1 Question

By giving a detailed mathematical derivation (as given in the lecture slides), show how for a simple arbiter PUF, a linear model can predict the time it takes for the upper signal to reach the finish line. Specifically, give derivations for a map $\varphi : \{0,1\}^{32} \to \mathbb{R}^D$ mapping 32-bit 0/1-valued challenge vectors to D-dimensional feature vectors (for some $D > 0$) so that for any arbiter PUF, there exists a D-dimensional linear model $W \in \mathbb{R}^D$ and a bias term $b \in \mathbb{R}$ such that for all CRPs $c \in \{0,1\}^{32}$, we have $W^\top \varphi(c) + b = t_u(c)$ where $t_u(c)$ is the time it takes for the upper signal to reach the finish line when challenge c is input. Remember that $t_u(c)$ is, in general, a non-negative real number (say in milliseconds) and need not be a Boolean bit. W, b may depend on the PUF-specific constants such as delays in the multiplexers. However, the map $\varphi(c)$ must depend only on c (and perhaps universal constants such as $\sqrt{2}$, 2 etc). The map φ must not use PUF-specific constants such as delays.

1.1 Answer

We know that for each PUF model, each multiplexer takes time as following for its upper and lower signals to pass :
$t_2^u = (1 - c_2) \cdot (t_1^u + p_2) + c_2 \cdot (t_1^l + s_2)$

$t_2^l = (1 - c_2) \cdot (t_1^l + q_2) + c_2 \cdot (t_1^u + r_2)$

First we see two models:
One, that gives the difference between the two signals upper and lower
Second, That gives the sum of both signals upper and lower

We will then add the outputs of the the two models and divide by 2 to give the time of upper signal
Similarly, on subtracting one model from another and dividing by 2, we get the time of lower signal.

MODEL 1: Based on time difference of upper and lower signals:
Let us use the shorthand $\Delta_i = t_i^u - t_i^l$ to denote the lag.
Note: response is 0 if $\Delta_{64} < 0$ else response is 1

$\Delta_2 = (1 - c_2) \cdot (t_1^u + p_2 - t_1^l - q_2) + c_2 \cdot (t_1^l + s_2 - t_1^u - r_2)$
$= (1 - 2c_2) \cdot \Delta_1 + (q_2 - p_2 + s_2 - r_2) \cdot c_2 + (p_2 - q_2)$

To make notation simpler, let

$\mathrm{d}_i \stackrel{\text{def}}{=} (1 - 2c_i)$
$\alpha_i \stackrel{\text{def}}{=} (p_i - q_i + r_i - s_i)/2$
$\beta_i \stackrel{\text{def}}{=} (p_i - q_i - r_i + s_i)/2$

$\Delta_2 = \Delta_1 \cdot d_2 + \alpha_2 \cdot \ d_2 \ + \ \beta_2$

A similar relation holds for any stage:

$$\Delta_i = d_i \cdot \Delta_{i-1} + \alpha_i \cdot d_i + \beta_i$$

We can safely take $\Delta_0 = 0$ (absorb initial delays into p_1, q_1, r_1, s_1).

We can keep going on recursively:

$\Delta_1 = \alpha_1 \cdot d_1 + \beta_1$ since $\Delta_0 = 0$,
$\Delta_2 = \Delta_1 \cdot d_2 + \alpha_2 \cdot d_2 + \beta_2$ now plug in the value of Δ
$\quad = \alpha_1 \cdot d_2 \cdot d_1 + (\alpha_2 + \beta_1) \cdot d_2 + \beta_2,$
$\Delta_3 = \alpha_1 \cdot d_3 \cdot d_2 \cdot d_1 + (\alpha_2 + \beta_1) \cdot d_3 \cdot d_2 + (\alpha_3 + \beta_2) \cdot d_3 + \beta_3,$
$\Delta_4 = \alpha_1 \cdot d_4 \cdot d_3 \cdot d_2 \cdot d_1 + (\alpha_2 + \beta_1) \cdot d_4 \cdot d_3 \cdot d_2 + (\alpha_3 + \beta_2) \cdot d_4 \cdot d_3 + (\alpha_4 + \beta_3) \cdot d_4 + \beta_4.$

