CS250/EE387: Final project guidelines

Winter 2025

Overview

The final project should be in teams, of size up to four. Solo projects are allowed if you want. There are three forms the project can take, which are described in more detail below. They are:

- **Reading/Report**. You read some stuff and write a report.
- **I've had it with all this theory**. Investigate some instances of coding theory in the real world and teach us how they work.
- **Original work**. Original research involving error correcting codes, or new implementation/creation/development of something related to error correcting codes.

All types of projects will involve a write-up (10-15 pages long; instructor permission required for anything longer than that), and a short presentation. The format of the presentation will be determined once we know how many groups there are (after the project proposals are due). If there are many groups, we may do poster sessions and/or pre-recorded videos.

Please review the possibilities below, find teammates, and let us know what you plan to work on by **Friday February 14.**

Timeline

- Project Proposals due Friday Feb. 14 on Gradescope.
- We will confirm that your proposal looks good by Friday Feb. 21. If you get us your project proposal earlier, we can probably confirm that it looks okay earlier.
- In-person presentations will be in Week 10 (3/10 and 3/12).
- Pre-recorded project presentations due 3/11 by 11:59pm; folks doing pre-recorded presentations will watch others' pre-recorded presentations and post comments/questions by 3/14.
- Project Reports are due 3/14 (last day of classes) at 11:59pm, on Gradescope.

Project Proposal

What to hand in on gradescope by 11:59pm on Feb. 14:

- Who you plan to work with (each group should only submit one thing)
- Which type of project you want to do

- If the project is a reading/report type, on a topic listed below, say which topic and which papers you will read.
- If the project is not a topic suggested below, please include a few paragraphs describing the project or topic you have in mind, and what resources you will start with.
- If different groups want to work on the same topic, we may suggest that either they merge or work on slightly different aspects of the topic so that we don't have redundant presentations. Thanks in advance for being flexible.

We are happy to chat in office hours or answer questions beforehand to discuss your options or possible project ideas!

Grading

The report will be half of the grade, and the presentation is the other half.

The report will be assessed on the following criteria. First, all reports will be assessed on the technical content: is it sound, and does it convey a good understanding of the material? For Reading/Report projects or "I've had it with all this theory" projects, you will also be assessed on how well you synthesize all the materials out there into a coherent and engaging summary. For original work, you will be assessed as one would assess a normal academic paper, including motivation and clarity.

Reading/Report projects

A project of this type will involve reading up on a topic that we didn't have time to cover (in depth) in class. You should read a few papers (and you are also welcome/encouraged to look at other resources like lecture notes or textbooks), and synthesize this information into a coherent literature review that covers some of the main technical ideas in the area. Your intended audience should be someone who has just finished this class. Your goal is *not* to convince the reader that you are very smart and/or did a lot of work (we know both of those already). **Rather, the goal is to *teach* the reader something.**

Note that part of the project is to decide what you think is interesting: don't just regurgitate the material from the first source you find. Rather, do some research, find good sources, and come up with a nice story to tell.

Your report should be 10-15 pages long, demonstrate a technical understanding of the chosen topic, and be interesting to the reader.[1]

You are welcome to choose whatever topic you like, with instructor approval. A few suggestions are listed at the end of this document, but it's definitely not an exhaustive list! If you have another topic in mind (either which you stumbled on independently, or which we discussed in class too briefly), please ask us about it! We can help you find some starting resources if you would like.

I've had it with all this theory

A project of this type will involve you going out and finding some example of error correcting codes (ideally, one of the families we've talked about in class) in the real world (for example, in storage, satellite communication, etc), and reading up on it to figure out how it works in practice. Your report should highlight:

- What are the main coding-theoretic ideas behind this implementation?
- Why was this coding scheme chosen for this application instead of another one? What are the important desiderata and how are they acheived?

[1] I (Mary) am interested in any core material you can possibly come up with; this requirement is to synthesize it in an interesting way and to tell a good (if technical) story.

- What are the real-world issues that complicate matters? How does this implementation deal with them?
- Interesting historical trivia about the development of this application is encouraged.

