Step 1: Choose an NLP problem, not restricted to the following:

- reconstruction (auto-correct)
- document classification
  - plagiarism detection
  - author identification
  - sentiment analysis
  - . . .
- token classification
  - part-of-speech tagging
  - named entity recognition
  - word sense disambiguation
  - . . .
- language modeling (auto-complete)
- machine translation
- . . .

Step 2: Identify or construct a solution based on a *generative* probabilistic (language) model. Describe the model in detail and develop a solution using parameter inference (and/or decoding).

Step 3: Identify or construct a solution based on a (discriminative) neural network. Describe the network structure in detail and develop a solution using gradient descent.

Step 4: Train and apply both approaches to “real” data acquired legally. Evaluate the results qualitatively and quantitatively. Highlight situations where each approach performs well and poorly. Any unusual/unexpected results require explanation (and frankly, probably debugging).

Step 5: Train and apply both approaches to synthetic data that you generate according to the generative model from Step 2. Evaluate the results qualitatively and quantitatively. Highlight situations where each approach performs well and poorly.

Step 6: Discuss pros and cons of the two approaches. Consider:

- quality/correctness
- data, time, and computational requirements
- interpretability
- . . .

Your descriptions of methods should be sufficient for anyone who has taken this course, e.g. your classmates. Note that this means you do not need to repeat description of methods presented in class.

You may work in groups of up to 3. Please submit your report in PDF form along with your code as a ZIP file. Include with your code a README file if it requires any special setup, e.g. extra Python packages or datasets.