# Exploiting Support Vector Machine Algorithm to Break the Secret Key

*Shourong HOU, Yujie ZHOU, Hongming LIU, Nianhao ZHU*

Dept. of Electronic Engineering, Shanghai Jiao Tong University, 800 Dongchuan Road, Shanghai, China

ruihou@sjtu.edu.cn, mlscagroup@163.com, hongming.liu@aisinochip.com, nianhao.zhu@aisinochip.com

Submitted November 14, 2017 / Accepted February 14, 2018

**Abstract.** *Template attacks (TA) and support vector machine (SVM) are two effective methods in side channel attacks (SCAs). Almost all studies on SVM in SCAs assume the required power traces are sufficient, which also implies the number of profiling traces belonging to each class is equivalent. Indeed, in the real attack scenario, there may not be enough power traces due to various restrictions. More specifically, the Hamming Weight of the S-Box output results in 9 binomial distributed classes, which significantly reduces the performance of SVM compared with the uniformly distributed classes. In this paper, the impact of the distribution of profiling traces on the performance of SVM is first explored in detail. And also, we conduct Synthetic Minority Oversampling TEchnique (SMOTE) to solve the problem caused by the binomial distributed classes. By using SMOTE, the success rate of SVM is improved in the testing phase, and SVM requires fewer power traces to recover the key. Besides, TA is selected as a comparison. In contrast to what is perceived as common knowledge in unrestricted scenarios, our results indicate that SVM with proper parameters can significantly outperform TA.*

## Keywords

Power analysis, support vector machine, synthetic minority oversampling technique, Hamming Weight class

## 1. Introduction

Kocher et al. [1] first brought forward power analysis (PA) attacks in 1999. Since then, a variety of PA attacks have emerged, such as differential power analysis (DPA) [1], template attacks (TA) [2], correlation power analysis (CPA) [3], stochastic model based power analysis (SMPA) [4] and so on. The cryptographic device must maintain the secret key regardless of whether the algorithm itself is public or not. Thus, a crucial requirement is that the key-related information of a cryptographic algorithm must not be disclosed during execution. So far, none of the cryptographic devices has been able to prevent this relevant information from being leaked through various side channels. The book [5] comprehensively summarizes PA attacks and countermeasures.

As early as the nineties of last century, Rivest [6] had recognized the similarities between machine learning (ML) and cryptography. In recent years, a large number of ML algorithms have been applied to PA attacks, e.g. multilayer perceptron [7], [8], k-means clustering [9], k-nearest neighbors [10], support vector machine (SVM) [11–16], etc. Hospodar et al.[11] first applied SVM to PA attacks. Although no real attack has been performed, it provides a novel perspective on how SVM is used in PA attacks. The first extension to 9 Hamming Weight (HW) classes for SVM was given in [13]. Lerman et al. [15] suggested the attack based on ML against a masked AES implementation. The authors studied ML algorithms mostly using 9 or up to 16 classes. We successfully recovered the secret key by using SVM in [16]. These related contributions suggest that some ML algorithms are effective in PA attacks. Furthermore, the performance of SVM is slightly superior to other ML algorithms. However, almost all studies on SVM in PA attacks [12–16] assume power traces are sufficient to reveal key-related leakage information and the number of profiling traces belonging to each class is equal. Indeed, the number of profiling traces may be different in the real attack scenario.

On the one hand, one can consider the S-Box output value itself as a sensitive variable, resulting in 256 uniformly distributed classes. On the other hand, the attack target can be also the HW of an 8-bit intermediate value, resulting in 9 binomial distributed HW classes. Even more, considering 256 classes yields direct information about the secret key because each class is only relevant to one guessing key. However, each class is associated with multiple guessing keys when using 9 HW classes. For instance, the HW class 4 needs to handle the largest number of guessing keys, where there are 70 possible values. However, the number of possible value is 1 when the HW value is 0 or 8. This is called data imbalance in the ML community.

The authors [17] have confirmed that the separating hyperplane of SVM trained with data imbalanced will skew towards the minority class, and this skewness reduces the performance of SVM. It seems to be more advisable to use the S-Box output directly as the label value of an SVM classifier.

However, with the increase in the number of classes, the computational complexity of SVM also rises. This complexity of multi-class SVM rises with $O(|\Theta|^2)$ when the one-against-one strategy [18] is used, where $|\Theta|$ is the number of classes we need to classify. This will make the SVM algorithm inefficient in the parameter tuning phase because of the high computational burden. In light of this, the Hamming Weight model, which assumes the intermediate power consumption value of the S-Box output, is selected as the hypothetical power leakage model in our paper. From our point of view, the importance of the distribution of power traces in PA attacks has not yet been investigated. Therefore, the purpose of this article is to explore the impact of the distribution of profiling traces belonging to each class on the performance of SVM.

In this paper, we first calculated the label of each power trace and then predicted the probability that all instances belong to each class. Finally, the correct key was obtained by the maximum likelihood estimation. All experiments were performed on the publicly available power traces. We used Synthetic Minority Oversampling TEchnique (SMOTE) [19] instead of Different Error Costs (DEC) [20] to compensate for the distribution of HW classes. By using SMOTE, we modified 9 binomial distributed HW classes to 7 uniformly distributed HW classes, which could get more appropriate SVM parameters in the parameter tuning phase. Our results demonstrated that the success rate of 7 uniformly distributed HW classes was higher than that of 9 binomial ones for SVM in the testing phase. Moreover, SVM-RBF only required about 4 power traces to recover the secret key when classifying 7 classes. The remainder of this article is organized as follows: Section 2 introduces the basic knowledge of profiling attacks and SVM. Section 3 gives our methodologies used in this article. Our experiments and results are presented in Sec. 4. We conclude this article in Sec. 5.

# 2. Background

In this section, we briefly introduce the basic information of previous profiling attacks and the SVM algorithm used in this paper.

## 2.1 Profiling Attacks

Profiling and non-profiling attacks are two main types of PA attacks. Profiling attacks assume that an attacker has an identical cryptographic device that is almost completely controlled by him. For this device, he is free to set the key and plaintext and then calculate the intermediate value. Thus, an attacker can guess the secret key according to an appropriate power leakage model. Profiling attacks contain two phases: a profiling (learning) phase and key recovery (attacking) phase. In the profiling phase, the key-related leakage information caused by intermediate values being processed can be characterized by profiling traces. The attacker uses these profiles (features, templates) to predict the correct secret key in the attacking phase.

TA is a typical profiling attack based on multivariate Gaussian distribution $N(\mathbf{t}; (\mathbf{m}, \mathbf{C}))$, as described below.

$$(2\pi)^{-\frac{N}{2}} |\mathbf{C}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{t}-\mathbf{m})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{t}-\mathbf{m})\right) \tag{1}$$

where $\mathbf{t}$ represents a $N$-dimensional vector, $\mathbf{m}$ is the mean vector, $\mathbf{C}$ is the covariance matrix, which is called templates. For TA, the attacker builds different templates for different classes, which corresponds to different intermediate values in the learning phase. In the attacking phase, the attacker uses the maximum likelihood estimation as a distinguisher. The log likelihood of each possible key $k$ is as follows [2]:

$$\log L_k = \log \prod_{i=1}^{M_k} P(\mathbf{t}_i|(\mathbf{m}, \mathbf{C}) = \sum_{i=1}^{M_k} \log P(\mathbf{t}_i|(\mathbf{m}, \mathbf{C}) \tag{2}$$

where $M_k$ is the number of power traces belonging to the secret key $k$.

