Abed Elrahman abo hussien
Michael Rodel

PPEM Progression Log

13.3:

- Created the basic algorithm but not yet implemented it as a class. This class should allow us to manipulate each component of the algorithm independently.

8.4.23:

- Created the skeleton for the EM algorithm. Now we need to work on the specifics of this algorithm. The E-step and M-step should be altered using a new class for each algorithm, PPEM and EM, so that we can compare them.

7.5.23:

- Initial EM algorithm works on all dimensions with plots describing the final result.

14.5.23:

- Full fast EM algorithm works with log-likelihood and responsibilities. Added a feature representing the plot of the algorithm throughout its lifespan.
- Added randomly created points with multiple dimensions. Note that the plot works only in 2 dimensions, but the algorithm itself works in all dimensions.

17.5.23:

- Corrected the algorithm of full fast EM, which wasn't working correctly and took too many iterations for convergence.

25.5.23:

- Worked on the idea of using k-means to approximate each parameter received from the client (miu and sigma) in the server to cluster the correct parameters together, then calculate the mean of them as the centroid.

30.5.23:

- Realized that the previous idea can alter our results. Its implementation on encrypted data is also challenging and time-consuming. It may not be a valid solution.
- Discussed that our approach might not be good enough unless we assume an honest but curious adversary on the server's side.

6.6.23:

- We came up with the following approach:

Working on the main approach

- Client-server-based algorithm:
    1. Server-side:
        - Receives the data and the standard deviations for each client.
        - Uses k-means to approximate the parameters of the distributions.
        - Applies EM algorithm using those parameters while the parameters are encrypted.
    2. Each client:
        - Uses BFV encryption to encrypt its data.
        - Applies EM on its data set.
        - Sends the results to the server using a secure SSL connection.

8.6.23:

- Updates on the algorithm itself:
    - The client uses k-means to initialize the data, which might send part of the information but not all of it.
    - The server might just use a part of the M-step from each client.
    - We are exploring threshold encryption.

10.6.23 - 21.6.23:

- Our approach was flawed and not a good idea, as k-means will be biased in certain cases.
- Researching different Privacy-Preserving k-means and EM algorithms to think of a new concept.

Abed Elrahman abo hussien
Michael Rodel

1.7.23:

- Currently thinking of solutions. We didn't want to create the "standard" algorithm, but this might have been a mistake.
- Considering threshold encryption but unsure if it's fully secure.
- Looking for an option to split the work between the server and the client without compromising too much security or performance.

3.7.23:

- Based on this approach: Link to Approach
- We are now basing our approach on a peer-to-peer concept, inspired by "A Local Scalable Distributed Expectation Maximization Algorithm for Large Peer-to-Peer Networks."
- In our implementation, each peer is a client, and the data sent to the server is encrypted via homomorphic encryption.
- The E-step is performed in each "peer"'s environment, while the M-step is split between the server and the client.

$$\overline{\mathcal{L}}(\Theta|\mathcal{G}) = \frac{\sum_{i=1}^{p}\sum_{a=1}^{m_i}\log\left(\sum_{s=1}^{k}\pi_s\mathcal{N}(\overrightarrow{x_{i,a}};\overrightarrow{\mu_s},\mathbf{C}_s)\right)}{\sum_{i=1}^{p} m_i}$$

E-step:

$$q_{i,s,a} = \frac{\pi_s\mathcal{N}\left(\overrightarrow{x_{i,a}};\overrightarrow{\mu_s},\mathbf{C}_s\right)}{\sum_{r=1}^{k}\pi_r\mathcal{N}\left(\overrightarrow{x_{i,a}};\overrightarrow{\mu_r},\mathbf{C}_r\right)}$$

M-step:

$$\begin{aligned}
\pi_s &= \frac{\sum_{i=1}^{p}\sum_{a=1}^{m_i} q_{i,s,a}}{\sum_{i=1}^{p} m_i} \\
\overrightarrow{\mu_s} &= \frac{\sum_{i=1}^{p}\sum_{a=1}^{m_i} q_{i,s,a}\overrightarrow{x_{i,a}}}{\sum_{i=1}^{p}\sum_{a=1}^{m_i} q_{i,s,a}} \\
\mathbf{C}_s &= \frac{\sum_{i=1}^{p}\sum_{a=1}^{m_i} q_{i,s,a}(\overrightarrow{x_{i,a}}-\overrightarrow{\mu}_s)(\overrightarrow{x_{i,a}}-\overrightarrow{\mu}_s)^{\mathrm{T}}}{\sum_{i=1}^{p}\sum_{a=1}^{m_i} q_{i,s,a}}
\end{aligned}$$

- Security is guaranteed if the server is an honest but curious adversary. Additional security measures may include Elliptic Curve Digital Signature Algorithm or Edwards-curve Digital Signature Algorithm to verify the identities of the server and clients.

