CS771: Introduction to Machine Learning
Assignment 1

Avadhi Jindal, Kartik Jhanwar, Khyathi Vagolu, Nikhil Mehta, Sarthak Gothalyan

Problem 1.1

By giving a mathematical derivation, show the exists a way to map the binary digits 0, 1 to signs -1, +1 as say, m : {0, 1} $\rightarrow$ {-1, +1} and another way f : {-1, +1} $\rightarrow$ {0, 1} to map signs to bits (not that m and f need not be inverses of each other) so that for any set of binary digits b_1, b_2, . . . , b_n for any n $\in$ N, we have

$$XOR(b_1, ..., b_n) \; = \; f\left(\prod_{i=1}^{n} m(b_i)\right)$$

Thus, the XOR function is not that scary – it is essentially a product.

Solution

We know that XOR is associative, i.e.,

$$XOR(p, q, r) = XOR(XOR(p, q), r) = XOR(p, XOR(q, r))$$

So, we can group all the 0's in the input together and take their XOR separately, so we have XOR(0, 0,..., 0) = 0.
Let there be b 1's in the input.

- **Case 1**: b is even
  Divide the numbers into groups each having two 1's. XOR of each group will therefore be 0, so now we have some 0's left and we know their XOR will be 0. So, in case of even 1's, we will get the XOR to be 0.

- **Case 2**: b is odd
  For all the $(b-1)$ 1's, divide them into groups of two, XOR of each group will be 0, we will then combine all the 0's that we get and take their XOR, which is 0. So we are left with a 0 and a 1 and their XOR will be 1. So, in case of odd 1's, we will get the XOR to be 1.

So, we have a mapping as follows:

$$m(x) \; = \begin{cases} -1 & x = 1 \\ 1 & x = 0 \end{cases}$$

So we have

$$\prod_{i=1}^{n} m(b_i) = \begin{cases} -1 & \text{odd number of } b_i\text{'s are 1} \\ 1 & \text{otherwise} \end{cases}$$

So we define f as follows

$$f(x) \; = \; \begin{cases} 1 & x = -1 \\ 0 & x = 1 \end{cases}$$

which gives

$$f(\prod_{i=1}^{n} m(b_i)) \;=\; \begin{cases} 1 & \prod_{i=1}^{n} m(b_i) = -1 \\ 0 & \prod_{i=1}^{n} m(b_i) = 1 \end{cases}$$

This results in

$$XOR(b_1, ..., b_n) \;=\; f\left(\prod_{i=1}^{n} m(b_i)\right).$$

Problem 1.2

By giving a mathematical proof, show that for any real numbers (that could be positive, negative, or zero) $r_1, r_2, \ldots, r_n$ for any n $\in$ N, we always have

$$\prod_{i=1}^{n} sign(r_i) = \ sign\left(\prod_{i=1}^{n} r_i\right)$$

Solution

We know that if the signs of two numbers are the same, then the resulting number on multiplying is positive. If the signs are different, then the resulting number on multiplying is negative. And given sign(0) = 0.

- **Case 1**: There exists r_i such that r_i = 0
  then, LHS
  $$\prod_{i=1}^{n} sign(r_i) = 0$$
  and RHS
  $$\prod_{i=1}^{n}(r_i) = 0 \implies \ sign\left(\prod_{i=1}^{n} r_i\right) = 0$$

- **Case 2**: r_i is positive $\forall\ i \in n$
  LHS
  Sign of all numbers is positive and we know that product of positive numbers gives positive result
  $$\prod_{i=1}^{n} sign(r_i) = +ve$$
  RHS
  Product of all positive numbers is positive and so the sign doesn't change
  $$\prod_{i=1}^{n}(r_i) = +ve\ number \implies \ sign\left(\prod_{i=1}^{n} r_i\right) = +ve$$

- **Case 3**: When there are b negative numbers and b is even
  LHS
  Sign of b numbers is negative and the rest $n - b$ numbers are positive. The product of even negatives is positive and the product of positives is also positive so therefore the product of all numbers is positive
  $$\prod_{i=1}^{n} sign(r_i) = +ve$$
  RHS
  Product of two negative numbers is positive and here we have even number of negative numbers, so the sign is positive. The product of $n - b$ positive numbers is also positive, therefore the sign of resultant product is positive
  $$\prod_{i=1}^{n}(r_i) = +ve\ number \implies \ sign\left(\prod_{i=1}^{n} r_i\right) = +ve$$