## 2.2 Understanding the SVM

Cortes and Vapnik [21] proposed the SVM algorithm to address the linear binary classification with high generalization. Let $D_M = \{(\mathbf{X}_i, y_i) \big| \mathbf{X}_i \in R^N, y_i \in \{-1, +1\}, i = 1, 2, \ldots, M\}$ represent a training set, where $\mathbf{X}_i$ is a training vector, and $y_i$ is the label of $\mathbf{X}_i$. The training vector $\mathbf{X}$ is mapped into feature space by the nonlinear function $\phi(\cdot)$. Consequently, the maximum margin of a binary-class SVM classifier is a constrained optimization problem as follows:

$$\begin{gathered} \min_{\omega, b, \xi} \left(\frac{1}{2}\|\omega\|^2 + C \sum_{i=1}^{M} \xi_i\right), \\ \text{s.t.} y_i(\omega^{\mathrm{T}}\phi(\mathbf{X}_i) + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, 2, 3, \ldots, M \end{gathered} \tag{3}$$

where $\omega \in R^N$, $b \in R$, and $C > 0$ is the penalty parameter which evaluates the trade-off between training error and margin size, and $\xi_i$ is the training error of $\mathbf{X}_i$. After the Lagrange multiplier is introduced, the optimization problem in (3) is simplified as follows:

$$\begin{gathered} \min_{\alpha} \frac{1}{2} \sum_{i=1}^{M} \sum_{j=1}^{M} \alpha_i \alpha_j y_i y_j K(\mathbf{X}_i, \mathbf{X}_j) - \sum_{i=1}^{M} \alpha_i, \\ \text{s.t.} \sum_{i=1}^{M} \alpha_i y_i = 0, 0 \leq \alpha_i \leq C, i = 1, 2, \ldots, M \end{gathered} \tag{4}$$

where $\alpha_i$ are Lagrange multipliers, and the kernel function is $K(\mathbf{X}_i, \mathbf{X}_j) = \phi(\mathbf{X}_i)^{\mathrm{T}}\phi(\mathbf{X}_j)$.

The kernel function maintains the reasonable computational complexity of SVM in feature space. The common kernel functions are linear kernel ($K^{\mathrm{Linear}}$) and RBF kernel ($K^{\mathrm{RBF}}$).

$$K^{\mathrm{Linear}}(\mathbf{X}_i, \mathbf{X}_j) = \mathbf{X}_i{}^{\mathrm{T}}\mathbf{X}_j, \tag{5}$$

$$K^{\mathrm{RBF}}(\mathbf{X}_i, \mathbf{X}_j) = \exp\left(-\gamma \left\|\mathbf{X}_i - \mathbf{X}_j\right\|^2\right) \tag{6}$$

where $\gamma$ is the hyperparameter in (6), and the notation $\|\cdot, \cdot\|$ represents the $L^2$ norm (Euclidean length) between two vectors [22].

For consideration of training time and accuracy (ACC, the ratio of true positives and true negatives to the total number of all instances), the one-against-one strategy [18] can

be used to train an SVM classifier for each pair of possible classes. In order to use the maximum likelihood estimation to recover the secret key, an attacker is more interested in the probability of an instance $\mathbf{X}_i$ belonging to the class $c$. Accordingly, we give the posterior conditional probability $P_{\mathrm{SVM}}(\mathbf{X}_i|c)$ of each instance [23].

# 3. Methodologies

In order to ensure the reproducibility of our results, we used a publicly available dataset. The DPA Contest v4 (DPACv4) [24] provides 100,000 power traces of the masked AES software implementation. Since the mask value is known in [16], we can directly convert this dataset to an unprotected scenario. We selected 4000 (DS0) and 8000 (DS1) random power traces to make a fair comparison of all experiments. And also, we only explored how to recover the secret key more efficiently and ignored the mask recovery phase.

Our experimental methodology was as follows: Given a dataset, a random two-thirds was used as the learning set and the remaining one-third was reserved as the testing set. The learning set was divided into training and validation sets by using 10-fold cross validation. The validation sets of all folds were used in the parameter tunning phase. The best parameters (the one with the highest average accuracy on all validation folds) were used for training the final SVM model in the testing phase. Furthermore, the correct key was obtained by the maximum likelihood estimation in the key recovery phase.

Figure 1 illustrates the framework of our experimental procedures and concepts involved in this article. SMOTE is used to compensate for the binomial distributed HW classes after the execution of feature selection. The parameter tuning phase finds the best parameters for the training and testing phases. Each experiment was repeated ten times in the loop block. The testing results are given in a form of ACC/AveP/F-measure. The guessing entropy is used to evaluate the number of remaining keys.

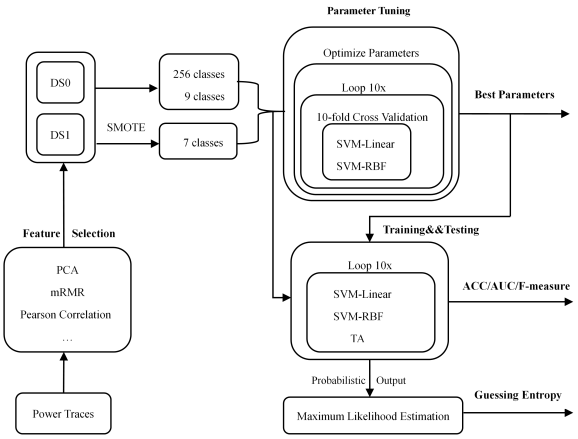


**Fig. 1.** Block diagram of the framework of our experimental procedures and concepts involved in this article.

## 3.1 Feature Selection

Our dataset is focused on all bytes (0 to 15) of the first round key of AES. Although it is a software implementation, the most leaking operation is not register reading or writing, but the S-Box operation of the first round of AES. As shown in Fig. 2, the HW model is used to characterize the hypothetical power consumption of the S-Box output. The HW value of the S-Box output, i.e., $\mathrm{HW}(\mathrm{Sbox}[t_i \oplus k_i]), i = 0, 1, \ldots, 15$, is selected as the label of an SVM classifier. Here $t_i$ represents the $i$th byte of a random plaintext, $k_i$ denotes the $i$th byte of the fixed secret key, and $\mathrm{Sbox}[\cdot]$ is a substitution operation. Consequently, the label value of an SVM classifier corresponds to the HW value from 0 to 8. In this case, the number of power traces belonging to each HW class obeys the binomial distribution.

According to article [16], the interesting points were extracted from 16 S-Boxes. We calculated Pearson correlation coefficients between each sample instant of power traces and the HW of the S-Box output to locate interesting points. Moreover, the 32 highest correlated sample instants were selected as interesting points. As we can see from Fig. 3, for the eighth S-Box, most of the sample instants have no prominent power leakage. We omit the details about the remaining S-Boxes due to the lack of space.

Here we only used the Pearson correlation method for feature selection. In addition, many signal preprocessing techniques can also be used to choose interesting points in PA attacks, e.g. minimum redundancy maximum relevance (mRMR) [25] and principal component analysis (PCA) [26], etc.

