Project in Secure Machine Learning

203.3762, 203.4762

Fall 2024-2025 סמסטר א' תשפ"ה

Lecturer: Dr. Adi Akavia (CS, UoH), akavia@cs.haifa.ac.il

TA: Mr. Ramy Massalha ramymassalha@gmail.com

Prerequisites: Data Structure 203.2310, Programming 214.1960

First meeting: Tue, Nov 12th, 12:15-13:45 on Zoom – the Zoom link is available via the course page in Moodle

Reception hour: Tue 12:15, at Amir bld. 112 or zoom (depending on missile status).

Emails: In all emails to Lecturer/TA, the subject line must start with "**Lab24fall"**

Class overview: The intersection of cryptography and machine learning is a thriving research area with ample intriguing research problems of major significance. Some examples follow. *Privacy preserving machine learning* (PPML) aims at protecting sensitive information while the utility of machine learning models, such as training while protecting the confidentiality and authenticity of the training data, or classification/prediction while protecting model and/or data. Other examples include *adversarial examples* and *backdoors*, spanning from attacks to means of protection; *explainability* of AI decision to allow human auditing hedging their potential risks; *fairness* and multi-calibration considerations; and *alignment* of artificial intelligence with human goals. In your project you will dive into one topic in the intersection of Cryptography and Machine Learning, getting acquainted with current research and making your own contribution.

Submission deadline: January 28th, 2025 (on Moodle).

Grace period: late submissions are accepted until February 9th, with a penalty of 1 point in the grade for each late day.

Important Notes:

1. Strict submission deadline: Late submission will not* be accepted.
   *Except for unique circumstances of: MILUIM / maternity leave / extended illness – proper documentations and advance approval by lecturer are required.

2. Appeals imply a re-examination of the work that could decrease grades.

3. Submission is in the designated submission box in Moodle only.

4. Weekly/biweekly progress meetings are encouraged.

5. Student submitting excellent projects are invited to discuss with me the possibility of extending their project to a research project (open to both BSc, MSc and PhD students).

6. The collaboration and grading policy are complicated; see details below.

Grading and collaboration policy: Students are graded primarily on their individual work. Collaboration in a group of size at most 2 students is allowed, as follows:

1. **Option 1 – A Solo Project** (group size: 1): Students are encouraged to complete the full project (Parts A+B+C below) on their own, without collaboration. Grading accounts for each part (A, B, C) as 33.3% of total grade.
2. **Option 2 – A Joint Project** (group size: 2): You are allowed to join forces in pairs, where one student it in charge of Part A of the project and the other student on Part B, and both students are jointly in charge of part C. You report must clearly indicate, when specifying the authors, which student is in charge of which part. The grade will be computed as follows:
   - 67% of your grade is the grade for your part
   - 33% of your grade is for the grade of the integrated project (Part C)
3. **Option 3 – A Solo Project, Submitting A Single Part (Only A or B, but not both):** You are allowed to submit a single part: Part A or Part B, and not both. Important notes on this option follow.

- You must clearly indicate in the title of your report, that your project is for one part and which (A or B).
- Your final grade will be the grade for the submission part, multiplied by 0.8. I.e., the final grade is at most 80.
- Option 1 vs. 3: your grade is likely to be higher in Option 3 than in Option 1 in case you accomplish well only one part. For example, a 100% grade on a single part and 0% on the other results, when taking Option 3, in final grade 80, compared with final grade 67 when taking Option 1. This is designed to encourage you to submit only work that is well done.
- Switching from options 1 or 2 to 3: If you planned to take option 1 or 2, but it doesn't work out (e.g., you did not manage to complete the work required for option 1, or the collaboration with your partner went sour in option 2), you are allowed to switch to option 3 when submitting your final project.

Submission Material: You should submit in the Moodle submission box a zip file named <your full name>, that includes the following materials:

1. **Scientific report** (written in **LaTex**. You are required to submit **.pdf** file as well as **.tex**, **.bib** files and any other file required for compiling your report to produce the .pdf file.) – **This is your main product**; the report should be comprehensive and contain everything your grader should know about your work. See detailed guidelines in Appendix A.

