CS224N: Project Report Instructions

Last updated on March 8, 2020

Each team submits one project report, which is worth 30% of your overall grade. This document specifies what information you should include in your report. It applies to both default and custom projects. For this report, we require that you create your PDF file using the LaTeX template provided in this link:

`https://www.overleaf.com/read/hhshmzgtcsmh`

In addition, we encourage you to take a look at sample reports from last year, which can be found here:

`https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/project.html`

1 Report contents (6-8 pages[1])

Your final report should be written in the same style as a NLP / Deep Learning research paper, and written in a way that a fellow CS224N student could understand. Your report should be 6-8 pages (excluding references). Consider using the following section structure, though you can use a different structure.

Key information. Your report should have the following information:

- **Title**: The title of your project.
- **Team member names**: List the names and `@stanford.edu` email addresses of all of your team members.
- **Custom or Default Project**: Indicate which you are doing.
- **Mentor**: If you have an external mentor and/or CS224N staff mentor, write their name(s).
- **(Optional) External Collaborators**: If you have any collaborators who are not CS224N students, list them.
- **(Optional) Sharing Project**: If you are sharing this project between CS224N and another class, indicate it here.

[1] We expect that default project reports may generally be shorter than custom project reports. If you're doing a relatively straightforward default project, make sure to thoroughly describe your approach and experimental details, and thoughtfully discuss your results.

Abstract. An abstract should concisely (less than 300 words) motivate the problem, describe your aims, describe your contribution, and highlight your main finding(s).

Introduction. The introduction explains the problem, why it's difficult, interesting, or important, how and why current methods succeed/fail at the problem, and explains the key ideas of your approach and results. Though an introduction covers similar material as an abstract, the introduction gives more space for motivation, detail, references to existing work, and to capture the reader's interest.

Related work. This section helps the reader understand the research context of your work, by providing an overview of existing work in the area.

- You might discuss: papers that inspired your approach, papers that you use as baselines, papers proposing alternative approaches to the problem, papers applying your methods to different tasks, etc.
- This section shouldn't go into deep detail in any one paper (for example, there probably shouldn't be any equations) – instead it should explain how the papers relate to each other, and how they relate to your work.
- Attempt to demonstrate, as you review the literature, limitations or motivations that point to why *your* work is a nice next step, or useful replication, or promising analysis (or otherwise, if your work doesn't fall into these categories!).

Approach. This section details your approach(es) to the problem. For example, this is where you describe the architecture of your neural network(s), and any other key methods or algorithms.

- You should be specific when describing your main approaches – you probably want to include equations and figures.
- You should also describe your baseline(s). Depending on space constraints, and how standard your baseline is, you might do this in detail, or simply refer the reader to some other paper for the details. Default project teams can do the latter when describing the provided baseline model.
- If any part of your approach is original, make it clear (so we can give you credit!). For models and techniques that aren't yours, provide references.
- If you're using any code that you didn't write yourself, make it clear and provide a reference or link. When describing something you coded yourself, make it clear (so we can give you credit!).
- As you're setting up equations, notation, and the like, be sure to agree on a fixed technical vocabulary (that you've defined, or is well-defined in the

literature) before writing and use it consistently throughout the report! This will make it easier for the TAs to follow, and is nice practice for research writing in general.[2]

Experiments. This section contains the following.

- **Data**: Describe the dataset(s) you are using (provide references). If it's not already clear, make sure the associated task is clearly described. Being precise about the exact form of the input and output can be very useful for readers attempting to understand your work, especially if you've defined your own task.
- **Evaluation method**: Describe the evaluation metric(s) you use, plus any other details necessary to understand your evaluation. Some projects will have clear metrics from prior work on given datasets, but we realize that other projects will define their own metrics. If you're defining your own metrics, be clear as to what you're hoping to measure with each evaluation method (whether quantitative or qualitative, automatic or human-defined!), and how it's defined.
- **Experimental details**: Report how you ran your experiments (e.g. model configurations, learning rate, training time, etc.)
- **Results**: Report the quantitative results that you have found so far. Use a table or plot to compare results and compare against baselines. [3]
  - If you're a default project team, you should **report the F1 and EM scores you obtained on the test leaderboard** in this section. Make it clear whether you are on the non-PCE or PCE leaderboard. You can also report dev set results if you like.
  - Comment on your quantitative results. Are they what you expected? Better than you expected? Worse than you expected? Why do you think that is? What does that tell you about your approach?