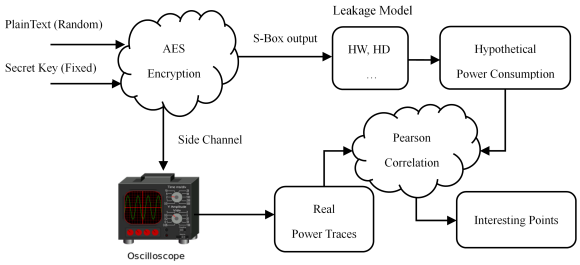


**Fig. 2.** Block diagram of our feature selection.

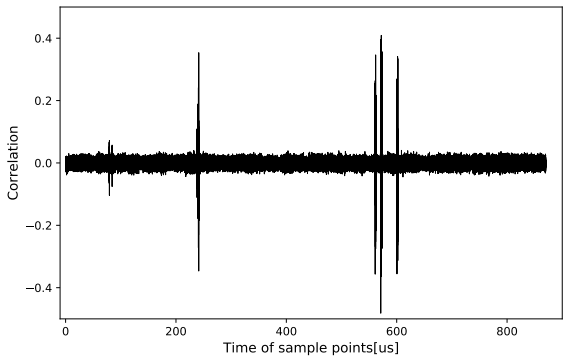


**Fig. 3.** Correlation between each sample instants and the HW of the 8th S-Box output in the first round of AES.

### 3.2 SMOTE

The strategy of SVM to solve the problem of data imbalance is divided into algorithm level and data level methods [27], [28]. Algorithm level methods focus on modifying existing algorithms to mitigate their bias towards the majority class. The Different Error Costs (DEC) method is a typical representative of this category proposed in [17] to overcome the same cost $C$ for both minority and majority misclassification. As given in (7) below:

$$\min\left(\frac{1}{2}\|\omega\|^2 + C^+ \sum_{i|y_i=+1}^{M} \xi_i + C^- \sum_{i|y_i=-1}^{M} \xi_i\right), \quad \text{s.t.} y_i\left(\omega^{\mathrm{T}}\phi(\mathbf{X}_i) + b\right) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, 2, \ldots, M \tag{7}$$

where $C^+$ is the misclassification cost for the minority class, while $C^-$ is the misclassification cost for the majority class. The DEC method improves SVM by allocating the minority class instances with a higher misclassification cost (i.e., $C^+ > C^-$). The improved SVM algorithm would not tend to skew the separating hyperplane towards the minority class instances, which reduces the total misclassifications. Here we simply set the $C^+/C^-$ equal to the ratio of the minority examples to the majority examples [20].

At the data level, the implemented state-of-the-art methods can be categorized into over-sampling, under-sampling, the combination of under and over-sampling, and ensemble learning methods [29]. Compared with other sampling techniques, SMOTE is the most powerful technique that has been a great success in many applications [19]. SMOTE creates synthetic data based on similarities among existing minority examples in feature space. The minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments mixing any of the $k$ minority class nearest neighbors. Synthetic examples are created in the following way: Calculate the difference between the selected feature vector and its nearest neighbors. Multiply this difference by a random number between 0 and 1, and add it to the selected feature vector. This causes the selection of a random point on the line between two particular features.

The HW of an 8-bit S-Box output has resulted in 9 binomial distributed classes. Naturally, this distribution does not provide an equal number of power traces for each HW class. Moreover, in our datasets, the number of power traces belonging to the HW class 0 and 8 accounts for about 0.8% ($2\times\frac{1}{256}$). The synthetic power traces cannot represent the true distribution of the HW class 0 or 8. Hence, the HW class 0 and 8 in our datasets are discarded directly, and then we set different nearest neighbors for the remaining HW classes (1 to 7). We can get 7 uniformly distributed HW classes by using SMOTE. As a comparison, we will report the experimental results of classifying 7, 9, and 256 classes in the next section.

## 4. Experiments and Results

LIBSVM (Library for Support Vector Machine) [30] was used as the framework for conducting our attacks. All experiments were performed on Asus laptop with 2.50 GHz Intel Core (TM) i5-7200U, 16 GB 2133 MHz DDR4 (Windows10 x64). The attack lasted about 12 weeks without considering the time to create two datasets.

### 4.1 Parameter Tuning Phase

There is no an effective learning method to cover all attack scenarios in the parameter tuning phase. According to paper [31], we selected the penalty parameter $C$ from 0.01 to 256 with a step of 2, epsilon (tolerance of termination criterion) from 0.01 to 0.25 with a step of 0.05, the hyper-parameter $\gamma$ in (6) from 0.001 to 32 with a step of 2. Here we gave the parameter range but omitted tuning details. An open-source python toolbox, namely imbalanced-learn [32], was used to generate synthetic power traces.

| | **256 classes** | **9 classes** | **7 classes** |
|---|---|---|---|
| **S-Box0** | $20.6_{128.0/0.0156}$ | $77.6_{8.0/0.25}$ | $79.3_{2.0/0.5}$ |
| **S-Box1** | $49.3_{4.0/0.25}$ | $92.3_{16.0/0.125}$ | $93.8_{4.0/0.25}$ |
| **S-Box2** | $36.0_{64.0/0.03125}$ | $93.9_{4.0/0.25}$ | $96.2_{16.0/0.25}$ |
| **S-Box3** | $39.8_{32.0/0.03125}$ | $93.3_{4.0/0.25}$ | $94.0_{4.0/0.125}$ |
| **S-Box4** | $33.6_{64.0/0.03125}$ | $77.5_{32.0/0.5}$ | $80.7_{2.0/0.25}$ |
| **S-Box5** | $33.9_{32.0/0.0625}$ | $91.9_{4.0/0.25}$ | $93.0_{2.0/0.25}$ |
| **S-Box6** | $46.7_{8.0/0.125}$ | $90.7_{8.0/0.25}$ | $92.2_{2.0/0.25}$ |
| **S-Box7** | $32.0_{128.0/0.0156}$ | $94.3_{4.0/0.25}$ | $95.7_{16.0/0.125}$ |
| **S-Box8** | $24.4_{16.0/0.125}$ | $69.5_{32.0/0.5}$ | $74.8_{8.0/0.5}$ |
| **S-Box9** | $45.2_{8.0/0.125}$ | $92.0_{8.0/0.25}$ | $93.0_{2.0/0.5}$ |
| **S-Box10** | $44.1_{8.0/0.125}$ | $95.7_{16.0/0.25}$ | $96.7_{32.0/0.125}$ |
| **S-Box11** | $40.9_{16.0/0.0625}$ | $94.4_{4.0/0.125}$ | $95.6_{2.0/0.5}$ |
| **S-Box12** | $26.3_{32.0/0.03125}$ | $73.0_{4.0/0.125}$ | $75.3_{8.0/0.125}$ |
| **S-Box13** | $42.9_{32.0/0.03125}$ | $92.1_{4.0/0.25}$ | $94.6_{16.0/0.25}$ |
| **S-Box14** | $42.7_{32.0/0.03125}$ | $90.4_{2.0/0.25}$ | $92.1_{4.0/0.125}$ |
| **S-Box15** | $45.3_{64.0/0.03125}$ | $95.2_{8.0/0.25}$ | $97.2_{2.0/0.25}$ |

**Tab. 1.** Success rates of SVM-RBF in the parameter tuning phase by using DS0.

| | **256 classes** | **9 classes** | **7 classes** |
|---|---|---|---|
| **S-Box0** | $29.7_{32.0/0.125}$ | $79.1_{4.0/0.25}$ | $80.8_{16.0/0.125}$ |
| **S-Box1** | $61.3_{8.0/0.5}$ | $93.8_{4.0/0.25}$ | $94.7_{2.0/0.5}$ |
| **S-Box2** | $45.1_{8.0/0.25}$ | $95.4_{4.0/0.5}$ | $96.6_{4.0/0.25}$ |
| **S-Box3** | $50.4_{8.0/0.125}$ | $93.9_{8.0/0.25}$ | $94.7_{2.0/0.5}$ |
| **S-Box4** | $46.7_{8.0/0.03125}$ | $78.9_{2.0/0.5}$ | $82.3_{4.0/0.25}$ |
| **S-Box5** | $45.3_{16.0/0.125}$ | $92.9_{8.0/0.5}$ | $95.4_{4.0/0.25}$ |
| **S-Box6** | $57.5_{4.0/0.25}$ | $91.7_{8.0/0.25}$ | $93.2_{16.0/0.5}$ |
| **S-Box7** | $45.3_{32.0/0.0625}$ | $95.1_{4.0/0.125}$ | $97.0_{4.0/0.125}$ |
| **S-Box8** | $33.1_{64.0/0.0625}$ | $71.3_{2.0/0.5}$ | $77.4_{4.0/0.125}$ |
| **S-Box9** | $57.5_{8.0/0.125}$ | $92.9_{2.0/0.5}$ | $93.9_{4.0/0.25}$ |
| **S-Box10** | $53.1_{16.0/0.125}$ | $96.8_{8.0/0.125}$ | $98.1_{2.0/0.25}$ |
| **S-Box11** | $52.4_{8.0/0.125}$ | $95.4_{4.0/0.25}$ | $97.3_{4.0/0.25}$ |
| **S-Box12** | $39.6_{32.0/0.0625}$ | $74.1_{2.0/0.5}$ | $76.5_{128.0/0.125}$ |
| **S-Box13** | $53.1_{32.0/0.0625}$ | $93.8_{2.0/0.25}$ | $97.5_{2.0/0.25}$ |
| **S-Box14** | $54.6_{16.0/0.0625}$ | $91.0_{4.0/0.25}$ | $94.4_{2.0/0.25}$ |
| **S-Box15** | $56.1_{8.0/0.125}$ | $96.1_{4.0/0.25}$ | $97.4_{2.0/0.25}$ |

