Final Project Guide
CS267A Spring 2023

1 Introduction

This document describes the instructions for the final project. A project will be done by a team consisting of 3 people. We highly recommend you to start finding your teammates as soon as possible after reading this guide. Ideally this project would be related to your research interest, though this is not necessary. This project should require a significant amount of work, and should involve either theoretical or experimental work. This means you either need to state and prove interesting theorems, provide interesting working code, or both. The work must be original and written solely by the group participants.

The projects will be managed by the TAs. Ask on Piazza or email the TAs if you have questions about the projects, and please include "CS267A Project" at the beginning of your email:

- Discussion forum: `https://piazza.com/ucla/spring2023/cs267a`
- William: `williamxcao@ucla.edu`
- Poorva: `poorvagarg@cs.ucla.edu`

Before selecting a project, you must read the syllabus, the rest of this document, and take note of the topics that will be relevant to your project.

If you have a topic you like and need to form a team, feel free to reach out to the rest of the class on the discussion forum on BruinLearn.

Academic Honesty You are encouraged to use open-source code and libraries as long as you respect the licenses of those libraries. In particular, be sure to attribute any code that you use from an open-source library appropriately: *do not copy and paste someone else's code without properly citing its original source*. In general, we have *zero tolerance* of academic plagiarism or any other dishonest behavior.

1.1 Rubric

The project will be evaluated on the following rubric:

- 10% Project Proposal
- 10% Project Check-In
- 80% Final Writeup

The proposal and check-in steps will be evaluated on a pass/no-pass basis. The proposal and check-in deadlines are set up to encourage you to start working on this project early instead of leaving all work to the last minutes. Based on the past year's experience, office hours will be overwhelmed after week 8, and constructive suggestions and comments can only be promised if you check in with us before the deadlines. A detailed description of how the final writeup will be evaluated is provided in Section 2.3.

2 Project Requirements

The project will consist of three parts: a proposal, a check-in, and a final writeup. All projects should be uploaded to BruinLearn.[1] Contact the TA if there are any questions or concerns. A LaTeX template is provided for you in BruinLearn along with this guide.

[1] `https://bruinlearn.ucla.edu/courses/129738/`

2.1 Project Proposal (Due Friday May 12)

The project proposal will describe the scope and direction of your project. The proposal should be typeset using LaTeX.[2] The goal of the proposal is to help you to choose a good project to work on: one which is feasible, appropriately scoped, and educational. Please limit the length of this document to 3 pages of the provided template.

You must provide a proposal document before you begin working on a self-selected project. The proposal may be rejected and returned to you for revision. In this case, talk to your TA and revise your proposal before proceeding with the project.

If you want to change the proposal in a major way or change your group after your proposal has been accepted, talk to your TA. For the project proposal, we expect a PDF to be uploaded to BruinLearn with the following sections:

- *Abstract*: In about 200 words, describe what you want to do for your project. Clearly explain the goals and anticipated technical work involved.
- *Group*: The project must be done in groups, preferably of 3 people. Special consideration may be given to smaller groups; talk to us *early* if you want to do this. Comment on the background of your group. Include the preferred contact information for each member of the group.
- *Motivation and Introduction*: What is the motivation for your proposal? Why is this an interesting project? Why do you want to do this?
- *Course relevance*: Which topic from the syllabus does your project relate to? If it is not on the syllabus, how is it related to the topic of the course? Specifically, does your project involve any of the following:
  - relational or graph data;
  - a relational modeling framework discussed in the course;
  - probabilistic programming languages;
- *Background*: What is the relevant technical material for doing your project? What do you need to learn that you do not know yet, and what do you already know? Provide some relevant references if they are useful.
- *Measures of success*: What are your measures of success for your project? Provide the following:
  - Baseline: What is the minimum level of progress at the end of the quarter to be considered a success?
  - Medium: What would be considered an unqualified success?
  - Stretch: If you have extra time, what might you consider working on?

  These are to help you scope the project, and are subject to change during the quarter.
- *Planning and Timeline*: Include a rough timeline for the rest of the quarter explaining key milestones towards completing your project. Provide a rough sketch of how you might divide up responsibilities amongst your group.

2.2 Project Check-In (Due Wednesday May 31)

The goal of this check-in is to help you to make progress towards your goal of finishing the project at the end of the quarter. This is a chance for you to re-evaluate and clarify your initial proposal in light of having worked on the project (you've started working on it, *right*?). We will open the link to submit check-ins after the proposal is due. And we also encourage you to submit check-ins early, so you have more chance to iterate your project with the TAs. For the Project Check-In, you must submit a PDF to BruinLearn containing the following sections:

- Update your Project Proposal to reflect any changes that have happened since it was accepted. Clearly note significant changes.
- *Technical Contribution*: At this point, you should have a clear idea of what exactly it is you want

[2] If you do not know how to use LaTeX, follow the tutorial at `https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes`.

to achieve technically with your project. Add a new section between "Background" and "Measures of Success" called "Technical Contribution". This is the most important part of your project, and it is more open-ended. The goal of this section is to convince us that your project has technical merit.

