CS670: Cryptographic Techniques for Privacy Preservation Assignment 1

Shubham Rawat (251110070)
Department of Computer Science and Engineering, IITK

Secret Sharing Implementation

The Matrix U and V are split into additive shares between two parties S_0 and S_1:

$$U = U_0 + U_1, \quad V = V_0 + V_1$$

Each party stores its share locally and not allowed to share actual share with each other.
For implementing update on user matrix, we first need to 2 thing user index and item index.
User index is public but item index is not public.
For item index we use standard basis vector of size : number of items, and secret shared it between two parties.
Then we will compute MPC dot product of u_i, and v_j. Once $\langle u_i, v_j \rangle$ is computed: then we will compute :

$$\delta = 1 - \langle u_i, v_j \rangle$$

Here 1 is also secret shared between parties.
After that we will do MPC multiplication of vector v_j and δ

The finally each party can update user vector locally securely as:

$$u_i \leftarrow u_i + v_j \cdot \delta$$

For the multiplication either vector dot vector or vector multiplied by scalar or vector multiply by matrix we used Beaver Triplet method for obtaing shares of MPC multiplication.

MPC Inner Product and Updates

Secure Dot Product

To compute $\langle u_i, v_j \rangle$ securely:

1. A trusted dealer sends shares of Beaver triplet to both parties.
2. Using triplet share each party computed blinded shares and sent to each other.
3. after local computation each party gets their share of multiplication.

Each party updates its own shares of u_i and then re-shares them if necessary. The operations are performed using only the local shares and MPC communication.

Communication and Efficiency

- Trusted dealer sends Beaver triplet shares, share of 1, shares of standard vector to both parties.
- Party0 and Party1 sends blinds to each other
- Blind 1 for computing v_j
- Blind 2 for computing $\langle u_i, v_j \rangle$
- Blind 3 for computing $\cdot\delta$
- C++ coroutines (`boost::asio`) are used to handle party communication efficiently.

Code Structure

- `shares.h`: Defines the structure of secret shares and helper functions for randomization.
- `mpcOperations.h/cpp`: Implements MPC functions including `MPC_DOTPRODUCT` and vector updates.
- `gen_queries.cpp`: Generates random queries and secret-shared vectors.
- `main.cpp`: Reads input, performs MPC updates, and prints debug output.
- party0 and party1:Each party handle local computation and comunicate with dealer and with each other.

Build and Run Commands

- Build the project:

```
make all
```

- Generate queries and shares:

```
make run_generate
```

- Run the MPC parties in three separate terminals:

```
./dealer.out
./party0.out
./party1.out
```

Conclusion

This assignment demonstrates secure multiparty computation using additive secret sharing to perform recommendation system updates without leaking user or item data. The implementation ensures correctness, security, and efficiency in communication.