**Tab. 2.** Success rates of SVM-RBF in the parameter tuning phase by using DS1.

| | DS0 | | | DS1 | | |
|---|---|---|---|---|---|---|
| | **SVM-Linear** | **SVM-RBF** | **TA** | **SVM-Linear** | **SVM-RBF** | **TA** |
| **S-Box0** | 13.60/0.09/0.13 | 16.07/0.12/0.15 | 13.50 | 20.97/0.17/0.18 | 24.78/0.21/0.22 | 16.01 |
| **S-Box1** | 41.62/0.32/0.40 | 43.49/0.39/0.41 | 36.75 | 51.77/0.48/0.51 | 56.46/0.55/0.56 | 41.33 |
| **S-Box2** | 26.55/0.20/0.24 | 29.89/0.23/0.27 | 26.90 | 34.10/0.30/0.33 | 36.83/0.35/0.36 | 30.67 |
| **S-Box3** | 28.02/0.23/0.27 | 30.07/0.24/0.27 | 26.45 | 43.96/0.43/0.42 | 45.45/0.43/0.44 | 31.26 |
| **S-Box4** | 28.30/0.24/0.27 | 29.07/0.24/0.27 | 26.55 | 34.13/0.27/0.33 | 40.09/0.37/0.39 | 29.70 |
| **S-Box5** | 28.37/0.23/0.25 | 30.36/0.25/0.27 | 26.70 | 37.32/0.34/0.36 | 38.46/0.36/0.38 | 31.60 |
| **S-Box6** | 33.88/0.27/0.32 | 41.96/0.39/0.40 | 36.00 | 46.41/0.44/0.46 | 49.99/0.51/0.49 | 39.78 |
| **S-Box7** | 24.33/0.18/0.22 | 26.55/0.19/0.23 | 22.75 | 37.18/0.31/0.36 | 39.12/0.35/0.38 | 26.65 |
| **S-Box8** | 15.82/0.11/0.14 | 17.58/0.12/0.16 | 14.65 | 23.59/0.16/0.22 | 25.34/0.20/0.24 | 20.47 |
| **S-Box9** | 37.28/0.31/0.35 | 38.15/0.33/0.36 | 34.50 | 50.37/0.48/0.50 | 52.55/0.57/0.52 | 42.14 |
| **S-Box10** | 34.47/0.28/0.32 | 35.76/0.30/0.33 | 33.45 | 43.48/0.41/0.43 | 47.96/0.45/0.47 | 38.13 |
| **S-Box11** | 30.66/0.24/0.29 | 34.29/0.29/0.32 | 27.05 | 44.89/0.41/0.44 | 46.47/0.44/0.46 | 32.14 |
| **S-Box12** | 19.05/0.13/0.17 | 20.63/0.14/0.18 | 20.20 | 31.32/0.26/0.30 | 33.09/0.32/0.30 | 22.56 |
| **S-Box13** | 33.41/0.27/0.31 | 35.29/0.29/0.32 | 34.40 | 46.44/0.43/0.45 | 47.44/0.47/0.46 | 37.23 |
| **S-Box14** | 33.88/0.27/0.31 | 34.41/0.28/0.32 | 31.05 | 45.68/0.43/0.45 | 47.85/0.46/0.47 | 34.20 |
| **S-Box15** | 36.75/0.27/0.34 | 37.75/0.31/0.35 | 33.40 | 47.41/0.45/0.47 | 50.69/0.48/0.50 | 36.25 |

**Tab. 3.** Testing results (ACC/AveP/F-measure) of SVM and TA for 256 classes using power traces of DS0 and DS1.

In Tabs. 1 and 2, the success rates of SVM-RBF for all S-Boxes are given in $\mathrm{ACC}_{C/\gamma}$ form. All values of ACC are given in percentages, and we provide the parameter combinations penalty parameter $C$ and hyperparameter $\gamma$) reaching those values. The success rate of 7 uniformly distributed classes is significantly higher than that of 9 binomial ones for SVM-RBF. This proves that the distribution of profiling traces affects the performance of SVM-RBF. When the dataset size is expanded from DS0 to DS1, the success rate of 9 binomial distributed classes is improved by less than 1.5%. Excitingly, the success rate of 7 HW classes using DS0 is basically equivalent to that of 9 binomial ones using DS1. In other words, our method improves the performance of SVM-RBF without increasing the number of profiling traces, which is an interesting aspect of PA attacks.

For SVM-RBF, the success rate of 256 classes is significantly lower than that of 7 and 9 classes. This can be explained by the fact that the number of profiling traces is not enough to train good parameters to classify 256 classes. However, when considering random classification, there is $1/9$ chance of a successful guess for 9 classes, while there is $1/256$ chance for a random hit in the 256 classes scenario. Obviously, the success rate of 256 classes is higher than a random guess. The results may even be further improved through a more exhaustive parameter tuning phase, which requires more profiling traces and longer tuning time. Nevertheless, the complexity of parameter tuning makes it difficult to give some theoretical explanations for the performance of SVM. Furthermore, the high complexity of the attack method makes the investigated algorithm unattractive for some security evaluation scenarios.

We also used the SVM-RBF with DEC to solve the problem caused by data imbalance. Figure 4 gives the success rate of SVM-RBF with DEC when using DS1 to classify 9 binomial distributed HW classes. Compared to using the same cost, the performance of SVM-RBF with DEC has

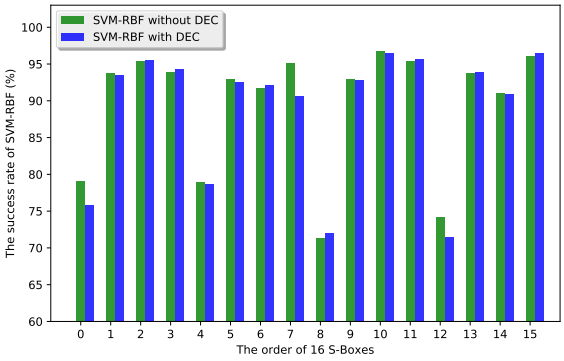


**Fig. 4.** Comparison of the success rate of SVM-RBF when using DS1 to classify 9 HW classes.

hardly been improved, and even worse for some S-Boxes.The reason may be that the strategy we described in Sec. 3.2 for setting penalty parameters is inappropriate. However, the penalty parameter requires to be calculated iteratively, which is difficult to set in the real problem. Hence, in all subsequent experiments, we did not report the results of SVM using the DEC method.

## 4.2 Testing Results

In this section, we only reported the results of SVM-Linear, SVM-RBF with the best parameter combinations, and TA when using DS0 and DS1. Our experiments were executed on the independent testing set to verify the performances of SVM and TA for classifying 7, 9, and 256 classes. Note that for SVM and TA, we used the same datasets and the same interesting points. In order to make our experimental results accurate, each experiment was repeated ten times and then their average score was regarded as the final result.