2. **PPTX presentation:** file presenting the background, research goal, results, conclusions, bibliographic list for any prior work cited in the presentation.

3. **All source files:** code files, data, excel/data files, images.

4. **Test/demo files** (source code) the can be executed to reproduce all your results.

5. **README** file explaining how to execute your code and test/demo files.

6. If invited by the lecturer: **presenting your project (oral presentation)** in front of the lecturer or an entire class.

A detailed project description follows. A table of contents is provided for your convenience.

Table of Contents

The Project: Privacy Preserving Biometric template protection using Homomorphic Encryption.

Biometric identification relies on biological characteristics (e.g. fingerprints, iris, face) of individuals to identify them among several identities whose characteristics are stored in a dataset. Biometric identification has been used for decades in various applications such as law enforcement, border control and numerous commercial applications. Still, the adoption of biometric identification in a wide variety of applications comes with an inherent privacy risk, as potential security breaches may leak private information of the individuals whose biometric data is stored.

A promising method to tackle this privacy risk is to use *Fully Homomorphic Encryption (FHE),* which is a type of encryption that supports computations on encrypted data. A user who has confidential data may encrypt the data using FHE and deploy it to a cloud service for further computations. The entire computation process in the cloud is applied while the data is encrypted. Thus, FHE can be used to support biometric identification applications without compromising the privacy of the individuals whose data is stored in the cloud.

However, this entails several challenges. In your project you will address one such challenge – the precision challenge – as detailed next.

A commonly used FHE scheme, called HEAAN [CKKS'17], adds noise to the data it works on. For example, computing the product 5*6 over FHE might decrypt to the result 29.999999999 instead of 30. When FHE is used to implement privacy preserving biometric identification, this noise may ruin the quality of the results.

Your goal in this project is composed of three parts:

- **Part A:** research the effect of limited precision (or, noise) on the correctness of biometric identification algorithm.
- **Part B:** implement privacy preserving similarity metric computation.
- **Part C:** integrate parts A and B to implement privacy preserving biometric identification.

Details follow.

Part A – Biometric Identification with Limited Precision, over cleartext data

The goal of this part is to research the effect of limited precision (equivalently, noisy computation) on the correctness of biometric identification algorithm.

The suggested project requires the students to find a public biometric identification algorithm, including a public **DB** and a public **model** mapping samples to vectors (**vector embedding**) with state-of-the-art performance, run it using low precision computations and check how the limited precision affects the performance of the biometric identification. If the model performance under a low accuracy environment is the same or close to the accuracy of the original model, we can conclude this model is a good option to be implemented over FHE.

The steps of this objective may be summarized as follows:

1. **Find a public biometric identification DB and model, download it and test the accuracy of the model.** See Appendix B for some initial links and examples.
   Requirements (if not achievable, discuss with Lecturer/TA):
   a. DB > 10K individuals
   b. >99 individuals with 2 distinct samples, where 1 sample is used for testing and not exposed while training the model.
   c. High accuracy, preferable >95%.
   d. Similarity score is computed via a low degree polynomial, such as Euclidean, Hamming, Cosine.
   e. A research paper recently published in reputable venues present the DB, model and results.
2. Re-run the model using computations with **limited precision**. Test the accuracy of the model when working on a p-bit precision environment, for varying values of p.
3. Prepare a **report** (following the guidelines in Appendix A) introducing the used model and DB and showing graphs of the performance metrics as a function of p.

Remark: Materials produced in the projects may be made available for use in my lab (e.g., by students working on follow-up projects); participating in this course implies consent.

Part B – Similarity metric computation, over encrypted vectors

The goal of this part is to implement a privacy preserving version for computing the similarity score that you implement in Part A.

1. Generate two **vectors** of the same dimension and precision as the vectors generated by your model from part A (i.e., the vector embedding).

