Assignment 1:
CS771: Intro to Machine learning

Group name: Classyfiers
Keshav Raj Gupta (210508) Arnesh Dadhich (210189) Amol Patil (210711)
Ayush Himmatsinghka(210245) Vishal Himmatsinghka(211175)

Problem 1.1: Companion Arbiter PUF

Part 1:

By giving a detailed mathematical derivation (as given in the lecture slides), show how a CAR-PUF can be broken by a single linear model. Give derivations for a map $\phi : \{0,1\}^{32} \to \mathbb{R}^D$ mapping 32-bit 0/1-valued challenge vectors to D-dimensional feature vectors (for some D > 0) so that for any CAR-PUF, there exists a D-dimensional linear model $W \in R^D$ and a bias term $b \in R$ such that for all CRPs (c, r) with $c \in \{0,1\}^{32}$, $r \in \{0,1\}$, we have

$$\frac{1 + \text{sign}(W^T \phi(c) + b)}{2} = r$$

Ans:

We know, for an arbiter PUF,

$\Delta = w_0 x_0 + w_1 x_1 + \cdots + w_{31} x_{31} + \beta_{31} = w^T x + b$

where, conversion of $c_i \to x_i$ is as follows:
$x_i = d_i \cdot d_{i+1} \cdot \cdots \cdot d_{31}$
$d_i = 1 - 2c_i$
$w_0 = \alpha_0$
$w_i = \alpha_i + \beta_{i-1}$

Clearly, $x \in \{-1, 1\}^{32}$. Also, α_i and β_i are the intrinsic delays of the i^{th} mux and b is the bias term.

Since Melbo is using two PUFs, namely working and reference, let the working PUF be denoted by the linear model (u, p) and the reference PUF be denoted by the linear model (v, q). i.e.

$\Delta_w = u^T x + p$

$\Delta_r = v^T x + q$

For Melbo's CAR-PUF, as mentioned,

$$f(x) = \begin{cases} 1, & \text{if } |\Delta_w - \Delta_r| > \tau \\ 0, & otherwise \end{cases}$$

equivalently,

$$f(x) = \begin{cases} 1, & \text{if } |\Delta_w - \Delta_r|^2 > \tau^2 \\ 0, & otherwise \end{cases}$$

Preprint. Under review.

Hence,

$$r = \frac{1 + \text{sign}(|\Delta_w - \Delta_r|^2 - \tau^2)}{2}$$

Let $f = |\Delta_w - \Delta_r|^2 - \tau^2$

$$\begin{aligned} f &= |u^T x + p - v^T x - q|^2 - \tau^2 \\ &= |(u-v)^T x + p - q|^2 - \tau^2 \end{aligned}$$

Let's call $u - v = z$ vector and $p - q = k$ constant

$$\begin{aligned} f &= |(z^T x + k)|^2 - \tau^2 \\ &= \left[\left(\sum_{i=0}^{31} z_i x_i\right) + k\right]^2 - \tau^2 \\ &= \left(\sum_{i=0}^{31} z_i x_i\right)^2 + 2k\left(\sum_{i=0}^{31} z_i x_i\right) + k^2 - \tau^2 \\ &= \sum_{i=0}^{31} (z_i x_i)^2 + 2\sum_{i=0}^{31}\sum_{j>i}^{31} z_i z_j z_i z_j + 2k\sum_{i=0}^{31} z_i z_i + k^2 - \tau^2 \\ &= 2\sum_{i=0}^{31}\sum_{j>i}^{31} z_i z_j x_i x_j + 2k\sum_{i=0}^{31} z_i x_i + \sum_{i=0}^{31} (z_i)^2 + k^2 - \tau^2 \text{.....eqn(1)} \\ &= \widetilde{\mathbf{z}}^T \widetilde{\mathbf{x}} + \widetilde{k} \end{aligned}$$

The eqn (1) represents a new linear model $(\widetilde{\mathbf{z}}^T, \widetilde{k})$, which has a total of 528 dimensions (496 terms from the first term of the eqn (1) and 32 from the second) and a constant.

$\widetilde{z}$ is the new weights vector [1×528] and $x \in \{-1, 1\}^{528}$ as defined before

Hence,

$$r_{\text{final}} = \frac{1 + \text{sign}(\widetilde{\mathbf{z}}^T \widetilde{\mathbf{x}} + \widetilde{k})}{2}$$

For any particular challenge c we define a map $\phi(c)$ in such a way:-

$$\phi(\mathbf{c}) = \widetilde{\mathbf{x}}(c \to \mathbf{x})$$

The mapping function $\phi(c) : \{0,1\}^{32} \to \mathbb{R}^{528}$ transforms 32-bit binary-valued challenge vectors into 528-dimensional feature vectors.