The testing results are given in a form of ACC/AveP/F-measure for SVM-Linear and SVM-RBF while for TA we only give the success rate. Here, F-measure (F1-score) is

| | DS0 | | | DS1 | | |
|---|---|---|---|---|---|---|
| | **SVM-Linear** | **SVM-RBF** | **TA** | **SVM-Linear** | **SVM-RBF** | **TA** |
| **S-Box0** | 70.22/0.71/0.60 | 73.09/0.78/0.70 | 63.80 | 71.79/0.74/0.61 | 75.19/0.81/0.71 | 65.02 |
| **S-Box1** | 88.37/0.90/0.85 | 90.40/0.92/0.90 | 78.55 | 86.12/0.89/0.83 | 91.72/0.93/0.88 | 80.09 |
| **S-Box2** | 88.99/0.92/0.88 | 91.57/0.94/0.89 | 83.45 | 91.29/0.93/0.90 | 93.19/0.95/0.93 | 84.85 |
| **S-Box3** | 90.89/0.93/0.90 | 91.97/0.93/0.90 | 85.10 | 89.48/0.93/0.89 | 92.23/0.94/0.91 | 85.60 |
| **S-Box4** | 72.99/0.76/0.55 | 74.74/0.81/0.65 | 56.90 | 73.93/0.77/0.72 | 76.15/0.82/0.69 | 58.79 |
| **S-Box5** | 83.19/0.83/0.82 | 85.32/0.86/0.84 | 77.20 | 86.24/0.88/0.86 | 88.27/0.90/0.88 | 78.69 |
| **S-Box6** | 85.95/0.90/0.84 | 87.32/0.89/0.88 | 81.75 | 86.90/0.91/0.86 | 90.28/0.94/0.90 | 81.29 |
| **S-Box7** | 91.97/0.93/0.90 | 93.99/0.95/0.89 | 84.90 | 92.15/0.94/0.91 | 94.64/0.95/0.94 | 85.86 |
| **S-Box8** | 66.84/0.68/0.52 | 68.09/0.71/0.58 | 59.05 | 68.69/0.71/0.61 | 69.28/0.73/0.64 | 60.33 |
| **S-Box9** | 87.17/0.93/0.83 | 88.22/0.92/0.85 | 80.45 | 88.33/0.94/0.81 | 89.12/0.94/0.89 | 80.94 |
| **S-Box10** | 92.87/0.92/0.88 | 94.12/0.94/0.92 | 81.05 | 94.45/0.93/0.94 | 95.11/0.95/0.94 | 82.14 |
| **S-Box11** | 90.09/0.93/0.90 | 92.82/0.93/0.92 | 84.35 | 91.08/0.92/0.88 | 92.77/0.95/0.92 | 86.69 |
| **S-Box12** | 67.97/0.68/0.51 | 70.52/0.71/0.51 | 58.45 | 69.69/0.72/0.67 | 71.90/0.75/0.66 | 58.27 |
| **S-Box13** | 90.15/0.92/0.88 | 91.77/0.94/0.90 | 80.60 | 91.07/0.92/0.89 | 92.46/0.93/0.91 | 81.31 |
| **S-Box14** | 84.77/0.91/0.77 | 85.22/0.90/0.79 | 78.40 | 86.42/0.92/0.87 | 88.60/0.94/0.88 | 80.65 |
| **S-Box15** | 91.60/0.92/0.88 | 94.27/0.94/0.94 | 84.55 | 93.39/0.93/0.93 | 95.51/0.95/0.95 | 83.23 |

**Tab. 4.** Testing results (ACC/AveP/F-measure) of SVM and TA for 9 classes using power traces of DS0 and DS1.

| | DS0 | | | DS1 | | |
|---|---|---|---|---|---|---|
| | **SVM-Linear** | **SVM-RBF** | **TA** | **SVM-Linear** | **SVM-RBF** | **TA** |
| **S-Box0** | 73.63/0.79/0.73 | 76.53/0.83/0.75 | 67.57 | 77.96/0.86/0.76 | 79.69/0.86/0.79 | 69.20 |
| **S-Box1** | 90.75/0.96/0.90 | 92.35/0.98/0.92 | 82.43 | 91.05/0.96/0.91 | 93.34/0.98/0.93 | 83.68 |
| **S-Box2** | 92.39/0.97/0.92 | 95.07/0.98/0.95 | 87.14 | 94.75/0.98/0.94 | 97.32/0.99/0.97 | 88.54 |
| **S-Box3** | 91.75/0.96/0.91 | 93.03/0.97/0.93 | 85.92 | 91.80/0.97/0.91 | 93.89/0.98/0.93 | 86.73 |
| **S-Box4** | 74.38/0.81/0.74 | 76.46/0.83/0.75 | 62.93 | 76.90/0.83/0.76 | 80.33/0.87/0.80 | 64.51 |
| **S-Box5** | 86.57/0.88/0.86 | 89.21/0.91/0.88 | 78.92 | 90.25/0.93/0.90 | 93.99/0.98/0.93 | 84.63 |
| **S-Box6** | 88.32/0.94/0.88 | 90.71/0.95/0.90 | 83.43 | 89.60/0.94/0.89 | 92.18/0.97/0.92 | 83.82 |
| **S-Box7** | 92.82/0.97/0.92 | 94.75/0.98/0.95 | 87.00 | 95.07/0.98/0.95 | 95.69/0.99/0.96 | 88.02 |
| **S-Box8** | 70.56/0.77/0.70 | 72.56/0.77/0.72 | 62.43 | 72.99/0.79/0.72 | 75.04/0.82/0.75 | 64.36 |
| **S-Box9** | 88.57/0.94/0.88 | 90.25/0.97/0.89 | 82.93 | 89.16/0.95/0.90 | 91.98/0.98/0.92 | 83.66 |
| **S-Box10** | 94.89/0.98/0.93 | 96.03/0.99/0.96 | 83.79 | 95.87/0.99/0.95 | 96.69/0.99/0.97 | 84.31 |
| **S-Box11** | 93.46/0.98/0.93 | 95.18/0.99/0.95 | 87.71 | 94.27/0.98/0.94 | 96.03/0.99/0.96 | 88.90 |
| **S-Box12** | 70.35/0.76/0.70 | 72.13/0.78/0.72 | 61.21 | 71.92/0.77/0.71 | 74.38/0.81/0.74 | 63.86 |
| **S-Box13** | 91.43/0.96/0.91 | 93.57/0.97/0.93 | 83.57 | 93.14/0.98/0.93 | 95.07/0.98/0.95 | 84.84 |
| **S-Box14** | 89.89/0.96/0.90 | 90.89/0.96/0.90 | 82.97 | 91.07/0.96/0.90 | 92.12/0.97/0.92 | 84.57 |
| **S-Box15** | 93.64/0.98/0.93 | 95.99/0.99/0.95 | 87.72 | 95.98/0.99/0.95 | 96.32/0.99/0.96 | 88.38 |

**Tab. 5.** Testing results (ACC/AveP/F-measure) of SVM and TA for 7 classes using power traces of DS0 and DS1.

the harmonic mean of the precision and recall, where precision is the ratio of true positives to predicted positives, while recall is the ratio of true positives to actual positives [33]. The receiver operating characteristic curve is usually used to present the results of binary classification problems with the uniformly distributed classes. However, when dealing with highly skewed datasets, the precision-recall curve provides more information about the performance of a learning algorithm [34]. The average precision (AveP) is defined as the area under the precision-recall curve. In Tabs. 3, 4, and 5, all values of ACC are given in percentages, while AveP and F-measure are in the range [0, 1]. The higher the value, the better the result.