Analysis. Your report should include *qualitative evaluation*. That is, try to understand your system (e.g. how it works, when it succeeds and when it fails) by inspecting key characteristics or outputs of your model.[4]

[2] In experimental work, this could mean giving a specific name to each method, each dataset, each baseline; it could also mean making consistent use of mathematical notation where appropriate

[3] Some analysis-centric custom projects will not have model-based baselines to compare against; in this case, consider whether there are simpler analysis methods than your proposed method which might achieve the same goal. It would be great to evaluate them, or at least discuss why they are insufficient compared to your method. Even in analysis projects, it's important to ask whether we could be achieving similar insights using simpler ("baseline") methods. So, if you don't have a baseline analysis method to compare against, discuss some alternative methods of analysis that one could use, and why (mathematically, linguistically, or otherwise) you believe your proposed method is superior.

[4] For some analysis-centric custom projects, it might seem odd to have a separate analysis section. It's up to you whether this section is included explicitly, but it might be a good idea.

- Types of qualitative evaluation include: commenting on selected examples, error analysis, measuring the performance metric for certain subsets of the data, ablation studies, comparing the behaviours of two systems beyond just the performance metric, and visualizing attention distributions or other activation heatmaps.
- The *Practical Tips for Final Projects Lecture Notes* has a detailed section on qualitative evaluation – you may find it useful to reread it.

Conclusion. Summarize the main findings of your project, and what you have learnt. Highlight your achievements, and note the primary limitations of your work. If you like, you can describe avenues for future work.

References. Your references section should be produced using BibTeX.

Appendix (optional). If you wish, you can include an appendix, which should be part of the main PDF, and does not count towards the 6-8 page limit. Appendices can be useful to supply extra details, examples, figures, results, visualizations, etc., that you couldn't fit into the main paper. However, your grader *does not* have to read your appendix, and you should assume that you will be graded based on the content of the main part of your paper only.

2 Improving your technical writing

As a reminder, the *Milestone Instructions* included extensive resources to help you improve your technical writing. You can use these, and previous feedback you've received, to improve your technical writing.

Always remember to be precise, use consistent technical terminology, and define terms that are clear to you now but aren't known to the average CS224N student.

3 Grading and feedback

Your project report will be graded holistically, taking into account many criteria: originality, performance of your methods, complexity of the techniques you used, thoroughness of your evaluation, amount of work put into the project, analysis quality, writeup quality, demonstrating strong understanding, etc.

Your report will be graded by two staff members, whose scores will be combined into your final score. You will also receive some brief feedback on your

Your analysis method hopefully has some quantitative evaluation, and you likely had to come up with it. In this section, consider giving examples of your analysis method in action on a given input, or graphs of the dataset; in general, attempt to give the reader intuition and insight into what your model is doing beyond the "topline numbers" described and discussed in the results section.

report. If you are doing a custom project, your CS224N staff mentor will be one of the graders.

4 Code

We ask you to submit your code as a zip file (up to 1MB) to Gradescope.

- **Do** include all project code written or adapted by you.
- **Don't** include the whole source code for off-the-shelf packages that you used without adapting (e.g. CoreNLP or PyTorch).
- **Don't** include model checkpoints or data.

Your code will not be graded—we collect it so that we can investigate honor code issues if necessary.

5 Team contributions

If you are a multi-person team, we ask you to submit (on Gradescope) a brief summary of what each team member did for the project (about 1 or 2 sentences per person). We will read these descriptions. For almost all teams, it will have no effect (i.e. team members all receive same grade), but for teams with considerably unequal contribution, we may investigate and/or give different grades to team members.

6 Submission instructions

To summarize, here are the instructions:

- Submit your report to Gradescope under [Default Final Project - Final Report] or [Custom Final Project - Final Report]. Make sure to tag all of your team members—only tagged team members will receive credit.
- Submit your code to Gradescope under [Final Project - Code].
- If you are a multi-person team, submit a brief description of team contributions to Gradescope under [Final Project - Team Contributions].

The due date is **11:59 PM on Friday March 13** (note that we extended the time to be midnight!), and teams can spend up to three late days on the project report. However, all reports must be uploaded to Gradescope by **4:30 PM (not 11:59 PM) on Monday March 16** in order to be graded—if you miss this deadline, your report might go ungraded.

7 Posting reports online

All final reports will be posted on the CS224N website. If you *do not* want your report to be published online, please let us know in a private Piazza post, and we won't upload it.