by minimizing (a regularized version of) the *empirical risk*

$$R_{emp}(h) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, h(\boldsymbol{x}_i)) \, . \qquad (5)$$

In the following, we denote by $\hat{g} \in \mathcal{H}$ an empirical risk-minimizer, i.e., a minimizer of (5).

Evaluation Metrics

We define evaluation measures used for binary classification as per Koyejo et al. (2015). For a given $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=0}^N$, let $\boldsymbol{y}$ be the ground-truth labels and $\hat{\boldsymbol{y}} = (\hat{y}_1, \dots, \hat{y}_N)$ predictions, such that $\hat{y}_i = \hat{g}(\boldsymbol{x_i}), \forall i \in [N] := \{1, \dots, N\}$.
Accuracy is defined as the proportion of correct predictions:

$$\mathrm{m}_{ACC}(\hat{\boldsymbol{y}}, \boldsymbol{y}) := \frac{1}{N} \textstyle\sum_{i=0}^N [\![\hat{y} = y]\!].$$

Confusion Matrix. Many evaluation metrics is defined using *true positive* (TP), *true negative* (TN), *false positive* (FP) and *false negative* (FN). Formally, they are defined as: $TN(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \sum_{i=0}^N [\![y_i = 0, \hat{y}_i = 0]\!], TP(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \sum_{i=0}^N [\![y_i = 1, \hat{y}_i = 1]\!], FP(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \sum_{i=0}^N [\![y_i = 0, \hat{y}_i = 1]\!], FN(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \sum_{i=0}^N [\![y_i = 1, \hat{y}_i = 0]\!]$. Using these, the Confusion Matrix is defined as:

$$\mathrm{m}_{CM}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \begin{pmatrix} TN(\hat{\boldsymbol{y}}, \boldsymbol{y}) & FP(\hat{\boldsymbol{y}}, \boldsymbol{y}) \\ FN(\hat{\boldsymbol{y}}, \boldsymbol{y}) & TP(\hat{\boldsymbol{y}}, \boldsymbol{y}) \end{pmatrix} .$$

F1-Score. F1-SCORE is an accuracy measure which penalizes the FP more than FN and defined as:

$$\mathrm{m}_{F_1}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{2TP(\hat{\boldsymbol{y}}, \boldsymbol{y})}{(2TP(\hat{\boldsymbol{y}}, \boldsymbol{y}) + FN(\hat{\boldsymbol{y}}, \boldsymbol{y}) + FP(\hat{\boldsymbol{y}}, \boldsymbol{y}))} .$$

False Negative Rate. FNR is defined as the ratio of FN to the total positive instances:

$$\mathrm{m}_{FNR}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{FN(\hat{\boldsymbol{y}}, \boldsymbol{y})}{(FN(\hat{\boldsymbol{y}}, \boldsymbol{y}) + TP(\hat{\boldsymbol{y}}, \boldsymbol{y}))} .$$

False Positive Rate. FPR is defined as the ratio of FP to the total negative instances:

$$\mathrm{m}_{FPR}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{FP(\hat{\boldsymbol{y}}, \boldsymbol{y})}{(FP(\hat{\boldsymbol{y}}, \boldsymbol{y}) + TN(\hat{\boldsymbol{y}}, \boldsymbol{y}))} .$$

Statistical Tests

In this section, we explain the statistical tests in more detail, which are used for our proposed information leakage detection (ILD) approaches in Section 3.

Paired T-Test (PTT)

PTT is used to compare two samples (generated from an underlying population) in which the observations in one sample can be paired with observations in the other sample (Demšar, 2006). We apply K-Fold cross-validation (KFCV) and use the same K train-test datasets pairs for evaluating the majority voting classifier and binary classifiers, to produce K paired accuracy estimates of majority voting classifier ($\boldsymbol{a}_{mc}$) with the binary classifier ($\boldsymbol{a}_j$). For binary classifier C_j, let $H_0(\boldsymbol{a}_j = \boldsymbol{a}_{mc})$ be the null hypothesis and $H_1(\boldsymbol{a}_j \neq \boldsymbol{a}_{mc})$ be the alternate hypothesis. $H_0(\boldsymbol{a}_j = \boldsymbol{a}_{mc})$ indicates that the underlying distribution of two populations is the same, which means that there is no difference between the performance of 2 binary classifiers (Demšar, 2006). $H_1(\boldsymbol{a}_j \neq \boldsymbol{a}_{mc})$ instead implies that there is a significant difference between the performance of 2 binary classifiers.