2. Generate **keys** for the CKKS FHE scheme as implemented in one of the following libraries: pyhe*layers* or HEaaN or *OpenFHE* or *Microsoft SEAL*.

3. **Encrypt** both vectors from Step 1 under the public key generated in Step 2. Note: you may like to use SIMD/PACKING to pack as many values as possible in each ciphertext.

4. **Homomorphically compute the similarity** score between the two encrypted vectors.

Performance: Repeat steps 1-4 100 times (if too slow, reduce to 20 times), and measure:

- **Accuracy:** measure the absolute difference between computing similarity score over cleartext vs. encrypted vectors, in each repetition; and report the average, std and max of these differences.
- **Runtime:** Measure the runtime of each step (1-4), , in each repetition; and report the average, std and max of these runtimes.

Prepare a **report** (following the guidelines in Appendix A) reporting your privacy preserving similarity computation.

Part C – Biometric Identification, over encrypted vectors

The goal of this part is to implement a privacy preserving version of the biometric identification algorithm that you implement in Part A while using the homomorphic computation of the similarity score as implemented in Part B. The flow of the privacy preserving version, including steps for measuring accuracy and runtime and communication (written in blue font), is as follows:

5. **Compute vector embedding:** Raw images (templates and test samples) are passed through the model from Part A to produce the corresponding vector embedding.

6. **For accuracy testing purposes: compute similarity score over cleartext:** for each sample in the test set, compute (a) its similarity score with respect to all templates, and (b) the top-10 scoring templates in descending order. Save these values in a data file (.cvs):

    a. A file with a row for each sample, and a column for each template, specifying the similarity score for each sample vs. template pair. Name this file: ***scores.csv***.

    b. A file with a row for each sample, and 10 columns specifying for each sample the indices of its top-10 templates in decreasing order. Name this file: ***top10_dec.csv***.

7. **Encrypt all vectors:** Generate keys for the CKKS FHE scheme as implemented in *pyhelayers* or HEaaN or *OpenFHE* or *Microsoft SEAL* library. Encrypt all vectors (templates and samples), while using SIMD/PACKING* to pack as many values as possible in each ciphertext.
   *Note: Choosing the best packing may be complicated, you are encouraged to consult with the TA.

8. **Homomorphically compute similarity scores over encrypted values:** for each pair of an encrypted sample from the test set and an encrypted template from the template DB. The outputs of this step should be *nxm* encrypted score values, where n is the number of samples and m the number of templates.

9. **For accuracy testing purposes do:**

a. Decrypt the similarity scores computed in Step 8, and save these nxm cleartext values in a data file (.cvs) with a row for each sample, and a column for each template. Name this file: ***scores_dec.csv***.

b. For each sample (i.e., each row in the file *scores_dec.csv*), compute the template with best similarity score (also, 5-top and 10-top scores). Save these values in a data file (.cvs) with a row for each sample, and 10 columns specifying for each sample the indices of its top-10 templates in decreasing order. Name this file: ***top10_dec.csv***.

10. **Measure accuracy** of your homomorphic biometric identification vs. the biometric identification over cleartexts, and for all precision values:

    a. Compare the files score_dec.csv vs. score.csv. Measure average, std, max, min absolute distance between each (i,j) value in *score* vs. *score_dec*.

    b. Compare the files top10_dec.csv vs. top10.csv. For each column i=0,1, …, 9, measure the percentage of test samples j for which the i-th most similar template to sample j is the same in *top10.csv* and *top10_dec.csv*.

11. **Measure runtime** of each step 1-5 (separately). Report your accuracy and runtime findings.

12. **Measure the size** of the ciphertexts holding the encrypted similarity scores.

Prepare a **report** (following the guidelines in Appendix A) reporting your privacy preserving similarity biometric identification computation. This is a final report that must include ALL material you would like to be graded on, including an integrated presentation of your work in A, B and C.

Recommended Timeline

Note: The provided timeline is for a solo project (Option 1). For Options 2: shift each date in Parts B and C to be 4 weeks earlier. For Option 3: you may take longer for each step in your part. **Note:** Only the final submission (in the Final Submission Moodle box) will be checked and graded.