For every Capacitive Physically Unclonable Function (CAR-PUF), there exists a 528-dimensional linear model $\widetilde{\mathbf{z}} \in \mathbb{R}^{528}$ and a bias term $\widetilde{k} \in \mathbb{R}$ such that for all Challenge-Response Pairs (CRPs) (c, r) where $c \in \{0,1\}^{32}$ and $r \in \{0,1\}$, the following holds:

$$r = \frac{1 + \text{sign}(\widetilde{\mathbf{z}}^T \phi(\mathbf{c}) + \widetilde{k})}{2}$$

Therefore we can write as:-
$|\widetilde{z}^\top \widetilde{x}| > \tau$

which is equivalent to
$(\widetilde{z}^\top \widetilde{x})^2 > \tau^2$

and further equivalent to
$\sum_{i=0}^{31} (\widetilde{z}_i \widetilde{x}_i)^2 + \sum_{i=0}^{31} \sum_{j \neq i}^{31} \widetilde{z}_i \widetilde{x}_i \widetilde{x}_j - \tau^2 > 0.$

This can be expressed again as a linear equation
$Z = [\widetilde{z}_0 \cdot \widetilde{z}_1, \widetilde{z}_0 \cdot \widetilde{z}_2, ..., \widetilde{z}_i \cdot \widetilde{z}_j, ..., \widetilde{z}_{30} \cdot \widetilde{z}_{31}]$,
$\phi(c) = [\widetilde{x}_0 \cdot \widetilde{x}_1, \widetilde{x}_0 \cdot \widetilde{x}_2, ..., \widetilde{x}_i \cdot \widetilde{x}_j, ..., \widetilde{x}_{30} \cdot \widetilde{x}_{31}]$,
$k = \sum_{i=0}^{31} (\hat{z}_i)^2 - \tau^2$.

Thus we finally obtained:-

$$\mathbf{Z}^T \phi(\mathbf{c}) + k > 0$$

Thus we mathematically derived that, how the given CAR-PUF can be broken by a single linear model.

Part 2:

Code Outline: In this section, our task is to implement the mymap() and myfit() functions. These functions serve the purpose of mapping the challenges to a D-dimensional vector and learning the linear model, respectively. To accomplish this, we start by preprocessing the training data, converting 0s to -1s to ensure compatibility with certain algorithms. Next, we generate the final feature map by taking the product of the features with themselves, selecting 2 at a time using loops. We utilize various regression models, experimenting with different algorithms and hyperparameters to achieve optimal performance. Through this iterative process of model selection and parameter tuning, we aim to obtain the best results for our task.

Part 3:
Experimenting with parameters: In this section, we see the effect of varying hyperparameters for both LinearSVC and Logistic Regression, in particular their effect on training time and test accuracy. Below follow tables indicating the variation of model parameters on changing Loss Hyperparameter and Value of C for LinearSVC, and Value of C for Logistic Regression:

Table 1: Changing the Loss Hyperparameter (Hinge v/s Squared Hinge Loss) in LinearSVC

Loss Hyperparameter	**t_train**	**t_map**	**Accuracy**
Squared-hinge	95.69	0.074	99.198
Hinge	98.55	0.075	99.191

Thus we see that for LinearSVC, the performance of Squared Hinge Loss is slightly better than Hinge Loss for accuracy while the prior takes slightly lesser time than the later for training. Here,value of C=100 and tol=0.1 and is not changed.

Table 2: Varying C in LinearSVC

Regularization parameter(C)	**t_train**	**t_map**	**Accuracy**
0.001	2.241	0.043	95.97
0.01	4.523	0.042	98.65
0.1	14.10	0.045	98.99
1	12.68	0.049	99.13
10	11.78	0.044	99.02
100	11.80	0.044	99.03
1000	11.43	0.043	98.97

Table 3: Varying C in Logistic Regression

Regularization parameter(C)	**t_train**	**t_map**	**Accuracy**
0.001	0.670	0.049	90.69
0.01	0.773	0.048	96.35
0.1	0.984	0.063	98.71
1	1.189	0.060	99.07
10	1.336	0.058	99.22
100	1.847	0.056	99.30
1000	2.625	0.054	99.23

By varying the values of the regularization parameter C from low to medium and high for both Logistic Regression and LinearSVC models, we observed significant changes in the accuracy of the models. Generally, increasing the value of C, which acts as a regularization constant, improved the accuracy of the models. However, it's worth noting that increasing C too much may slightly decrease the accuracy, as it does not guarantee continuous improvement.

The observed trend suggests that regularization plays a crucial role in model performance. By tuning the regularization parameter, we can effectively control the trade-off between fitting the training data well and preventing overfitting.

Table 4: Varying tol in LinearSVC

Tolerance (tol)	**t_train**	**t_map**	**Accuracy**
0.001	12.22	0.044	99.14
0.1	11.58	0.043	99.11
100	0.854	0.046	93.42

Table 5: Varying tol in Logistic Regression

Tolerance (tol)	**t_train**	**t_map**	**Accuracy**
0.001	1.192	0.061	99.07
0.1	1.051	0.051	99.07
100	0.653	0.055	98.54

While changing different values of Tolerance, it is observed that increasing the tolerance parameter during model training often leads to a decrease in accuracy. This occurs because higher tolerance values relax the convergence criteria for optimization algorithms, causing them to terminate prematurely before reaching the optimal solution. As a result, the trained model may not capture the underlying data information accurately, resulting in reduced accuracy.