Core Concepts

Recap: packing

Core concept(s):

- Ciphertext packing
- SIMD

Parameters

- Weights: [1,2,3,4]
- Number of slots = 32
- Row-size: 8

Packing the weights:

1) padded = [1,2,3,4, 0,0,0,0]
2) Repeat: [padded,padded,padded,padded]

Recap: multiplication

Core concept(s):

- Ciphertext packing
- SIMD

Parameters

- Weights: [1,2,3,4]
- Number of slots = 32
- Row-size: 8

Variables:

1) padded = [1,2,3,4, 0,0,0,0]
2) Repeat: [padded,padded,padded,padded]

repeat * repeat

= [padded**2, padded**2, padded**2, padded**2]

[1,4,9,16, 0,0,0,0 1,4,9,16, 0,0,0,0

…1,4,9,16, 0,0,0,0 1,4,9,16, 0,0,0,0]

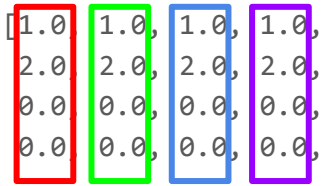
Matrix packing orientation

```
in_mat = [1.0, 1.0, 1.0, 1.0,
          2.0, 2.0, 2.0, 2.0]
```

Horizontal Packing

```
[1.0, 1.0, 1.0, 1.0, 0, 0, 0, 0, 0
 2.0, 2.0, 2.0, 2.0, 0, 0, 0, 0, 0]
```

Vertical Packing



Summation

```
in_mat = [1.0, 1.0, 1.0, 1.0,
          2.0, 2.0, 2.0, 2.0]

Row_size = 4

Batch_size = 8
```

```
EvalSum = 12.0
= [12.0, 12.0, 12.0, 12.0, 12.0, 12.0, 12.0, 12.0]
```

```
EvalSumRows
[1.0, 1.0, 1.0, 1.0,
 2.0, 2.0, 2.0, 2.0]
= [3.0, 3.0, 3.0, 3.0, 2.0, 2.0, 2.0, 2.0]
```

```
EvalSumCols
[1.0, 1.0, 1.0, 1.0,          [4.0,
 2.0, 2.0, 2.0, 2.0]           8.0]
= [4.0,4.0,4.0,4.0, 8.0, 8.0,8.0,8.0]
```

dot-product

Ciphertext-sum o ciphertext-multiplication

Parameters:

- Weights: [1,2]
- Data = [0.1, 0.1,
  0.3, 0.5]
- batch_size = 8
- Row-size: 4

Variables

- packed_w = [1,2, 0,0
  1,2, 0,0]
- Packed_data = [0.1, 0.1, 0, 0,
  0.3, 0.5, 0, 0]

Mult: rep_w * packed_data

- [0.1, 0.2, 0, 0,
  0.3, 1.0, 0,0
  ]

!! Sum (EvalSumCols)!!

- [0.3, 0.3, 0.3, 0.3,
  1.3, 1.3, 1.3, 1.3
  ]

Sum (EvalSumRows)

- [0.4, 1.2, 0,0,
  0.4, 1.2, 0,0
  ]

Recap: simple optimizations

Linear regression:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$\theta_j := \theta_j - \alpha \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^i$$

In-the-clear optimizations:

- doing -X.T
- Scaling directly by the α

Bootstrapping

Expensive but amenable to more use-cases

Pack multiple ciphertexts into a single one, then extract

A = [1, 2, 3, 4]

B = [10, 20, 30, 40]

Multi_pack = [1, 10, 2, 20, 3, 30, 4, 40]

Mask_a = ?

Mask_b = ?

Some tips for working with FHE problems

:

1. start with a small-ish ring dimension
2. turn off the security setting (via `HEStd_NotSet`)
3. create a reference numpy implementation
4. Try to do as much as possible in plaintext-space before finally working with ciphertexts
5. ciphertext refreshing speeds up iteration, so start with that for prototyping then move to bootstrapping