It turns out that

$$\Delta_{32} = \mathbf{w}^T \mathbf{x} + b$$

where $\mathbf{x} = [x_1, \ldots, x_{32}]$, $\mathbf{w} = [w_1, \ldots, w_{32}] \in \mathbb{R}^{32}$

$$x_i = d_i \cdot d_{i+1} \cdot \ldots \cdot d_{32}$$
$$= (1 - 2c_i) \cdot (1 - 2c_{i+1}) \cdot \ldots \cdot (1 - 2c_{32})$$

$$w_1 = \alpha_1$$
$$w_i = \alpha_i + \beta_{i-1} \quad (\text{for } i = 2, 3, \ldots, 32)$$
$$b = \beta_{32}$$

MODEL 2: Based on time summation of upper and lower signals:
Let us use the shorthand $\sigma_i = t_i^u + t_i^l$ to denote the sum.
$\sigma_2 = (1 - c_2) \cdot (t_1^u + p_2 + t_1^l + q_2) + c_2 \cdot (t_1^l + s_2 + t_1^u + r_2)$
$= \sigma_1 + (1 - c_2) \cdot (q_2 + p_2) + c_2 \cdot (s_2 + r_2)$

To make notation simpler, let

$\mathrm{d}_i \stackrel{\text{def}}{=} (1 - 2c_i)$
$a_i \stackrel{\text{def}}{=} (p_i + q_i - r_i - s_i)/2$
$b_i \stackrel{\text{def}}{=} (p_i + q_i + r_i + s_i)/2$

$\sigma_2 = \sigma_1 + a_2 \cdot d_2 + b_2$

A similar relation holds for any stage:

$$\sigma_i = \sigma_{i-1} + a_i \cdot d_i + b_i$$

We can safely take $\sigma_0 = 0$ (absorb initial delays into p_1, q_1, r_1, s_1).

We can keep going on recursively:

$\sigma_1 = a_1 \cdot d_1 + b_1$	since $\sigma_0 = 0$,
$\sigma_2 = \sigma_1 + a_2 \cdot d_2 + b_2$	now plug in the value of σ_1 to get,
$\quad = a_1 \cdot d_1 + b_1 + a_2 \cdot d_2 + b_2,$	
$\sigma_3 = a_1 \cdot d_1 + b_1 + a_2 \cdot d_2 + b_2 + a_3 \cdot d_3 + b_3$	

It turns out that

$$\sigma_{32} = \mathbf{w}^T \mathbf{x} + b$$

where $\mathbf{x} = [x_1, \ldots, x_{32}]$, $\mathbf{w} = [w_1, \ldots, w_{32}] \in \mathbb{R}^{32}$

$$x_i = d_i$$
$$= (1 - 2c_i)$$

$$w_1 = a_1$$
$$w_i = a_i (\text{for } i = 2, 3, \ldots, 32)$$

b = $\mathrm{b}_1 + b_2 + b_3 \ldots b_{32}$

Now we have our two models whose summation will give us the time taken by upper signal to reach final arbiter.

$t_1^u = (\sigma_1 + \Delta_1)/2$

similarly for any i,

$t_i^u = (\sigma_i + \Delta_i)/2$

Now we can clearly write this in terms of a map $\varphi(c)$ which gives the relation for $W^\top \varphi(c) + B = t_u(c)$ where $t_u(c)$

Lets define the terms in the final model for $t_u(c)$ as:
$2 \cdot W = [w_1, \ldots, w_{31}, a_1, a_2, a_3, \ldots, a_{32} + w_{32}]$

where $w_i = \alpha_i + \beta_{i-1}$ from model 1
and $a_i \stackrel{\text{def}}{=} (p_i + q_i - r_i - s_i)/2$ is from the model 2

NOTE: term of c multiplying with w_{32} and a_{32} is the same$(1 - 2 \cdot c_{32})$which will be one term on expansion of linear model

$\varphi(c) = [x_1, x_2, \ldots, x_{31}, d_1, d_2, \ldots, d_{32}]$

where $x_i = d_i \cdot d_{i+1} \cdot \ldots \cdot d_{32}$ is from Model 1 and d_i is from Model 2.

lastly,
$2 \cdot B = [\beta_{32} + b_1 + b_2 + \ldots + b_{32}]$

Thus we have a linear model $W^\top \varphi(c) + B = t_u(c)$ which gives us the value of $t^u(c)$ for all inputs.

2 Question

What dimensionality does the linear model need to have to predict the arrival time of the upper signal for an arbiter PUF? The dimensionality should be stated clearly and separately in your report, and not be implicit or hidden away in some calculations.