As expected, the success rate of SVM-RBF in the parameter tuning phase is higher than the results in the testing phase because of the generalization of SVM. For 256 classes, success rates of SVM-RBF are generally reduced by 3% to 10%, in the worst case, the success rate drops from 39.8% to 30.07% (see S-Box3, DS0, Tab. 3). We can see that for 16 S-Boxes, success rates of SVM are obviously different because of the difference of their power consumption leakage information. When the dataset size is expanded from DS0 to DS1, the success rate of SVM increases by more than 10% for most S-Boxes. That is, the larger dataset size, the higher the success rate. The reason is that the performance of SVM is determined by its parameters, and the dataset size is critical to parameter optimization. Moreover, the success rate of SVM-RBF is 1% $\sim$ 6% higher than that of SVM-Linear when using DS1 in Tab. 3.

From Tabs. 4 and 5, we can see that the testing results of 7 uniformly distributed classes are much better than those of 9 binomial ones. In particular, success rates of 7 uniformly distributed classes are at least 3% higher than those of 9 binomial ones for S-Box0, 2, 5, 8, 11, and 14. For SVM and TA,

the success rate of 7 uniformly distributed classes using DS0 is higher than that of 9 binomial ones using DS1, which is a surprising result. Generally, with the increase of the number of profiling traces, the performance of SVM is improved. In this case, SMOTE is used to compensate for the distribution of existing learning set. However, the success rate of SVM is higher than that of using more profiling traces in the training phase owing to the use of synthetic power traces for the minority classes. This indicates the performance of SVM is dependent on the distribution of the number of profiling traces belonging to each class. Additionally, kernel functions play an important role in improving the performance of SVM. SVM-RBF has a higher success rate than SVM-Linear when classifying 7 and 9 classes. Even more, the success rate of SVM-RBF with using DS0 is higher than that of SVM-Linear with using DS1. The hyperparameter $\gamma$ in (6) provides greater flexibility for SVM-RBF. Inevitably, SVM-RBF also takes more time to find the optimal separating hyperplane in the parameter tuning phase.

Although the success rate gives the impression that SMOTE improves the performance of SVM, AveP and F-measure can analyze the testing results from a novel point of view. The AveP of 9 HW classes is scattered between 0.68 and 0.95, while that of 7 classes is between 0.76 and 0.99. By looking at the confusion matrix (matrix where each row represents the instances in an actual class while each column represents the instances in one predicted class), we find that an SVM classifier handles all instances as the class with more profiling traces when distinguishing between the HW class 0 (or 8) and another class. Naturally, classifying all instances into a single class will not be a successful attack, because this doesn't reveal any information about the secret key. The AveP values of 256 uniformly distributed classes are between 0.09 and 0.57 in Tab. 3, which are significantly lower than those of 7 and 9 classes in Tabs. 4 and 5. This is because the number of profiling traces is not sufficient to train high-precision classifiers when classifying 256 classes. In general, the higher the ACC value, the higher the F-measure value. Besides, F-measure is slightly lower than ACC. Thus, we do not discuss F-measure in detail due to the lack of space.

We also used the standard TA approach to compare the success rates available in the same attack scenario. TA is considered to be the most powerful attack technique from an information theoretic point of view, which assumes that sample points of each trace follow the multivariate Gaussian distribution. As shown in Tabs. 3, 4, and 5, the success rate of TA is obviously lower than that of SVM when using DS0 and DS1 to classify 7, 9, and 256 classes. Compared with SVM, the numerical instability of TA is highlighted when the profiling traces are not enough to reveal the key-related leakage information. In addition, with the increase of the number of profiling traces, the success rate of TA has not increased significantly. However, the success rate of 7 uniformly distributed classes for TA is higher than that of 9 binomial ones. Briefly, our testing results demonstrate that the using SMOTE to compensate for the distribution of HW classes is also effective for TA. Furthermore, SVM (especially SVM-RBF) can significantly outperform the classical TA when properly used.

In order to make our experiments more convincing, our attacks were executed hundreds of times, and then we gave the statistical results of the success rate. Figures 5 and 6 present some box plots that summarize the success rates of SVM and TA when using DS1 to classify 7 and 9 classes. In each box plot, the central bar corresponds to the median (the second quartile), and the green triangle represents the mean. The bottom and top of the box are always the first and third quartiles, and the whisker is the maximum/minimum value excluding outliers. The value is considered as an outlier when it is greater than $\frac{3}{2}$ times of upper/lower quartile. Consistent with our hypothesis, SMOTE can effectively solve the problems caused by data imbalance, which improves the performance of SVM and TA. Besides, the success rate of SVM (especially SVM-RBF) is higher and more concentrated than TA.

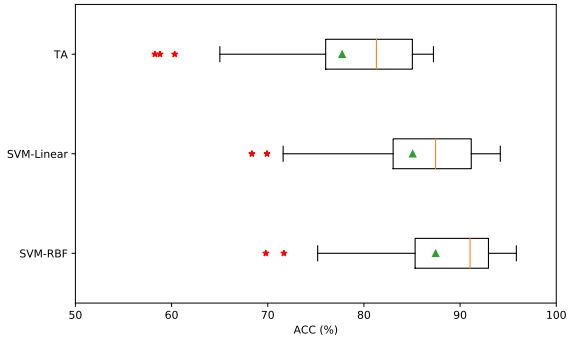


**Fig. 5.** Box plot of SVM and TA for 9 classes by using DS1.

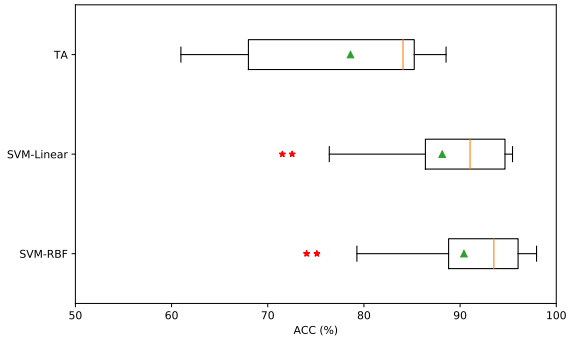


**Fig. 6.** Box plot of SVM and TA for 7 classes by using DS1.

## 4.3 Key Recovery Phase

The maximum likelihood estimation assumes that multiple power traces can be used to recover the secret key, thus the success rate is not suited as a measure. The guessing entropy [35] could be used to evaluate the number of remaining keys. The guessing entropy is defined as follows: let $g$ include the descending probability ranking of all possible keys and $i$ represent the position of the correct key in $g$. After performing $s$ experiments, one gets a matrix $[g_1, g_2, \ldots, g_s]$ and a corresponding vector $[i_1, i_2, \ldots, i_s]$. Namely, the guessing entropy represents the average amount of traces required

| | DS0 | | | DS1 | | |
|---|---|---|---|---|---|---|
| | **9 classes** | **256 classes** | **7 classes** | **9 classes** | **256 classes** | **7 classes** |
| **TA** | 21.89 | 18.97 | 13.34 | 18.14 | 14.36 | 10.12 |
| **SVM-Linear** | 13.42 | 12.24 | 9.45 | 11.33 | 9.88 | 5.23 |
| **SVM-RBF** | 11.29 | 9.51 | 5.46 | 9.75 | 7.36 | 3.97 |

**Tab. 6.** The number of power traces required by using SVM and TA when the guessing entropy is set to one ($GE^1$).

to recover the correct key. Hence, the guessing entropy is selected as a metric in the key recovery phase.

In this section, SVM and TA were used to recover the secret key when classifying 7, 9, and 256 classes. Instead of predicting the class $c$ of each trace, we gave the posterior conditional probability $P_{\mathrm{SVM}}(\mathbf{X}_i|c)$. The key that maximizes the log likelihood probability in (2) is selected as the correct key. As described in Sec. 3.1, for all possible subkeys $k_i{}^*$ (0x00 to 0xff) and the $i$th byte of a random plaintext $t_i$, the value of HW(Sbox $[t_i \oplus k_i{}^*]$) might be 0 or 8. Since the plaintext is subject to uniform distribution, the probability of the HW class 0 and 8 is about 0.8% in our datasets. Note that for 7 HW classes, we only gave the probability of each instance belonging to the HW classes 1 to 7 in the testing phase. To solve the problem, the probability belonging to the HW class 0 or 8 was considered to be one-tenth of the minimum value of these probabilities in terms of efficiency and effectiveness.