Timeline for Part A

- **By week 2 (Nov 19th meeting):**

  Do: Choose the paper, DB and model of your choice.

  Submit (in *Part A kickoff* submission box in Moodle): A paragraph specifying your choice, include full details and links for 1) paper, 2) DB and 3) model.

- **By week 3 (Nov 26th meeting):**

  Do: **Implement** the biometric identification (over cleartext data) using **full precision** (using standard datatypes, e.g., float). Execute the model and identification on the DB, to produce a score vector, and measure accuracy.

  Summarize your results in a short report specifying what you did and your results.

- **By week 4 (Dec 3rd meeting):**

  Do: Implement the biometric identification with **limited precision**, and test accuracy on various precision parameters. I.e., the process should be as follows:

  1) Raw images (templates and test samples) are passed through the model, as before, to produce the vector embedding;
  2) For each vector produced in step 1, truncate the low order digits of its entries to produce a new vector with a lower precision. The number of digits to be truncated should be a parameter that you control.
  3) Test the accuracy of your biometric identification on these low precision vectors, with various precision values (e.g., 1-10 digits after the decimal point, when values are normalized to (0,1) to (-1,1)).

- **By week 5 (Dec 10th meeting):**

  Do: Polish your report and code to maintain academic writing standards. Prepare a PPTX presentation. **Present your work in the weekly lab meeting (recommended).**

  Submit (in *Part A* submission box in Moodle): code, PPTX, and report.

Timeline for Part B (in a solo project):

- **By week 7 (Dec 17th meeting):**

  Do: Educate yourself on FHE focusing on the CKKS scheme (e.g., read a primer/tutorial on FHE and on CKKS). Make sure you are familiar with the following components: key generation, encoding, decoding, encrypting, decrypting, parameters setting, homomorphic addition, homomorphic multiplication, packing multiple message in a ciphertext, and slot rotations.

  Implement the knowledge you have acquired by downloading and installing an **FHE library** supporting **CKKS** (also known as **HEaaN**) encryption scheme (the library: **pyhelayers** or **OpenFHE** or **HEaaN** or **Microsoft SEAL**), and executing small examples to check you can execute the library to run simple computations over encrypted data.

- **By week 9 (Dec 31th meeting):**

  Do: Implement **homomorphic computation of the similarity metric** in the paper of your choice (typically, Euclidean, Hamming or Cosine distance). The output should be the encrypted vector of similarity scores. **Run** the implemented similarity score on synthetics vectors with the same dimension as in your biometric identification system. **Test** that decrypting your output produces the same results as if computing over cleartext. **Measure** runtime. Write a short report describing your algorithm, implementation and results.

  Submit (in *Part B* submission box in Moodle): code and report.

  Note: Setting CKKS parameters may be challenging. Moreover, computing over packed ciphertext is likely to require using "rotate-and-sum" algorithm for summing all slots of a ciphertext using log(slotCount) rotations. You are encouraged to consult the Lecturer/TA/classmates.

Timeline for Part C (in a solo project):

- **By week 10 (Jan 7th meeting)**:

  Do: Integrate the homomorphic similarity score computation from Part B with the vector embedded (model) from part A.

- **By week 11 (Jan 14th meeting):**

  Do: Measure accuracy and runtime performance on your DB. Summarize results in graphs and table and a short text.

- **By week 12 (Jan 28th meeting):**

  Submit (in ***Final Project*** submission box in Moodle): your final project including all required submission materials and in the appropriate formatting.

References (a kickoff list, non-comprehensive)

Notes: This bibliographic list of is an initial resource to help you get started; you are expected to find a more comprehensive list of related material online yourself. Google scholar is a recommended when searching research papers.