2.1 Answer

The dimensionality of W will be defined by number of independent terms in $\varphi(c)$.

$\varphi(c) = [x_1, x_2, \ldots, x_{31}, d_1, d_2, \ldots, d_{32}]$

Here $x_{32} = d_{32} = (1 - 2 \cdot c_{32})$

Hence they will combine to form one term whose coefficient in W will be given by $a_{32} + w_{32}$.

So in all we had 32 terms from Model 1 and 32 terms from Model 2. When combined together to give time of upper signal, the last terms being the same combine to form a single term.

Therefore the total terms are $= 32 + 32 - 1 = 63$

Hence we get dimensionality of W and **D=63**

3 Question

Use the derivation from Part 1 to show how a linear model can predict Response0 for a COCO-PUF. As in Part 1, provide an explicit map $\tilde{\phi} : \{0,1\}^{32} \to \mathbb{R}^D$ and a corresponding linear model $\tilde{W} \in \mathbb{R}^D, \tilde{b} \in \mathbb{R}$ that predicts the responses, i.e., for all CRPs $c \in \{0,1\}^{32}$, we have $1 + \text{sign}(\tilde{W}^\top \tilde{\phi}(c) + \tilde{b}) = r_0(c)$, where $r_0(c)$ is Response0 on the challenge c.

Similarly, demonstrate how a linear model can predict Response1 for a COCO-PUF. As before, your linear model may depend on the delay constants in PUF0 and PUF1, but your map must not use PUF-specific constants such as delays.

3.1 Answer

Using the derivation from the first question, we can easily calculate the Response 1.

1.)Firstly, we will calculate the time taken by the upper signal of the PUF-1 to reach the arbiter1.Let's call it t_1^u

2.)Similarly we will calculate the time taken by the upper signal of the PUF-0. to reach the arbiter 1.Let's call it t_0^u

.

3.) To find the Response-1, we will calculate the difference of both these times i.e($t_0^u - t_1^u$) . If this difference is negative, response-1 will be 0,else 1.

Response=$((1 + sign(t_0^u - t_1^u))/2)$

Expanding $t_0^u - t_1^u$:

t_0^u can be written as: $W^\top \varphi(c) + B = t_0^u$ (where W is a 63 dimension vector consisting of terms which are made of internal flipflops delays (p_i, q_i, r_i, s_i) from PUF-0 and $\varphi(c)$ is our map consisting of $C_i's$ and is also a 63 dimension vector.) .

t_1^u can be written as: $\tilde{W}^\top \varphi(c) + \tilde{B} = t_1^u$ (where $\tilde{W}$ is a 63 dimension vector consisting of terms which are made of internal flipflops delays (p_i, q_i, r_i, s_i) from PUF-1 and $\varphi(c)$ is our map consisting of $C_i's$ and is also a 63 dimension vector.)

Redefining terms from question-1:
$2 \cdot W = [w_1, \ldots, w_{31}, a_1, a_2, a_3, \ldots, a_{32} + w_{32}]$

where $w_i = \alpha_i + \beta_{i-1}$ from model 1
and $a_i \stackrel{\text{def}}{=} (p_i + q_i - r_i - s_i)/2$ is from the model 2

$\varphi(c) = [x_1, x_2, \ldots, x_{31}, d_1, d_2, \ldots, d_{32}]$

where $x_i = d_i \cdot d_{i+1} \cdot \ldots \cdot d_{32}$ is from Model 1 and d_i is from Model 2.

lastly,
$2 \cdot B = [\beta_{32} + b_1 + b_2 + \ldots + b_{32}]$

But the most interesting thing is that in the calculation of both t_0^u and t_1^u is the select bit inputs that the user provides, more specifically $\varphi(c)$ will be same in both the calculations since the select bit inputs are same for the puffs.

Thus $t_0^u - t_1^u = \hat{W}^\top \varphi(c) + \hat{B}$

Where $\hat{W} = W - \tilde{W}$ and $\hat{B} = B - \tilde{B}$

More specifically,

$2 \cdot \hat{W} = [w_1 - \tilde{w_1}, \ldots, w_{31} - \tilde{w_{31}}, a_1 - \tilde{a_1}, a_2 - \tilde{a_2}, a_3 - \tilde{a_3}, \ldots, a_{32} - \tilde{a_{32}} + w_{32} - \tilde{w_{32}}]$

where $w_i = \alpha_i + \beta_{i-1}$ from model 1
and $a_i \stackrel{\text{def}}{=} (p_i + q_i - r_i - s_i)/2$ is from the model 2

Similarly $\varphi(c) = [x_1, x_2, \ldots, x_{31}, d_1, d_2, \ldots, d_{32}]$

where $x_i = d_i \cdot d_{i+1} \cdot \ldots \cdot d_{32}$ is from Model 1 and d_i is from Model 2.