The following questions are for you to consider, but you are not required to answer them directly:

- If your project has a theoretical component, what are the theorems that you need to prove and important definitions?
- If your project is experimental, what is your objective? If you want to implement something, give a thorough description of the tool you have in mind. If you want to test a scientific hypothesis, describe the hypothesis.
- Have you made any progress towards your project objectives? If so, what? What do you anticipate being the challenges moving forward?

2.3 Final Writeup (Due Friday June 16)

The final writeup is a more open-ended document than the previous two documents. This document as a hard limit of 6 pages not including references, with unlimited appendices. It should follow a standard research paper structure and use the provided template. We suggest the following rough structure, but you may use whatever structure you like:

- *Abstract*: Describe the contents of the project in about 200 words.
- *Introduction and Motivation*: Explain the selected problem and describe why it is interesting and related to the course. Outline technical contributions.
- *Background*: Describe the relevant background material for understanding the technical work.
- *Technical Contribution*: Explain what you did for your project in detail. Describe your experiments or theoretical work.
- *Related Work*: Describe related work.
- *Conclusion*: Describe future work. Did you succeed in your initial objectives? What would you do differently next time?
- *Feedback*: Please give us some feedback on the project. What went well? What would you prefer be done differently next time?

We will evaluate your final writeup on the following metrics:

- *Quality of Writing*: Are the ideas clearly presented and easy to follow? Is there appropriate usage of figures?
- *Technical Quality*: Is the work technically sound? Is there a scientific hypothesis? If there are experiments, are they well-designed to answer scientific questions? If there are theorems and definitions, are they clearly stated and proven correct? If code is provided, does it work? How does the final product compare to what was discussed at the checkpoint?
- *Course relevance*: Is the project relevant to the subject matter of the course?
- *Scholarship*: Is the appropriate related work discussed and compared against? Is the work well contextualized in the field?

Provide a PDF and working code on BruinLearn. Provide instructions for how to run your code in a README.

2.4 Invited Presentations

A subset of projects will be selected to give short presentation in class on the final day. This is optional, but highly encouraged: it is a chance for you to sharpen your public speaking skills, for the class to learn about your interesting project, and for the instructors to give additional feedback.

3 Suggestions

Some suggested projects are listed below. A longer list of topics that could inspire you to formulate a project is available on `https://goo.gl/Pxzx8U`.

Implement a probabilistic programming language Pick an existing probabilistic programming language, and implement a simple compiler which does monte carlo inference. Alternatively, you may choose to define your own probabilistic programming language or a different inference algorithm.

Implement a new feature in existing PPL Pick an existing PPL and identify a functionality that it does not currently support. Motivate its use case and implement the functionality with test cases.

Modeling data with a probabilistic programming language Choose a dataset that you are interested in analyzing, and a probabilistic programming language or toolkit, and do some interesting analysis leveraging the power of the tool you choose. For example, you could try using Problog or Dice to model social networking or genomic network data.

Modeling systems with a probabilistic programming language Pick a system or mechanism from some other field (e.g., a routing protocol, randomized algorithm, etc.), and explore using a probabilistic programming language to reason about the behavior of such systems, diagnose errors, and provide useful analysis.

Implement a Probabilistic Database System Preferably using a standard database management system such as SQL, implement the lifted inference algorithm described in class for probabilistic databases. You can also implement an existing approximation scheme, or come up with your own.

Implement a first-order rule learner Pick an algorithm for learning first-order rules from data and implement it. Then, pick a relational data set and apply your rule learner to see if you can learn anything interesting. Think about how to incorporate uncertainty into the learned model.

Exploiting symmetries in MCMC samplers Markov Chain Monte Carlo (MCMC) samplers are widely applied in many domains where exact inference is not feasible. One recent noticeable line of efforts to speed up MCMC is to exploit model symmetries. This line of research also falls under the umbrella of lifted inference. For this project, one can either design a new method to more efficiently detect symmetries, or design a new sampler.

Prove novel properties of Inference algorithms and PPLs We defined the semantics of Dice in the class. Define the semantics of another PPL which does not define it formally. Prove the correctness of an inference algorithm. Alternatively, find other interesting properties and prove them.

Neural-symbolic learning for structured data Many domains that are appealing to machine learning researchers are structured, in the sense that there exists some relationship between the outputs. For example, data points are not i.i.d. samples or different parts of the outputs are not independent from one another. One staightforward starting point to work on this topic is to first find those domains and then identify the relationships in them and apply some established well-performing methods (e.g. DeepProbLog, Semantic Loss) on them.

Explore a recommender system Using a rating dataset such as MovieLens, choose a technique such as latent dirichlet allocation or probabilistic matrix factorization and implement it. Explore different techniques and their effect on performance.

Remember, you can always come up with your own projects from scratch that do not overlap with the aforementioned suggested projects. As long as the project falls under the broad scope of this

course, feel free to pursue it. To help you formulate an appropriate project we have compiled a non-exhaustive list of research topics that are in the scope of this course: `https://goo.gl/Pxzx8U`.

Good luck!