FHE

[Halevi'17] Fully homeomorphic encryption (FHE) survey (not covering [CKKS'17]): Halevi, Shai. "Homomorphic encryption." *Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich*. Cham: Springer International Publishing, 2017. 219-276. https://shaih.github.io/pubs/he-chapter.pdf

CKKS

The FHE scheme of Cheon et al 2017 [CKKS'17] was presented in the following research paper:

[CKKS'17] The research paper: Cheon, Jung Hee, et al. "Homomorphic encryption for arithmetic of approximate numbers." *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*. Springer International Publishing, 2017.

The research paper is hard to follows for novices in the field, a better place to start can be a tutorial, like this blog post (parts 1-4):

[CKKS'17] A blog entry: "CKKS Explained" https://blog.openmined.org/ckks-explained-part-1-simple-encoding-and-decoding

Libraries supporting CKKS (also known as HEaaN)

- **pyhelayers** https://ibm.github.io/helayers/user/installation.html
  A demo file, demonstrates how to use pyhelayers to compute matrix-vector multiplication over FHE, is available in the Moodle page.
- **HEaaN** https://heaan.it
- **OpenFHE** https://www.openfhe.org/

- **Microsoft SEAL** https://github.com/microsoft/SEAL (note, this library is no longer supported by its developers; making the above two are more highly recommended).

Biometrics

There are many tutorials and research papers on biometrics available online. Following are links to few examples:

- https://www.tutorialspoint.com/biometrics/index.htm
- https://biometrics.cse.msu.edu/Presentations/AnilJain_Biometrics_CMU04.pdf

Privacy preserving biometric using FHE

There are many papers of privacy preserving biometrics using homomorphic encryption. The following survey reviews many such papers that. You are welcome to pick one of the reviewed papers and try to implement it. Note: you must ensure that the paper you pick is of sufficiently high quality; consult with Lecturer*/TA when in doubt. You are also welcomed to seek other high-quality papers as well.

*Discussions with Lecturer take place only in the weekly reception hour; not via email.

[YWCTL'23] Yang, W., Wang, S., Cui, H., Tang, Z., & Li, Y. (2023). A review of homomorphic encryption for privacy-preserving biometrics. *Sensors*, *23*(7), 3566. https://pmc.ncbi.nlm.nih.gov/articles/PMC10098691/pdf/sensors-23-03566.pdf

Privacy preserving biometric using FHE – beyond the scope of our class

Note that in this class we focus on privacy preserving biometrics using FHE; however there are other privacy preserving methods available, here is one example:

[BM'24] Marina Blanton and Dennis Murphy, Privacy Preserving Biometric Authentication for Fingerprints and Beyond, Cryptology ePrint Archive, Paper 2024/525, 2024, https://eprint.iacr.org/2024/525

Appendix A. Report Structure Guidelines.

Your final report should have the following structure:

1. **Title:** write here the title of your project

2. **Author name(s):** write here the full names and IDs of the submitting student(s)

3. **Supervised by:** write here "This project was supervised by Prof. Adi Akavia, Fall 2024"

4. **Abstract**
   short (up to 250 words) summarization the problem you address and your key results

5. **Introduction**
   Here you should write:

   a. The most related **background** leading to the research/project question/goal (1 paragraph)

   b. Your **research goal** (1 paragraph)

   c. Your **results** (a subsection clearly describing your results emphasizing both the qualitative and the quantitative contribution)

   d. **Related work** (a subsection clearly describing the related works, with proper citations using \cite{…}, describing for each work what they achieve and the pros and const with respect to your results. Here again you should address both qualitative and quantitative aspects)

6. **Technical Background / Preliminaries**
   The report must be self-contained; this is a technical section where you can include any knowledge the reader should know, while citing the relevant prior works.>

7. **Results:** here you should describe your results in details.
   Start with an opening paragraph saying what is included be in each subsection of this section (typically, 1 sentence for each subsection).