- **Case 4**: When there are b negative numbers and b is odd
  LHS
  On multiplying odd number of negatives we get negative also the product of positives is positive hence the product of all numbers is negative

$$\prod_{i=1}^{n} sign(r_i) = \text{-}ve$$

  RHS
  Product of odd negative numbers gives a negative number and the product of remaining $n - b$ positive numbers gives a positive number. We know that the product if a positive and negative number is a negative number, therefore the sign of resultant product is negative.

$$\prod_{i=1}^{n}(r_i) = \text{-}ve\ number \implies sign\left(\prod_{i=1}^{n} r_i\right) = \text{-}ve$$

Hence, for any real numbers $r_1, r_2, \ldots, r_n$ for any n $\in$ N, we have

$$\prod_{i=1}^{n} sign(r_i) = sign\left(\prod_{i=1}^{n} r_i\right).$$

Problem 1.3

The above calculation tells us that all we need to get hold of is the following quantity

$$(\tilde{\mathbf{u}}^T\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{v}}^T\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{w}}^T\tilde{\mathbf{x}})$$

Now show that the above can be expressed as a linear model but possibly in a different dimensional space. Show that there exists a dimensionality D such that D depends only on the number of PUFs (in this case 3) and the dimensionality of $\tilde{\mathbf{x}}$ (in this case 8 + 1 = 9) and there exists a way to map 9 dimensional vectors to D dimensional vectors as $\phi : R^9 \rightarrow R^D$ such that for any triple $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}, \tilde{\mathbf{w}})$, there always exists a vector $\mathbf{W} \in R^D$ such that for every $\tilde{\mathbf{x}} \in R^9$, we have $(\tilde{\mathbf{u}}^T\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{v}}^T\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{w}}^T\tilde{\mathbf{x}}) = \mathbf{W}^T\phi(\tilde{\mathbf{x}})$. (10 marks)

Solution

Extending the hint to 3 PUFs, we have,

$$\begin{aligned}(\tilde{\mathbf{u}}^T\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{v}}^T\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{w}}^T\tilde{\mathbf{x}}) &= (\sum_{j=1}^{9} \tilde{u}_j\tilde{x}_j)(\sum_{j=1}^{9} \tilde{v}_j\tilde{x}_j)(\sum_{j=1}^{9} \tilde{w}_j\tilde{x}_j) \\ &= \sum_{i=1}^{9}\sum_{j=1}^{9}\sum_{k=1}^{9} \tilde{u}_i\tilde{v}_j\tilde{w}_k\tilde{x}_i\tilde{x}_j\tilde{x}_k\end{aligned}$$

Thus, if we create a $D = 9^3$, i.e., a 729-dimensional function $\phi : R^9 \rightarrow R^{729}$ that maps

$$\tilde{\mathbf{x}} = (\tilde{x}_1, ..., \tilde{x}_9)$$

to

$$\begin{aligned}\phi(\tilde{\mathbf{x}}) = (&\tilde{x}_1\tilde{x}_1\tilde{x}_1, \tilde{x}_1\tilde{x}_1\tilde{x}_2, ...\tilde{x}_1\tilde{x}_1\tilde{x}_9, \\ &\tilde{x}_1\tilde{x}_2\tilde{x}_1, \tilde{x}_1\tilde{x}_2\tilde{x}_2, ...\tilde{x}_1\tilde{x}_2\tilde{x}_9, \\ &. \\ &. \\ &\tilde{x}_1\tilde{x}_9\tilde{x}_1, \tilde{x}_1\tilde{x}_9\tilde{x}_2, ...\tilde{x}_1\tilde{x}_9\tilde{x}_9, \\ &. \\ &. \\ &. \\ &\tilde{x}_9\tilde{x}_9\tilde{x}_1, \tilde{x}_9\tilde{x}_9\tilde{x}_2, ...\tilde{x}_9\tilde{x}_9\tilde{x}_9)\end{aligned}$$

Now, by taking the column vector $W \in R^{729}$ as

$$\begin{aligned}\mathbf{W} = (&\tilde{u}_1\tilde{v}_1\tilde{w}_1, \tilde{u}_1\tilde{v}_1\tilde{w}_2, ...\tilde{u}_1\tilde{v}_1\tilde{w}_9, \\ &\tilde{u}_1\tilde{v}_2\tilde{w}_1, \tilde{u}_1\tilde{v}_2\tilde{w}_2, ...\tilde{u}_1\tilde{v}_2\tilde{w}_9, \\ &. \\ &. \\ &\tilde{u}_1\tilde{v}_9\tilde{w}_1, \tilde{u}_1\tilde{v}_9\tilde{w}_2, ...\tilde{u}_1\tilde{v}_9\tilde{w}_9, \\ &. \\ &. \\ &. \\ &\tilde{u}_9\tilde{v}_9\tilde{w}_1, \tilde{u}_9\tilde{v}_9\tilde{w}_2, ...\tilde{u}_9\tilde{v}_9\tilde{w}_9)\end{aligned}$$