4.7.23:

- Implemented the algorithm without encryption. Need to check if it works and then implement the encryption scheme.
- The full skeleton of the algorithm is ready.

5.7.23:

- Added more documentation to the code and made modifications to the Partial EM and Server (Full EM) algorithms.
- The full algorithm should be completed soon.
- Started creating the finished version of the algorithm with proofs in the paper we are working on.

6.7.23:

- Partial EM and full EM algorithms are complete.
- Conducted tests checking different parameters: number of clusters, number of parties, length of input. Still haven't experimented with dimensions.

8.7.23:

- Discovered another algorithm called the federated EM algorithm.
- Exploring ways to improve the efficiency of the algorithm, such as clustering the parameters of the EM algorithm to make the M-step more efficient.
- Exploring threshold encryption.

10.7.23:

- Realized that our assumption about responsibilities in the algorithm was incorrect.
- Experiencing issues with calculating new covariances according to new means in the M-step.

14.7.23:

- Improved the convergence of the EM algorithm by initializing each pi, means, and covariances of each client using a local EM algorithm.

- Considered adding a k-means method to cluster the "clusters from each client" and send the mean back to each client.

25.7.23:

- Scheduled an appointment with our mentor to review our approach and identify areas for improvement.

8.8.23:

- Started writing the article and finished section 1, including the introduction, in LaTeX using Overleaf.

15.8.23:

- Finished the second part of the article, explaining the general federated EM algorithm and providing essential background information.

22.8.23:

- Further elaborated on our approach and explained it in the article.

27.8.23:

- Worked on the algorithm itself and provided pseudocode.
- Conducted background research on the CKKS approach and ways to prove our approach's privacy.

5.9.23:

- Found articles supporting the lemma that two different cipher codes are indistinguishable from each other.
- Added some citations to the article.

10.9.23:

- Finished writing the privacy and correctness sections of the article based on the provided information.

Abed Elrahman abo hussien
Michael Rodel

12.9.23:

- Working on obtaining results for the Parkinson's dataset to include in the article.

13.9.23:

- Preparing for possible changes in the latest developments.

7.8.23:

- Started working on the LaTeX document and finished the first section.

14.8.23:

- Introduced the EM algorithm in the paper.
- Reshaped the entire document for readability.

20.8.23:

- Expanded on the federated EM in the paper, including the GMM equations, federated EM equations and steps, and federated EM over networks introduction in Overleaf.
- Addressed the issue of privacy leakage.
- Added privacy proof for our approach.

27.8.23:

- Expanded on the federated EM and explained the initial changes to address the issue of privacy leakage.
- Explained CKKS encryption, how it's handled, and why it's a good implementation.
- Provided pseudocode for the proposed approach.
- Worked on the presentation of the proposed approach and added the basic Federated EM.

3.9.23:

- Reviewed and added citations to the article.

- Discussed the privacy of our algorithm.
- Provided a more detailed explanation of our approach.
- Added our implementation to the presentation.

10.9.23:

- Added the Parkinson's dataset to the GitHub repository.
- Explained why we chose this dataset for testing our proposal.
- Used PCA to reduce dimensionality and created results to include in the article.

13.9.23:

- Completed more in-depth information in the article, focusing on correcting mistakes.
- Added scripts for testing and analyzing results.
- Made adjustments for parallelism to simulate real-world client-server interactions in the PPEM variant.
- Created methods for comparing results of both algorithms with the same parameters.
- Renamed files for better representation and organization.

18.9.23:

- Waiting for both algorithms to finish their runs to collect all results for inclusion in the article.

21.9.23:

- Completed writing the full article and prepared to submit all materials. Some work is needed on the presentation.

26.9.23:

- We edited the Article and finished the presentation.
- Reviewed all the code ,and added all the folders to github.