Your goal is to present not only the technical details (although you should demonstrate a good understanding of this) but also an interesting (if technical) story.[2]

Original work

A project of this type will be either a research project or a new implementation. For example, perhaps you have found a way to use error correcting codes in your own research. Or perhaps you are already working on something directly related to error correcting codes. Or perhaps you want to build an app (e.g. a game or a learning tool) based on coding theory, or make a youtube video or classroom plan[3] that does a really good job teaching some aspect of coding theory. Or, perhaps you realize that sage (the computer algebra system) could really use a better way to do XYZ which is related to error correcting codes; or that what the world really needs is a tool to which implements Reed-Solomon error correction via aesthetically pleasing 3D printing[4] and want to implement this yourself.

In this case, your report should contain your results (or a way to access your results, if they are open-source code or a 3D-printed Reed-Solomon-related piece of artwork.)

If you have done original research, your report should be formatted like a research paper, including background, motivation, your results, and demonstrations (proofs or empirical work) that your results are sound.

If you have done an original implementation or built/created something, your report should outline the motivation for why your implementation/construction/creation is/will be useful or interesting, any relevant background, details of the implementation, as well as any tricky issues you ran into when developing it and how you got around them. Your report should also include examples of your implementation at work (empirical results about the speed/accuracy, or anything else that demonstrates that you have effectively solved this problem).

Note: If you set out to do original work, but encounter an issue[5] and do not manage to succeed by the end of the quarter (research is unpredictable; that's how it goes), then write up what you tried, what you learned, where you got stuck, and what you are thinking about possible ways forward.

[2] same as the previous footnote.

[3] In the past someone developed a Stanford SPLASH! short course.

[4] Okay, I don't really know what this means, but if you figure it out it could probably be cool.

[5] An issue means that the problem was harder than you anticipated. It does not mean that you were not able to spend enough time on your project.

Presentation Logistics and Guidelines

We will have a *mix* of in-person and pre-recorded presentations. See the Canvas announcement for which your group is doing and when. (Note: we gave everyone their preference, as per the form; let us know ASAP if you'd like to change formats).

In-person presentation. For your in-person presentation, you should prepare for a 12-minute presentation (including questions). Planning to talk for 8-10 minutes and answer questions for 2-4 minutes is a good breakdown. **The goal of your presentation is *NOT* to convey everything you have done/learned for your project—there is no way you can do that in 12 minutes!** Instead, your goal should be to tell your audience a clear narrative, and maybe teach them something, clearly, in at most 12 minutes. For example, you could give an introduction to a literature/problem/application area and explain one cool technique that you saw when reading. Or you could explain the statement of one result and why it is interesting. Or... (Please feel free to ask us for advice about what to focus on for the presentation!)

You are welcome to use slides or the board, but slides might make it easier to get through more material quickly. (Writing on the board takes *way* longer than you think it will!)

A schedule for in-class presentations will be posted soon (in Week 9).

Engagement: If you are giving an in-person presentation, you are expected to attend others' in-person presentations (on *both* Monday and Wednesday), and engage and ask questions to the extent that there is time. If you want to view and comment on the asynchronous presentations too, that would be fantastic!

Asynchronous presentation. To do an asynchronous presentation, you will pre-record a short video version of your presentation.[6] Your video should be about 12 minutes long.[7]

You should finish your presentation by **Tuesday March 11.** By 11:59pm on Tuesday, you should:

- Post your presentation somewhere that's accessible by a link (e.g., Google Drive, YouTube, Vimeo, ...).
- Post on Ed with a title, abstract, names of group members, and a link to your presentation.

Your videos should follow the same guidance as the in-person presentations above. Don't try to fit everything you learned in 12 minutes!!!

Engagement: If you are giving an asynchronous presentation, you are expected to watch and leave a substantive comment or question on at least two other presentations (two specific ones will be assigned[8]; if you want to engage with more, that would be fantastic!). You will leave your comments/questions as a reply to the Ed thread created by the presenting groups. Your comments/questions will be due on Ed by **11:59pm Friday March 14.** (After that, please respond to any comments/questions left on your own presentation).

Possible Topics for a Reading/Report Project

- Polar Codes
  - Arikan: Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. https://arxiv.org/abs/0807.3917
  - Guruswami and Xia: Polar Codes: Speed of polarization and polynomial gap to capacity https://arxiv.org/abs/1304.4321
  - Blasiok, Guruswami, Nakkiran, Rudra, Sudan: General Strong Polarization. https://eccc.weizmann.ac.il/report/2018/027/

[6] A good way to do this is to present in Zoom and record the Zoom session. If you have a Mac, I've found that iMovie is pretty good for editing if you want to edit it afterwards, but do not feel obligated to have a high-production-value video :).

