CS378: Natural Language Processing Final Project

Proposal Due Date (for independent projects ONLY): Thursday, October 20 at 11:59pm CST
Check-In Due Date (for all projects): Friday, November 18 at 11:59pm CST (no slip days)
Final Report Due Date: Thursday, December 9 at 11:59pm CST (no slip days)

Collaboration You are free to work on this project in teams of two (encouraged) or individually. Individual projects can be less ambitious but should not be less complete: a half-implemented system does not make a good project outcome. All partners should contribute equally to the submission, and all partners will receive the same grade for it. You are also free to discuss your project with others in the course, though only the people on your team should contribute to the actual implementation/experimentation involved. Any external resources used must be clearly cited.

If you're pursuing an independent project, you **may** collaborate with someone from outside the course. You also **may** use an independent project as a final project for another course as well, **provided you have the approval of the other professor**.

Assignment

You have two options for your final project:

1. A standard project, to be announced later, which will still give you significant leeway for exploration. (You can see the project offering from Fall 2021; we expect this year will be similar)

2. An independent project of your choosing

1 Your own project

You also have the option to pursue your own independently-chosen final project. Your proposal in this case should, as precisely as possible, describe the problem you want to solve or understand and the work you propose to do to solve it. Make sure you describe any datasets you plan to use and demonstrate that what you're proposing is feasible. Working on adjacent areas of machine learning (e.g., computer vision or robotics) is okay as long as there is a reasonable connection to concepts from this course. Ask the course staff if you're unsure.

Scope The expected scope should be similar to the standard project. In particular, you should either have a nontrivial piece of implementation (putting together existing pieces can be nontrivial) or analysis. The project should also engage with the course concepts. One example of an **inappropriate** project is scaling an existing machine translation to run on a cloud framework: this may have some challenges and is probably a sufficient amount of work, but mostly involves engineering and concepts not related to linguistics, machine learning, or algorithms as discussed in class.

Access to Cohere Special for this semester, we are able to provide students access to the Cohere platform `https://os.cohere.ai/`. This is an interface to a large language model able to do classification, embedding, and generation tasks, similar to GPT-3. Cohere will provide up to $1000 of credits per student; a two-student team could receive $2000.

At the time of your proposal, you can request this access. If your project proposal is approved, we will follow up with next steps about how to obtain access. At the end of the semester, Cohere will send an optional survey soliciting feedback on your experience using Cohere's platform. You also have the opportunity to submit your project to Cohere, and student projects may be featured on the Cohere blog and you will receive branded swag from Cohere.

This is a special deal with Cohere only, and unfortunately we cannot provide funding for GPT-3 or to general-purpose cloud compute resources.

Proposal For independent projects only, there is a proposal due before the official start of the final project period. This proposal should be **1/2 to 1 page**. In it, you should describe the problem you plan to solve, what data you plan to use, any existing systems you plan to start from, and what your estimate of the computational requirements will be. We will evaluate these proposals primarily to see whether your idea is appropriate in scope and whether the project is feasible: will you run into issues with lack of data or proprietary data, and will the project's computational requirements be realistic given the resources you have available.

2 Deliverables and Grading

The final project is worth 25% of your course grade. The deliverables are as follows.

Check-In (10 points) You should turn in a check-in (at least a couple paragraphs, no more than 1 page) on check-in due date (a bit less than halfway through). This check-in should outline what you've looked into so far and what your plan is for the remainder of the project (e.g., "I/we want to investigate X, we got the basic system running and looked through 10 data examples to confirm that our idea might work"). The course staff will then provide feedback and guidance on the direction to maximize the project's change of succeeding. The check-in is worth 10 points and is graded based on whether you provided a good assessment of your progress so far, *not* the progress itself.

Code You should also submit any code you wrote on the project due date, but this is **for documentary purposes only**; we will not be attempting to verify your results by running the code. Please do not include large data files or external resources you used that might be needed to execute it.

Final Report The primary deliverable is a paper written in the style of an ACL[1]/NeurIPS/etc. conference submission.[2] It should begin with an abstract and introduction, clearly describe the proposed idea or exploration, present technical details, give results, compare to baselines, provide analysis and discussion of the results, and cite any sources you used.

This paper should be between 3 and 8 pages excluding references. Different projects may take different numbers of pages to describe, and it depends on whether you're working by yourself or in a group. If you have lots of analysis and discussion or are trying something more ambitious, your paper might be longer; if you're implementing something complex but succinctly described, your paper might be shorter.

Your project is *not* graded solely on the basis of results. You should approach the work in such a way that success isn't all-or-nothing. You should be able to show results, describe some successes, and analyze

[1] Style files available here: `http://www.acl2019.org/EN/call-for-papers.xhtml`

[2] The Iyyer et al. paper is a good example of this: `https://people.cs.umass.edu/~miyyer/pubs/2015_acl_dan.pdf`

why things worked or didn't work beyond "my code errored out." Think about structuring your work in a few phases so that even if everything you set out to do isn't successful, you've at least gotten something working, run some experiments, and gotten some kind of results to report.

Grading: We will grade the projects according to the following rubric:

- **Scope (25 points)**: Is the idea of sufficient depth for a course project? Concretely, we expect that you do something along one of the three axes above and make a reasonable effort to execute it. While it does not have to work wonderfully, you will lose points here if all you can show is shallow analysis of the base system.

- **Implementation (25 points)**: Is the implementation described reasonable? Is the idea itself technically sound? You might lose points here if we perceive there to be a technical error in your approach. For example, perhaps you added a module to the neural network that leads to no performance change, but it's because it mathematically led to no change in the model due a conceptual error.

- **Results/Analysis (25 points)** Whether the results are positive or negative, try to motivate them by providing examples and analysis. If things worked, what types of errors are reduced? If things didn't work, why might that be? What aspects of the data/model might not be right? There are a few things you should report here: **Key results**: You should report results from a baseline approach as well as your "best" model. **Ablations**: If you tried several things, analyze the contribution from each one. These should be *minimal* changes to the same system; try running things with just one aspect different in order to assess how important that aspect is.

- **Clarity/Writing (15 points)**: Your paper should clearly convey a core idea/hypothesis, describe how you tested it/what you built, and situate it with respect to related work as best you can. **Abstract and Introduction**: Did you provide a clear summary of the motivation, methodology, and results? **Method**: Is the presentation of what was done clear? **Results**: Is the work experimentally evaluated? Are there clear graphs and tables to communicate the results? Don't just inline 1-2 numbers; make your analysis more detailed than that.

Compute and Feasibility **Large neural net methods can be slow to train!** Training a model on even 10,000 QA examples can take hours. Keep this in mind when deciding what kind of project to do. Working on linguistic constraints is likely to be the least resource-intensive, whereas developing a new neural architecture is likely to be the most.

Google Cloud Platform offers free credits upon signing up for a new account, which are likely sufficient to run some large-scale experiments for the course. The course staff are able to provide limited support on how to use GCP, but you'll mostly have to figure this out yourself.

Extra Credit: You can earn 2 points of extra credit by filling out the eCIS course evaluation. **Take a screenshot of the page showing that you've completed the survey (not your response itself, which is confidential) and upload it along with your final project.**

References