The p-value quantifies the probability of accepting H_0 and is evaluated by determining the area under the Student's t-distribution curve at value t, which is $1 - \text{cdf}(t)$. The t-statistic is evaluated as $t = \frac{\mu}{\sigma/\sqrt{K}}$, such that $\mu = \frac{1}{K}\sum_{k=1}^{K} d_i, d_i = a_{jk} - a_{mck}$ and $\sigma^2 = \frac{1}{K}\sum_{k=1}^{K} \frac{(\mu - d_k)^2}{K-1}$. Nadeau and Bengio (2003) proposed to adjust the variance for considering the dependency in estimates due to KFCV is defined as $\sigma^2_{Cor} = \sigma(\frac{1}{K} + \frac{1}{K-1})$. This variance σ^2_{Cor} is used to calculate value of t-statistic as $t = \frac{\mu}{\sigma_{Cor}}$. PTT requires a large number of estimates (large K) to produce a precise p-value, which diminishes the effect of the correction term $\frac{1}{(K-1)}$ for σ^2_{Cor} and produces imprecise accuracy estimates as the test set size reduces.

Fisher's Exact Test (FET)

FET is a non-parametric test which is used to calculate the probability of significance independence (non dependence) between two classifications methods by analyzing the contingency table containing the result of classifying objects by two methods (Fisher, 1922). For example, given a sample of people, we can divide them based on gender and based on if they like cricket as a sport. Assuming that the sample is a good representation of people and, for the sake of argument, generally most of the men in our sample like cricket, and most of the women do not. The FET would produce a very low p-value, implying that the two classification methods are correlated.

The p-value is computed using Hypergeometric distribution as:

$$Pr(M)(N,R,r) = \frac{C^{(R)}_{(TN)} \times C^{(N-R)}_{(r-TN)}}{C^{(N)}_{(r)}} = \frac{C^{(FN+TN)}_{(TN)} \times C^{(FP+TP)}_{(FP)}}{C^{(TP+TN+FP+FN)}_{(TN+FP)}}$$

where $r = TN + FP$, $N = FP + TP + FN + TN$, $R = TN + FN$ and $C^{(n)}_{(r)}$ is the combinations of choosing r items from the given n items. The p-value is calculated by summing up the probabilities $Pr(M)$ for all tables having a probability equal to or smaller than that observed M. This test considers all possible tables with the observed marginal counts for TN of the matrix M, to calculate the chance of getting a table at least as "extreme". P-value for majority voting classifier using above equation is $Pr(X = TN) = 1.0$, as if the predicted class is 0 ($\hat{\boldsymbol{y}} = \mathbf{0}$), then $FP = 0, TP = 0$ and if it is 1 ($\hat{\boldsymbol{y}} = \mathbf{1}$), then $FN = 0, TN = 0$.

Relation to Mathews Correlation Coefficient. Mathews Correlation Coefficient (m_{MCC}) is a balanced accuracy evaluation measure that penalizes FP and FN equally and accounts for imbalance in the dataset (Chicco et al., 2021). Camilli (1995) showed that, m_{MCC} is directly proportional to the square root of the χ^2 statistic, i.e., $|\text{m}_{MCC}| = \sqrt{\chi^2/N}$ and the χ^2 statistical test is asymptotically equivalent to FET. For majority voting classifier and random guessing $\text{m}_{MCC} = 0$, thus $\chi^2 = 0$, producing the χ^2 p-value as 1 (Camilli, 1995). This indirect relation to Mathews Correlation Coefficient makes FET an appropriate statistical test for testing the learnability of a binary classifier using the confusion-matrix while taking imbalance in the dataset into account.

Holm-Bonferroni Correction

The Holm–Bonferroni method controls the family-wise error rate (probability of false positive or Type 1 errors) by adjusting the rejection criteria α for each individual hypotheses (Holm, 1979). We consider a family of null hypotheses $\mathcal{F} = H_1, \ldots, H_J$ and obtain p-values $p_1, \ldots, p_J$, from independently testing each classifier $C_j \in \mathcal{C}$, such that $J = |\mathcal{C}|$ For getting an aggregated decision, the significance level for the set $\mathcal{F}$ is not higher than the pre-specified threshold $\alpha = 0.01$. The p-values are sorted in ascending order, i.e. $p_1 \leq p_2, \ldots p_{j-1} \leq p_J$, and for each hypothesis $H_j \in \mathcal{F}$, if $p_j < \frac{\alpha}{J+1-j}$, H_j is rejected. Let H_{m+1} be first hypothesis for which the p-value does not validate rejection, i.e. $p_{m+1} > \frac{\alpha}{J-m}$. Then, the rejected hypotheses are $\{H_1, \ldots, H_m\}$ and the accepted hypothesis are $\{H_{m+1}, \ldots, H_J\}$.

IL Detection. We imply that even if one of the $H_j \in \mathcal{F}$ is rejected, i.e. $m >= 1$, then IL exists in the system. This increases the probability of obtaining false positives increases. So, we have also analyzed the performance of our approaches for different values of m on real-dataset as described in Section 4.3