So we have

$$\mathbf{W}^T\phi(\tilde{\mathbf{x}}) = \sum_{i=1}^{9}\sum_{j=1}^{9}\sum_{k=1}^{9} \tilde{u}_i\tilde{v}_j\tilde{w}_k\tilde{x}_i\tilde{x}_j\tilde{x}_k$$

Hence,

$$(\tilde{\mathbf{u}}^T\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{v}}^T\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{w}}^T\tilde{\mathbf{x}}) = \mathbf{W}^T\phi(\tilde{\mathbf{x}}).$$

Problem 1.5
We used **Mini Batch gradient descent** which uses three hyper-parameters namely:

1. Step Length : $\frac{.5}{t}$ where t is number of iterations.
2. Batch Size : 10
3. Tuning/Penalty Parameter : 1

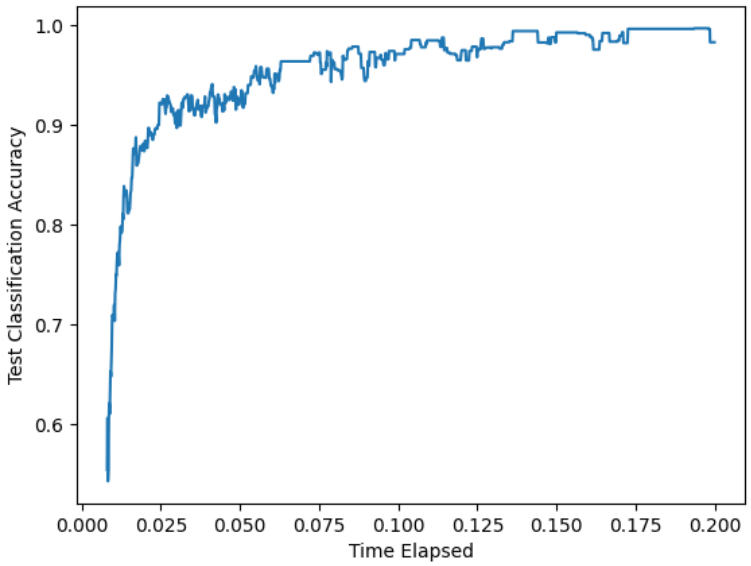
Batch Size: Batch Size was varied from 1 to Size of Data-set and fastest convergence was observed near small batch size of around 10 to 20. We also tried to have batch size in multiples of 2 as generally used on GPU's but misclassification rate slightly increased at 8 or 16. Both 10 and 20 were giving best classification at .2 Seconds. 10 was chosen as Batch size randomly in the code with equal preference to 20.

Step Length: Numerator was varied from very small values such as 0.0001 to nearly 100 and denominator was used in powers of t (number of iterations) such as t, $t^{1/2}$, $t^{1/3}$. The best rate comes out to be near $0.5/t$. Choosing this value also satisfies that step length converges to 0 as number of iteration increases and sum of step length diverges as it is a harmonic series.

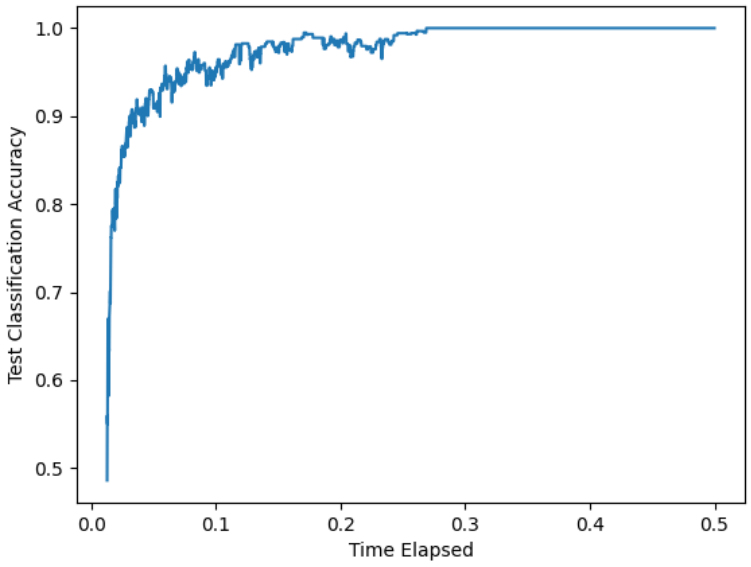
Penalty Parameter: This was varied from 1 to 10 but it didn't had much impact on misclassification rate around these values, so it was randomly chosen to be 1