Now simply Response1 will be $(1 + sign(\hat{W}^\top \varphi(c) + \hat{B}))/2$

Now to find the Response 0,first we will have to find the general formula for time take by the lower signal to reach the arbiter, denoted by $t^l(c)$

Using our previous knowledge:
$t_i^l = (\sigma_i - \Delta_i)/2$

Now we can clearly write this in terms of a map $\varphi(c)$ which gives the relation for $W^\top \varphi(c) + B = t_l(c)$

Lets define the terms in the final model for $t_l(c)$ as:
$2 \cdot W = [-w_1, \ldots, -w_{31}, a_1, a_2, a_3, \ldots, a_{32} - w_{32}]$

where $w_i = \alpha_i + \beta_{i-1}$ from model 1
and $a_i \stackrel{\text{def}}{=} (p_i + q_i - r_i - s_i)/2$ is from the model 2

NOTE: term of c multiplying with w_{32} and a_{32} is the same$(1 - 2 \cdot c_{32})$which will be one term on expansion of linear model

$\varphi(c) = [x_1, x_2, \ldots, x_{31}, d_1, d_2, \ldots, d_{32}]$

where $x_i = d_i \cdot d_{i+1} \cdot \ldots \cdot d_{32}$ is from Model 1 and d_i is from Model 2.

lastly,
$2 \cdot B = [-\beta_{32} + b_1 + b_2 + \ldots + b_{32}]$

Thus we have a linear model $W^\top \varphi(c) + B = t^l(c)$ which gives us the value of $t^l(c)$ for all inputs.

1.)Firstly, we will calculate the time taken by the lower signal of the PUF-1 to reach the arbiter0.Let's call it t_1^l

2.)Similarly we will calculate the time taken by the lower signal of the PUF-0. to reach the arbiter 0.Let's call it t_0^l
.

3.) To find the Response-0, we will calculate the difference of both these times i.e$(t_0^l - t_1^l)$. If this difference is negative, response-0 will be 0,else 1.

Response=$((1 + sign(t_0^l - t_1^l))/2)$

Expanding $t_0^l - t_1^l$:

t_0^l can be written as: $W^\top \varphi(c) + B = t_0^l$ (where W is a 63 dimension vector consisting of terms which are made of internal flipflops delays (p_i, q_i, r_i, s_i) from PUF-0 and $\varphi(c)$ is our map consisting of $C_i's$ and is also a 63 dimension vector.) .

t_1^l can be written as: $\tilde{W}^\top \varphi(c) + \tilde{B} = t_1^l$ (where $\tilde{W}$ is a 63 dimension vector consisting of terms which are made of internal flipflops delays (p_i, q_i, r_i, s_i) from PUF-1 and $\varphi(c)$ is our map consisting of $C_i's$ and is also a 63 dimension vector.)

Redefining terms:
$2 \cdot W = [-w_1, \ldots, -w_{31}, a_1, a_2, a_3, \ldots, a_{32} - w_{32}]$

where $w_i = \alpha_i + \beta_{i-1}$ from model 1
and $a_i \stackrel{\text{def}}{=} (p_i + q_i - r_i - s_i)/2$ is from the model 2

$\varphi(c) = [x_1, x_2, \ldots, x_{31}, d_1, d_2, \ldots, d_{32}]$

where $x_i = d_i \cdot d_{i+1} \cdot \ldots \cdot d_{32}$ is from Model 1 and d_i is from Model 2.

lastly,
$2 \cdot B = [-\beta_{32} + b_1 + b_2 + \ldots + b_{32}]$

But the most interesting thing is that in the calculation of both t_0^l and t_1^l is the select bit inputs that the user provides, more specifically $\varphi(c)$ will be same in both the calculations since the select bit inputs are same for the puffs.