[7] If you have a very good reason to go longer or shorter, you can make your case to us.

[8] Assignments will be posted by the time the videos are up.

  - Guruswami, Riazanov, Ye: Arikan meets Shannon: Polar codes with near-optimal convergence to channel capacity. https://arxiv.org/abs/1911.03858

- Expander Codes
  - Sipser and Spielman: Expander codes http://ldpccodes.com/papers/expandersIT.pdf
  - Zemor: On expander codes http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=910593 (The only online copy I can find is behind a pay-wall; you should be able to access it through the Stanford library with your Stanford credentials. If you are having trouble, please contact me).

- Reed-Muller Codes achieve capacity on the BEC/BSC
  - Kudkar, Kumar, Mondelli, Pfister, Sasoglu, Urbanke: Reed-Muller Codes Achieve Capacity on Erasure Channels https://arxiv.org/abs/1601.04689
  - Reeves, Pfister: Achieving Capacity on Non-Binary Channels with Generalized Reed-Muller Codes https://arxiv.org/abs/2305.07779

- Multiplicity Codes
  - You can choose either Local decoding of multiplicity codes, or list decoding, or both.
  - Local decoding:
    * Kopparty, Saraf, Yekhanin: High-rate codes with sublinear time decoding http://www.math.rutgers.edu/~sk1233/highratelocal.pdf
  - List decoding: (two papers with very similar results but different approaches)
    * Kopparty: List-decoding multiplicity codes http://www.math.rutgers.edu/~sk1233/part2.pdf
    * Guruswami and Wang: Optimal rate list-decoding via derivative codes https://arxiv.org/abs/1106.3951
    * (Building on the Guruswami-Wang paper: https://arxiv.org/abs/1805.01498)

- Applications in cryptography: There are many applications of error correcting codes in cryptography. One possibility for this project is to learn more about code-based schemes like McEliece. Here are a few suggestions along those lines:
  - You could take a look at some of the code-based schemes that participated in NIST's post-quantum cryptography competition. For example, three of the four candidates on this page https://csrc.nist.gov/projects/post-quantum-cryptography/round-4-submissions are code-based (classic McEliece, BIKE, HQC). For a project, you could take a look at how all of them work and compare and contrast. (Or you can try to break them!)
  - There are lots of connections between error correcting codes and zero knowledge proofs. The references linked in this podcast (https://zeroknowledge.fm/282-2/) seem like a good place to start. (Note: I have not listened to this podcast, it may also be useful, but I can't vouch for it. The references look reasonable.)

- Hardness of maximum-likelihood decoding of Reed-Solomon Codes
  - Guruswami and Vardy: Maximum Likelihood Decoding of Reed-Solomon Codes is NP-hard https://arxiv.org/abs/cs/0405005
  - Cheng and Wan: On the list and bounded distance decodability of Reed-Solomon Codes http://www.cs.ou.edu/~qcheng/paper/ld.pdf
  - Gandikota, Ghazi, Grigorescu: NP-Hardness of Reed-Solomon Decoding and the Prouhet-Tarry-Escott Problem https://www.cs.purdue.edu/homes/egrigore/papers/RS_BDD_hard.pdf

