

# Final Project Schedule and Guidelines

## CS366: Computational Linguistics (Spring 2020)

### 1 Timeline

- Project proposal due Monday 4/6 11:59PM
- Project status updates due every Friday 11:59PM, starting April 10 and ending May 8
- Project presentations (10 min. each) on zoom Monday 4/27, Wednesday 4/29, and Monday 5/4
- Final project submission May 19, 11:59pm

### 2 Preliminary project presentation

Your project presentation should include:

1. A description of your problem and motivations
2. A brief survey of existing work, including at least two references.
3. A concrete description of your proposed solution(s), including the data and tools you will be using.
4. Description of work you have already completed. This section must not be empty. Plan to start working on your project as soon as you submit the proposal on April 6! Not only does it give you more time, it is also a chance to test out your ideas or change your plans.
5. If you're not working individually, precise description of the responsibilities of each team member.
6. Milestones. Be realistic about the milestones, but not lazy.
  - a. The minimum outcome of your project by the final submission on May 19.
  - b. The ideal outcome of your project by the final submission.

### 3 Project teams

You may work on your project individually or with a partner. Teams of more than two students might be allowed if the project appears to be of sufficient scope, but check with me first. Your project write-ups and presentations may be done jointly, but the individual contributions of each member must be clearly delineated and specified. This is really to protect you in case your partner slacks off!

## 4 Choosing a topic

You should make an appointment to see me (as a team if working with others) during the week of April 2-6 to discuss your topic. A good approach is to come up with an idea or area that interests you, even if you don't have any idea how to implement it. In our discussion I can help you along, or find a way to delimit the project in ways that make it doable.

If you can't think of a project, you can do several things:

- Look at the so-called “shared tasks” run under the rubric of SemEval. [SemEval Wikipedia page](#) gives links to tasks up to 2015; [Semeval 2016](#), [SemEval 2017](#), and [SemEval 2018](#) list some recent tasks that could spark ideas.
- Look at recent papers from major NLP conferences, many of which can be found in the [ACL Anthology](#) and the [Language Resources and Evaluation Conference \(LREC\) proceedings](#).
- Look at some final project topics from undergraduate students in similar classes at [Stanford](#) and [Wisconsin](#).
- Extend a homework assignment into a final project by generalizing or adding techniques, etc.
- Talk to me. I have a lot of ideas about potential projects!

The most important criterion for choosing a topic is that it genuinely excites you. Don't be afraid to get creative. The second is feasibility—you have roughly a month and a half to work on this, so set your goals realistically. It goes without saying that your topic should also be (1) directly related to computational linguistics, though not necessarily something we've done in class, and (2) awesome.

### 4.1 Types of projects

There are a few different routes you could take (which are not mutually exclusive by any means).

1. **Computational Research** Identify a task and develop a method to solve the problem. The emphasis of a research project should be on coming up with a method for a given task and measuring its performance. The task by itself could be entirely novel, or you could explore new ways of attacking a traditional NLP task (e.g., coreference resolution, named entity recognition, discourse analysis, semantic role labeling, time and/or event recognition, relation extraction, textual entailment, etc.). Any traditional NLP tasks could also be applied to a new domain or language.
2. **Software Application** Build a usable program for a task. Unlike research projects, the emphasis is on the final product itself, so you can implement an existing algorithm.

3. **Data Analysis** Use computational tools to analyze linguistic data. Think of problems, possibly from other classes, that would benefit from a quantitative analysis. Frame your hypothesis clearly.

4. **Linguistic Annotation** There are several kinds of possible annotation projects:

- Write specifications and annotate documents for a particular phenomenon. Possibilities:
  - Find one annotator in addition to yourself, so that it is possible to measure inter-annotator agreement; multiply annotate and adjudicate a sample for evaluation purposes.
  - Design and implement a simple annotation project using Amazon Turk (it may cost a little money—If anyone is interested in this route, I can provide more info and possibly some funding (\$50 buys a lot of annotation).
  - Apply a known type of annotation (named entities, part of speech, chunking, semantic role labeling, etc.) to a new domain of text: web data, technical data, a new language, etc.
  - Develop specifications and annotate new classes of named entities, Relations, or Events
  - Develop specifications and annotate an interesting phenomenon (e.g., quantifier scope, sentiment (your version), idiomatic expressions, metaphor...)

You can use any external toolkits or open-source code in addition to code that you write yourself.

## 4.2 Questions to Consider

Come up with a few different problems at first, and sift through them with the following questions in mind.

1. Is the task well-defined? Specifically, what is the input and output? Do you have a hypothesis to test?
2. What is the data you will need for this project? Is this data easily available? Will it require pre-processing to be usable in your project (be careful not to end up having to spend most of your time cleaning data!). Talk to me if you're having trouble locating data sources. You should have the data in hand by the time you submit your proposal.
3. What prior work has been done on this idea or related questions? Try to find at least two major references. For Computational Research projects, search on Google Scholar or the ACL anthology (<http://aclweb.org/anthology-new>) for relevant papers. If the idea you're proposing is completely novel, look for papers that address similar topics. For

Data Analysis projects, make sure you're searching for computational work as well as relevant linguistics research.

4. What machinery do you need to solve your proposed problem? This is probably the most important point to keep in mind. You are free to use any techniques and tools you like; think about whether you're going to use off-the-shelf toolkits, open source code, write your code from scratch, or some combination.
5. For Computational Research projects, how will you evaluate the performance of your program? Are there gold standard annotations available? If there is prior work on the topic, will you be able to do an apples-to-apples comparison of your results to theirs?
6. For Software Application projects, where do you see the program being deployed? What is your target audience?

### 4.3 Resources

Please let me know if your project is sufficiently large to require cloud storage and/or high-powered computing resources, as I can provide access to both.

In addition to the resources in the course data directory, here are some other interesting NLP datasets:

- [Kaggle Datasets](#)
- [WikipediaXML](#)
- Sequence Tagging: [Named Entity Recognition](#) and [Chunking](#)
- [Dependency Parsing](#)
- [Quora Question Pairs](#)
- [Sentence-Level Sentiment Analysis](#)
- [Textual Entailment](#)
- [Machine Translation \(Ambitious\)](#)
- [Yelp Reviews](#)
- [WikiText Language Modeling](#)
- [Fake News Challenge](#)
- [Toxic Comment Classification](#)

There are also many test, development, and training datasets used in NLP shared tasks in SemEval (see links above) as well as for biological entities (google for BioNLP), and independent tasks like [WePS](#) (searching for entities on the Web).

## 5 Final submission

### 5.1 For Computational Research, Data Analysis, and Linguistic Annotation Projects

Your final project submission, due May 22, should include an 8-10 page (including figures but not including references) paper. The sections of the paper should include:

1. A description of your problem and motivations.
2. A reasonably thorough overview of existing work.
3. A description of your data, algorithms and methods.
4. The results of your experiments.
5. Analysis of any shortcomings of your work, and ideas for future research.

Your project will be evaluated on your results as well as thoroughness, technical depth, insight, creativity, and amount of work. Supporting data and code must be submitted in GitHub as well, but code will not be evaluated, unless your project is partly a software application.

### 5.2 For Software Application Projects

Your final submission will be a working program, uploaded to GitHub. Include documentation. In addition, submit a 4-6 page (including figures but not including references) write-up detailing:

1. A description of the motivating problem.
2. A brief survey of related research, citations to the algorithms you implement, and a description of similar existing products.
3. Analysis of any shortcomings of your program, and ideas for future development.

You will be evaluated on the idea, scope, functionality and utility of your program. Code style (documentation and readability) will also be evaluated, but to a much lesser extent.