Thus $t_0^l - t_1^l = \hat{W}^\top \varphi(c) + \hat{B}$

Where $\hat{W} = W - \tilde{W}$ and $\hat{B} = B - \tilde{B}$

More specifically,

$2 \cdot \hat{W} = [-w_1 + \tilde{w_1}, \ldots, -w_{31} + \tilde{w_{31}}, a_1 - \tilde{a_1}, a_2 - \tilde{a_2}, a_3 - \tilde{a_3}, \ldots, a_{32} - \tilde{a_{32}} - w_{32} + \tilde{w_{32}}]$

where $w_i = \alpha_i + \beta_{i-1}$ from model 1
and $a_i \stackrel{\text{def}}{=} (p_i + q_i - r_i - s_i)/2$ is from the model 2

Similarly $\varphi(c) = [x_1, x_2, \ldots, x_{31}, d_1, d_2, \ldots, d_{32}]$

where $x_i = d_i \cdot d_{i+1} \cdot \ldots \cdot d_{32}$ is from Model 1 and d_i is from Model 2.

Now simply Response0 will be $(1 + sign(\hat{W}^\top \varphi(c) + \hat{B}))/2$

4 Question

What dimensionality do your need the linear model to have to predict Response0 and Response1 for a COCO-PUF? This may be the same or different from the dimensionality you needed to predict the arrival times for the upper signal in a simple arbiter PUF. The dimensionality should be stated clearly and separately in your report, and not be implicit or hidden away in some calculations.

4.1 Answer

The dimensionality of $\hat{W}$ will be defined by number of independent terms in $\varphi(c)$.

$\varphi(c) = [x_1, x_2, \ldots, x_{31}, d_1, d_2, \ldots, d_{32}]$

Here $x_{32} = d_{32} = (1 - 2 \cdot c_{32})$

We know that:

for Response 1:

$$(2 \cdot \hat{W}) = [w_1 - \tilde{w_1}, \ldots, w_{31} - \tilde{w_{31}}, a_1 - \tilde{a_1}, a_2 - \tilde{a_2}, a_3 - \tilde{a_3}, \ldots, a_{32} - \tilde{a_{32}} + w_{32} - \tilde{w_{32}}]$$

for Response 0:

$$(2 \cdot \hat{W}) = [-w_1 + \tilde{w_1}, \ldots, -w_{31} + \tilde{w_{31}}, a_1 - \tilde{a_1}, a_2 - \tilde{a_2}, a_3 - \tilde{a_3}, \ldots, a_{32} - \tilde{a_{32}} - w_{32} + \tilde{w_{32}}]$$

Both have dimensionality of 63
i.e. **D=63**

5 Question

done in python

6 Question

Report outcomes of experiments with both the sklearn.svm.LinearSVC and sklearn.linear model.LogisticRegression methods when used to learn the linear model. In particular, report how various hyperparameters affected training time and test accuracy using tables and/or charts. Report these experiments with both LinearSVC and Logistic Regression methods even if your own submission uses just one of these methods or some totally different linear model learning method (e.g. RidgeClassifier) In particular, you must report how at least 2 of the following affect training time and test accuracy:
(a) changing the loss hyperparameter in LinearSVC (hinge vs squared hinge)
(b) setting C in LinearSVC and LogisticRegression to high/low/medium values

6.1 Answer

Table 1: **a) Training Times and Test Accuracies for Hinge Loss and Squared Hinge Loss in LinearSVC Model**

Loss Type	Training Time (s)		Test Accuracy (%)	
	t_{train}	t_{map}	acc_0	acc_1
Hinge Loss	1.78	0.14	98.3	99.6
Squared Hinge Loss	6.53	0.11	98	99.7

Table 2: **b) Linear SVC:** Training Times, Mapping Times, and Test Accuracies with Different C Values

C Value	Training Time (s) t_{train}	Mapping Time (s) t_{map}	Test Accuracy (%)	
			acc_0	acc_1
Low ($C = 0.01$)	1.04	0.11	97.9	99.5
Medium ($C = 1.0$)	6.22	0.10	98.0	99.7
High ($C = 100.0$)	9.30	0.12	95.5	99.6

Table 3: **b) Logistic SVC:** Training Times, Mapping Times, and Test Accuracies with Different C Values

C Value	Training Time (s) t_{train}	Mapping Time (s) t_{map}	Test Accuracy (%)	
			acc_0	acc_1
Low ($C = 0.01$)	1.03	0.25	98.0	99.3
Medium ($C = 1.0$)	1.10	0.18	98.0	99.7
High ($C = 100.0$)	1.14	0.19	98.08	99.9