- Fast implementations of the Guruswami-Sudan algorithm
  - Alekhnovich: Linear diophantine equations over polynomials and soft decoding of Reed-Solomon codes. http://ieeexplore.ieee.org/document/1181968/
  - Chowdhury, Jeannerod, Neiger, Schost, Villard: Faster Algorithms for Multivariate Interpolation with Multiplicities and Simultaneous Polynomial Approximations. https://arxiv.org/pdf/1402.0643v2.pdf, and the references in Table 1 of that paper.
  - If you are interested in a generalization to fast list-decoding of folded RS codes and multiplicity codes, check out: Goyal, Harsha, Kumar, Shankar: Fast list-decoding of Univariate Multiplicity and Folded Reed-Solomon Codes. https://arxiv.org/abs/2311.17841.
- More group testing
  - Porat and Rothschild: Explicit Non-Adaptive Combinatorial Group Testing Schemes https://arxiv.org/abs/0712.3876
  - Indyk, Ngo, Rudra: Efficiently Decodable Non-Adaptive Group Testing https://www.cse.buffalo.edu/~hungngo/papers/soda10.pdf
  - Guruswami, Wang: Nonadaptive Noise-Resilient Group Testing with Order-Optimal Tests and Fast-and-Reliable Decoding https://arxiv.org/abs/2311.08283 (and more references therein).
- Coding theory in compressed sensing
  - Gilbert, Li, Porat, Strauss: For-all sparse recovery in near-optimal time https://arxiv.org/pdf/1402.1726v1.pdf
  - Ngo, Porat, Rudra: Efficiently Decodable Compressed Sensing by List-Recoverable Codes and Recursion http://www.cse.buffalo.edu/faculty/atri/papers/coding/npr12.pdf
- List-decoding and pseudorandomness
  - Vadhan: Pseudorandomness. (Shorter survey: http://people.seas.harvard.edu/ salil/research/unified.pdf, Longer book form: http://people.seas.harvard.edu/~salil/pseudorandomness/) These have lots in them and it might be good to focus on just one relationship (eg, list decoding and extractors).
  - Ta-Shma and Zuckerman: Extractor codes https://www.cs.utexas.edu/~diz/pubs/extractor-codes.ps
- Locally testable codes and PCPs
  - Survey by Goldreich: http://www.wisdom.weizmann.ac.il/ oded/PSX/ltc+p.pdf
- Combinatorics of Reed-Muller codes
  - Kaufman, Lovett and Porat: Weight Distribution and List-Decoding Size of Reed-Muller Codes http://conference.iiis.tsinghua.edu.cn/ICS2010/content/paper/Paper_33.pdf
  - Bhowmick and Lovett: List decoding Reed-Muller codes over small fields https://arxiv.org/abs/1407.3433
- Concatenated codes on the GV bound and with efficient list decoding to capacity
  - Thommesen: The existence of binary linear concatenated codes with Reed-Solomon outer codes which asymptotically meet the Gilbert-Varshamov bound. http://ieeexplore.ieee.org/document/1056765/
  - Guruswami and Rudra: Concatenated Codes Achieve List-Decoding Capacity https://eccc.weizmann.ac.il/eccc-reports/2008/TR08-054/index.html

- Explicit codes near the GV bound
  - Ta-Shma: Explicit, almost optimal, epsilon-balanced codes. https://eccc.weizmann.ac.il/report/2017/041/
  - Jeronimo, Quintana, Srivastava, Tulsiani: Unique Decoding of Explicit ε-balanced Codes near the GV bound. https://ttic.uchicago.edu/~madhurt/Papers/gv-unique-decoding.pdf.
  - Richelson, Roy: Gilbert and Varshamov Meet Johnson: List-Decoding Explicit Nearly-Optimal Binary Codes. https://ieeexplore.ieee.org/document/10353086
- Coding for Interactive Communication.
  - Here's a great survey by Gelles: http://www.eng.biu.ac.il/~gellesr/survey.pdf
  - Gil Cohen, Bernhard Haeupler, Leonard Schulman: Explicit Binary Tree Codes with Polylogarithmic Size Alphabet. https://eccc.weizmann.ac.il/report/2018/032/
- Coding for flash memory
  - Jiang, Mateescu, Schwartz, Bruck: Rank Modulation for Flash Memories. http://www.paradise.caltech.edu/papers/etr086.pdf
  - Barg and Mazumdar: Codes in Permutations and Error Correction for Rank Modulation. https://arxiv.org/abs/0908.4094
  - Mazumdar, Barg, Zemor: Constructions of Rank Modulation Codes. https://arxiv.org/abs/1110.2557
- Quantum error correcting codes
  - There is a good introduction in Chuang and Nielson's "Quantum Computation and Quantum Information" textbook.
  - I like these YouTube lectures by Daniel Gottesman: https://www.youtube.com/watch?v=ltJ1jXQeDl8 (Part 1) and www.youtube.com/watch?v=cUqys29d0YA (Part 2)
- List-decodability of RS codes beyond the Johnson bound.
  - Ferber, Kwan, Sauermann: List-decodability with large radius for Reed-Solomon codes. https://arxiv.org/abs/2012.10584
  - Brakensiek, Gopi, Makam: Generic Reed-Solomon Codes Achieve List-decoding Capacity. https://arxiv.org/abs/2206.05256
  - Alrabiah, Guruswami, Li: Randomly punctured Reed–Solomon codes achieve list-decoding capacity over linear-sized fields . https://arxiv.org/abs/2304.09445