Then write the following subsections (minor adaptations might be needed, depending on your project, you are advised to approach the lecturer where in doubt):

a. **The Protocol:** describe the protocol that you implement (unless already specified in a subsection of the Technical Background section. If so, add a reference to it \ref{..} with an appropriate subsection label there).

b. **System description:** describe the proof of concept system that you have implement. Include:

    i. **A high-level verbal description of your system.**

    ii. **A diagram** describing the system**.** Use LaTex Figure environment with proper **caption** and **\label**{fig:system}, and refer to the figure from the aforementioned verbal description "(cf. a diagram in Figure \ref{fig: system}.

    iii. **Implementation details** including the public libraries your use in your code (with proper citing of the relevant papers and git repositories), the cryptographic/ML primitives/models they rely on (again, with proper citing), and any other details required for **reproducing your code (reproducible science)**

c. **Empirical Evaluation**. Clearly describe here:

    i. **The experiments (i.e., tests)** you ran, including:

        1. the **hardware** on which it was run,

        2. the exact **parameters** used in the execution,

        3. **what properties were measured** (including their exact definitions if not trivial). Typically you'd measure: **accuracy** (specify the exact measure), **runtime** (measured in a standard time unit such as ms, sec, min), **communication** number of

rounds and total **communication volume** (measured in a standard size unite such as Bytes, KB, MB).

Here again, your guideline should that the report should include enough info to make you experiments fully **reproducible**.

ii. **Performance:** here you should include **text**, **tables** and **graphs** presenting the performance of your system on the executed experiments on the measured properties.

iii. **Discussion:** discuss your results, explain why they are significant and what might be lacking.

8. **Conclusions** Here you should write 1-2 paragraphs summarizing your results, explaining its importance and potential impact, and discussion open issues/problems for future work.

9. **Bibliography** Using proper LaTex bib file and bibliography following standard academic practice (see UoH library website for details).

10. **Appendices** include here any information you think is required but does not fit the prior sections.

Appendix B. Examples of DB and Models.

In this appendix we provide you with some links to biometric datasets and vector embedding models.

- Hint: You are welcome to use chatGPT for help writing scripts for various tasks, such as fine-tuning and computing similarities.
- If you train/fine-tune your own model, a machine with GPU can make things faster.

Face recognition

- Models:
  - VGGFace2. This model can be loaded using python libraries without need to download it manually. You can ask chatGPT for an example python script that uses this model.
    - See relevant paper here.
  - InsightFace Model Zoo
  - You can find the leading models for the LFW dataset here.
- Datasets:
  - DigiFace1M
    - A dataset of synthetic faces. The dataset contains 1M images of about 100,000 different synthetic individuals.
    - See the corresponding paper here.
    - The only model I tested on this dataset is VGGFace2. The VGGFace2 model does not seem to give good accuracy on the DigiFace1M dataset. If you choose to work with the DigiFace1M dataset, you can either try to fine-tune VGGFace2 on the DigiFace1M dataset (or even train from scratch), or try other models.
  - LFW
    - 13,233 images of 5,749 (real) people.
    - VGGFace2/InsightFace Model Zoo models may also work on this dataset directly without need to fine-tune them (although I did not try)
    - See leading models for LFW dataset here.
  - See several other datasets here.

Fingerprint identification

- Synfing dataset
  - A dataset of 100K pairs of synthetic fingerprint images (2 images for each finger).
- You can find a list of several datasets here, although these ones include less than 10K identities per dataset.
- You can find a possible model here.
  - This model was pre-trained on a specific dataset and it is likely to produce worse accuracy for other datasets.
  - You are welcome to try fine-tuning this model on the Synfing dataset or training it from scratch, although it is a bit challenging since you have to extract the minutiae of the fingerprint images of the Synfing dataset. Maybe you can find a method online to extract fingerprint minutiae, but I didn't find an easy method.
- Alternatively, you can train your own fingerprint detection model. No need to reinvent the wheel, just search online for a paper that introduces such a model and write a python code to implement the model from the paper. You can attach the paper to chatGPT, point it to specific sections and ask it to help implement a relevant python code.

Other sources

This survey reviews many papers that implemented biometric algorithms over HE. You are welcome to pick one of the reviewed papers and try to implement it.