Figures 7, 8 and 9 report the guessing entropy of SVM-Linear, SVM-RBF, and TA as a function of the number of power traces used for respectively using DS1 to classify 7, 9, and 256 classes. The gray curves are used to describe the guessing entropy of 16 S-Boxes and their average is selected as the target results. In Tab. 6, we give the number of traces required by SVM and TA when the guessing entropy is set to 1 ($GE^1$). SVM-RBF requires a minimum number of power traces to recover the key when using DS1 to classify 7 classes, which only requires 3.97 traces in average. Due to the difference in power consumption between 16 S-Boxes, the number of traces required to recover the secret key varies greatly. Even more, for some S-Boxes, SVM-RBF only needs one trace to recover the key when classifying 7 or 256 classes. However, SVM and TA require more than one trace when classifying 9 binomial distributed HW classes.

Figure 10 illustrates the overall time required to perform an attack when using DS1 to classify 7, 9, and 256 classes. As the number of classes that we need to distinguish is reduced, the overall time of SVM and TA is greatly decreased. Compared with TA, SVM has a much lower computational burden. We can see that TA spends the most time and SVM-Linear needs the least time when classifying 7, 9, and 256 classes. In addition, the overall time required to classify 7 HW classes is much lower than that of 9 HW classes when using SVM. As described in the previous section, SVM (especially SVM-RBF) has a higher success rate and requires less number of power traces to recover the key when classifying 7 HW classes. In general, there is a trade-off to be made between accuracy and efficiency. In fact, this article is no exception. Since SMOTE is used to compensate for the distribution of HW classes, we need more time to obtain 7 uniformly distributed classes. However, SMOTE is executed before the parameter tuning phase, which can be independent of our attack phase. Therefore, the time required by SMOTE is not included in Fig. 10. From what has been discussed above, we firmly believe that SMOTE can solve the problem caused by data imbalance efficiently.

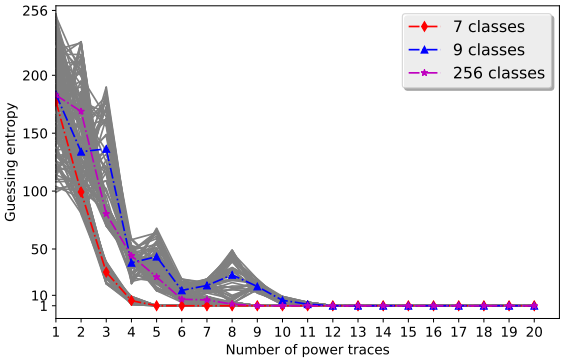


**Fig. 7.** The guessing entropy of SVM-Linear by using DS1.

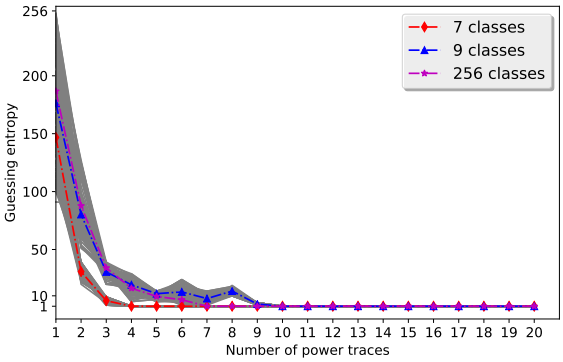


**Fig. 8.** The guessing entropy of SVM-RBF by using DS1.

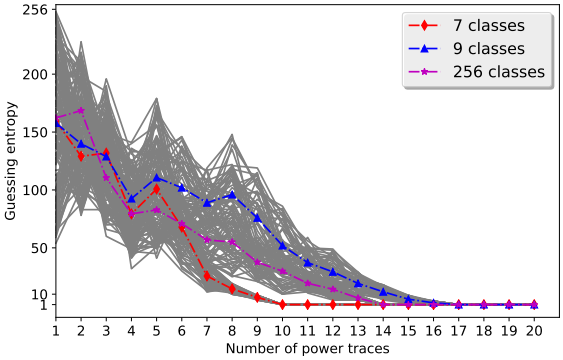


**Fig. 9.** The guessing entropy of TA by using DS1.

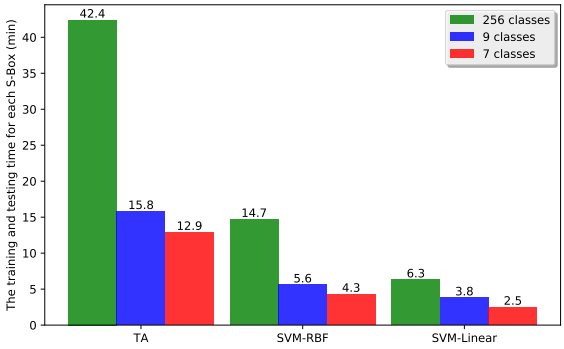


**Fig. 10.** The overall time required to perform a complete attack based on DS1 when using SVM and TA.

# 5. Conclusions

As described in the above section, PA attacks are considered to be the classification problem in the ML community. SVM and TA create features (templates) to characterize power traces of the training set and then calculate the similarity between these features and new traces of the testing set. Ultimately, the results are provided with a certain probability. Generally, TA assumes that sample points of power traces are approximated by a set of finite normal distributions. However, ML algorithms assume that sample points are subject to independent and identically distributed, but not limited to a certain distribution. Consequently, SVM can extract more information about the secret key than TA by analyzing the same interesting points.

This paper discusses the effect of the distribution of profiling traces belonging to each class on the performance of SVM in PA attacks. The SVM algorithm optimizes the overall accuracy without considering the distribution of profiling traces, which tends to perform poorly on highly skewed datasets. SMOTE is used to compensate for this deficiency when classifying the binomial distributed HW classes. The results demonstrate the success rate of 7 uniformly distributed classes is higher than that of 9 binomial ones for SVM and TA in the testing phase. Additionally, the performance of SVM with proper parameters is superior to that of TA. SVM-RBF requires an average of less than 4 power traces to recover the secret key when using DS1 to classify 7 classes. Further analysis indicates that SMOTE significantly improves the performance of SVM in terms of attack effectiveness and efficiency.

# References

[1] KOCHER, P. C., JAFFE, J., JUN, B. Differential power analysis. In *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*. London (UK), 1999, p. 388–397. DOI: 10.1007/3-540-48405-1_25

[2] CHARI, S., RAO, J., ROHATGI, P. Template attacks. In *Proceedings of the 4th International Workshop on Cryptographic Hardware and Embedded Systems-CHES 2002*. Redwood Shores (USA), 2002, p. 13–28. DOI: 10.1007/3-540-36400-5_3

[3] BRIER, E., CLAVIER, C., OLIVIER, F. Correlation power analysis with a leakage model. In *Proceedings of the 6th International Workshop on Cryptographic Hardware and Embedded Systems*. Cambridge (USA), 2004, p. 16–29. DOI: 10.1007/978-3-540-28632-5_2

[4] SCHINDLER, W., LEMKE, K., PAAR, C. A stochastic model for differential side channel cryptanalysis. In *Proceedings of the 7th International Workshop on Cryptographic Hardware and Embedded Systems*. Edinburgh (UK), 2005, p. 30–46. DOI: 10.1007/11545262_3

[5] MANGARD, S., OSWALD, E., POPP, T. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. 1st ed. Secaucus (USA): Springer US, 2007. ISBN: 978-0-387-30857-9. DOI: 10.1007/978-0-387-38162-6