Problem 1.6
Plots of Classification Accuracy vs Time Elapsed

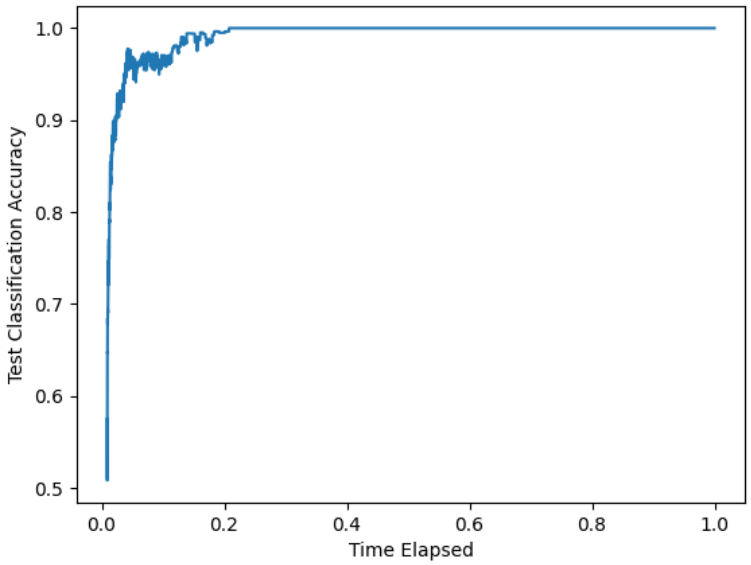
Time = .2 Sec



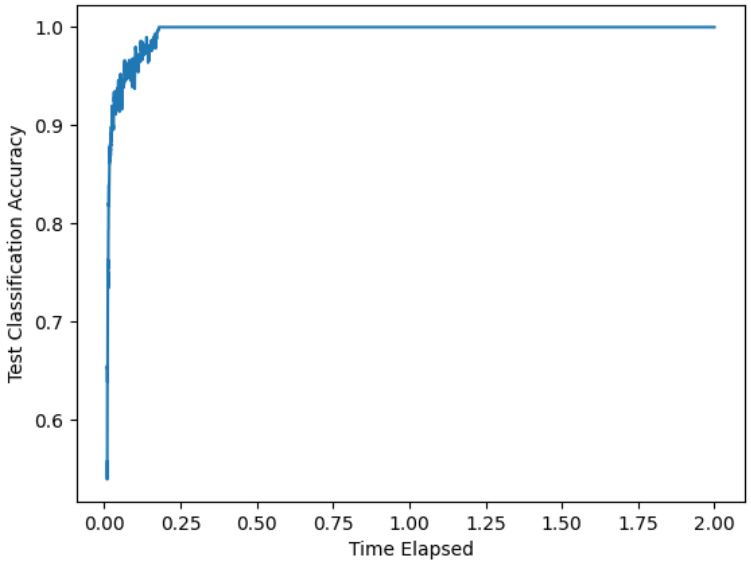
Time = .5 Sec



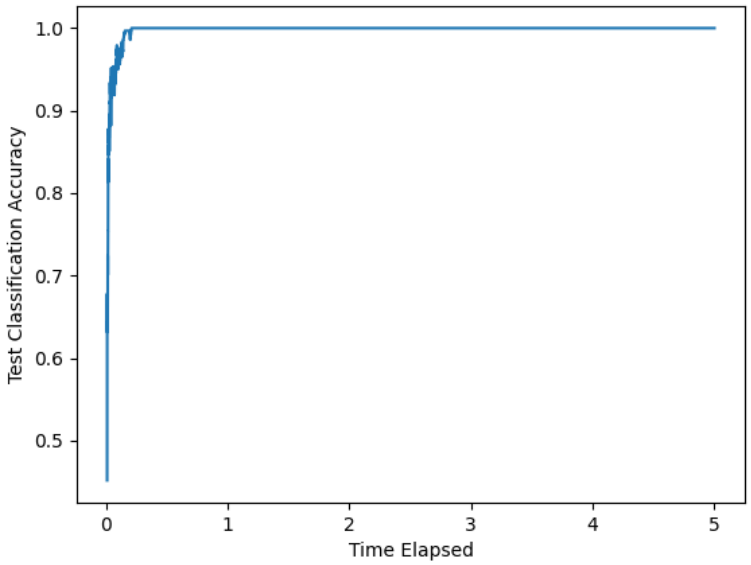
Time = 1 Sec



Time = 2 Sec



Time = 5 Sec


1.0
0.9
0.8
0.7
0.6
0.5
Test Classification Accuracy
0
1
2
3
4
5
Time Elapsed