[6] RIVEST, R. L. Cryptography and machine learning. In *Proceedings of the International Conference on the Theory and Applications of Cryptology: Advances in Cryptology (ASIACRYPT)*. 1991, Fujiyoshida (Japan), p. 427–439. DOI: 10.1007/3-540-57332-1_36

[7] MARTINASEK, Z., ZEMAN, V. Innovative method of the power analysis. *Radioengineering*, 2013, vol. 22, no. 2, p. 586–594. ISSN: 1805-9600

[8] MARTINASEK, Z., HAJNY, J., MALINA, L. Optimization of power analysis using neural network. In *Proceedings of the 12th International Conference Smart Card Research and Advanced Applications (CARDIS)*. Berlin (Germany), 2013, p. 94–107. DOI: 10.1007/978-3-319-08302-5_7

[9] WHITNALL, C., OSWALD, E. Robust profiling for DPA-style attacks. In *Proceedings of the 17th International Workshop Cryptographic Hardware and Embedded Systems (CHES)*. Saint-Malo (France), 2015, p. 3–21. DOI: 10.1007/978-3-662-48324-4_1

[10] MARTINASEK, Z., ZEMAN, V., MALINA, L., et al. k-Nearest neighbors algorithm in profiling power analysis attack. *Radioengineering*, 2016, vol. 25, no. 2, p. 365–382. DOI: 10.13164/re.2016.0365

[11] HOSPODAR, G., GIERLICHS, B., MULDER, D. E., et al. Machine learning in side-channel analysis: A first study. *Journal of Cryptographic Engineering*, 2011, vol. 1, no. 4, p. 293–302. DOI: 10.1007/s13389-011-0023

[12] HE, H., JAFFE, J., ZOU, L. *Side Channel Cryptanalysis Using Machine Learning: Using an SVM to Recover DES Keys from a Smart Card*. 2012, Stanford University - CS229 Fall Project

[13] HEUSER, A., ZOHNER, M. Intelligent machine homicide - breaking cryptographic devices using support vector machines. In *Proceedings of the Constructive Side-Channel Analysis and Secure Design: Third International Workshop (COSADE)*. Darmstadt(Germany), 2012, p. 249–264. DOI: 10.1007/978-3-642-29912-4_18

[14] BARTKEWITZ, T., LEMKE-RUST, K. Efficient template attacks based on probabilistic multi-class support vector machines. In *Proceedings of the Smart Card Research and Advanced Applications*. Graz (Austria), 2013, p. 263–276. DOI: 10.1007/978-3-642-37288-9_18

[15] LERMAN, L., MEDEIROS, S. F., BONTEMPI, G., et al. A machine learning approach against a masked AES. In *Proceedings of the 12th International Conference of Smart Card Research and Advanced Applications (CARDIS)*. Berlin (Germany), 2013, p. 61–75. DOI: 10.1007/978-3-319-08302-5_5

[16] HOU, S. R., ZHOU, Y. J., LIU, H. M., et al. Wavelet support vector machine algorithm in power analysis attacks. *Radioengineering*, 2017, vol. 26, no. 3, p. 890–902. DOI: 10.13164/re.2017.0890

[17] VEROPOULOS, K., CAMPBELLL, C., CRISTIANINI, N. Controlling the sensitivity of support vector machines. In *Proceedings of the International Joint Conference on Artificial Intelligence*. Stockholm (Sweden), 1999, p. 55–60. DOI: 10.1.1.42.7895

[18] HSU, C. W., LIN, C. J. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 2002, vol. 13, no. 2, p. 415–425. DOI: 10.1109/72.991427

[19] CHAWLA, N. V., BOWYER, K. W., HALL, L. O., et al. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 2002, vol. 16, p. 321–357. DOI: 10.1613/jair.953

[20] AKBANI, R., KWEK, S., JAPKOWICZ, N. Applying support vector machines to imbalanced datasets. In *Proceedings of the15th European Conference on Machine Learning*. Pisa (Italy), 2004, p. 39–50. DOI: 10.1007/978-3-540-30115-8_7

[21] CORTES, C., VAPNIK, V. Support-vector networks. *Machine Learning*, 1995, vol. 20, no. 3, p. 273–297. DOI: 10.1007/BF00994018

[22] HANG, L. *Statistical Learning Method.* 1st ed. Beijing (China): Tsinghua University press, 2012. ISBN: 978-7-302-27595-4

[23] LIN, H. T., LIN, C. J., WENG, R. C. A note on platt's probabilistic outputs for support vector machines. *Machine Learning*, 2007, vol. 63, no. 3, p. 267–276. DOI: 10.1007/s10994-007-5018-6

[24] GUILLEYHO, S. *DPA contest v4*. [Online] Cited 2017-10-24. Available at: http://www.dpacontest.org/v4/rsm_doc.php

[25] PENG, H., LONG, F., DING, C. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005, vol. 27, no. 8, p. 1226–1238. DOI: 10.1109/TPAMI.2005.159

[26] GIERLICHS, B., LEMKE-RUST, K., PAAR, C. Templates vs. stochastic methods. In *Proceedings of the 8th International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*. Yokohama (Japan), 2006, p. 15–29. DOI: 10.1007/11894063_2

[27] CHAWLA, N. V., JAPKOWICZ, N., KOTCZ, A. Editorial: Special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 2004, vol. 6, no. 1, p. 1–6. DOI: 10.1145/1007730.1007733

[28] HAIBO, H., GARCIA, E. A. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 2009, vol. 21, no. 9, p. 1263–84. DOI: 10.1109/TKDE.2008.239

[29] LEMAITRE, G., NOGUEIRA, F., ARIDAS, C. K. Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 2017, vol. 18, no. 17, p. 1–5

[30] CHANG, C. C., LIN, C. J. A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*. 2011, vol. 2, no. 27. DOI: 10.1145/1961189.1961199

[31] HSU, C. W., CHANG, C. C., LIN, C. J. A practical guide to support vector classification. *BJU International*, 2008. DOI: 10.1177/02632760022050997

[32] LEMAITRE, G., NOGUEIRA, F., ARIDAS, C. K. *Imbalanced-Learn*. [Online] Cited 2017-10-24. Available at: https://github.com/scikit-learn-contrib/imbalanced-learn

[33] POWERS, D. M. W. Evaluation: From precision, recall and F-measure to roc, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2011, vol. 2, p. 37–63. DOI: 10.1.1.214.9232

[34] DAVIS, J., GOADRICH, M. The relationship between precision-recall and ROC curves. In *Proceedings of the 23rd International Conference on Machine Learning*. New York (USA), 2006, p. 233–240. DOI: 10.1145/1143844.1143874

[35] STANDAERT, F. X., MALKIN, T. G., YUNG, M. A unified framework for the analysis of side-channel key recovery attacks. In *Proceedings of the 28th Annual International Conference on Advances in Cryptology: the Theory and Applications of Cryptographic Techniques*. Berlin (Germany), 2009, p. 443–461. DOI: 10.1007/978-3-642-01001-9_26

## About the Authors . . .

**Shourong HOU** was born in Shandong, China. He received his B.Eng. from Xidian University in 2015. His research interests include machine learning and side channel attack. He is now a Ph.D. candidate at Shanghai Jiao Tong University.

**Yujie ZHOU** was born in Zhejiang, China. She received his Ph.D. from University of Science and Technology of China in 1997. Her research interests include digital integrated circuit design, embedded system and digital copyright protection.

**Hongming LIU** was born in Jiangxi, China. He received his Ph.D. from Shanghai Jiao Tong University in 2014. His research interests include chip design and machine learning.

**Nianhao ZHU** was born in Jiangshu, China. She received his Ph.D. from Shanghai Jiao Tong University in 2014. His research interests include cryptographic chip design and side channel attack.