Natural Language Processing: Syllabus

Alan W. Black & David R. Mortensen
Carnegie Mellon University

Fall 2018

Instructors:	Prof. Alan W Black (`awb@cs.cmu.edu`) and David R. Mortensen (`dmortens@cs.cmu.edu`)
Teaching assistants:	Fatima Al-Raisi (`fraisi@andrew.cmu.edu`), Manisha Chaurasia (`mchauras@andrew.cmu.edu`), Pooja Chitkara (`pchitkar@andrew.cmu.edu`), Sarveshwaran Dhansekar (`sarveshd@andrew.cmu.edu`)
Lecture time:	Tuesdays & Thursdays, 3:00–4:20
Location:	WEH 4623
Web page:	`http://demo.clab.cs.cmu.edu/NLP/`
Faculty office hours:	By appointment (Black); By appointment at `https://davidmortensen.youcanbook.me` (Mortensen)
TA Office hours:	TBA

1 Summary

This course is about a variety of ways to represent human languages (like English and Chinese) as computational systems, and how to exploit those representations to write programs that do useful things with text and speech data, like translation, summarization, extracting information, question answering, natural interfaces to databases, and conversational agents.

This field is called Natural Language Processing or Computational Linguistics, and it is extremely multidisciplinary. This course will therefore include some ideas central to Machine Learning (discrete classification, probability models) and to Linguistics (morphology, syntax, semantics).

We'll cover computational treatments of words, sounds, sentences, meanings, and conversations. We'll see how probabilities and real-world text data can help. We'll see how different levels interact in state-of-the-art approaches to applications like translation and information extraction.

From a software engineering perspective, there will be an emphasis on rapid prototyping, a useful skill in many other areas of Computer Science. In particular, we will introduce some high-level formalisms (e.g., regular expressions) and tools (e.g., Python) that can greatly simplify prototype implementation.

2 Target

The course is designed for SCS undergraduate students, and also to students in graduate programs who have a peripheral interest in natural language, or linguistics students who know how to program. Prerequisite: Fundamental Data Structures and Algorithms (15-211) or equivalent; strong programming capabilities.

3 Evaluation

Students will be evaluated in five ways:

Exams (40%) one in-class midterm on **March** (20%) and one cumulative final exam (20%), date TBD.

Project (30%) a semester-long 4-person team project (see below).

Homework assignments (20%) 7 pencil-and-paper or small programming problems given roughly weekly.

Quizzes (10%) 10 Canvas quizzes given at the beginning of many lectures[1].

The lowest 2 homework grades and the lowest 3 quiz grades will be dropped.

Late Policy No work will be accepted late. The grading policy for pop quizzes and homework assignments permits some slack of an administratively simpler kind than deducting points for lateness or missing a lecture.

Academic Honesty Exams and pop quizzes are to be completed individually. Verbal collaboration on homework assignments is acceptable, but (a) you must not share any code or other written material, (b) everything you turn in must be your own work, and (c) you must note the names of *anyone* you collaborated with on each problem (the *only* exceptions are the instructor and TA), and the nature of the collaboration (e.g., "X helped me," "I helped X," "X and I worked it out together."). If you find material in published literature (e.g., on the Web) that is helpful in solving a problem, you must cite it and explain the answer in your own words. The project is to be completed by a team; you are not permitted to discuss any aspect of your project with anyone other than your team members, the instructor, and the TA. You are encouraged to use existing NLP components in your project; you must acknowledge these appropriately in the documentation.

Suspected violations of these rules will be handled in accordance with the CMU guidelines on collaboration and cheating (`http://www.cmu.edu/policies/documents/Cheating.html`).

4 Project

A major component will be a 4-person team project. The project involves two parts:

- a **questioning** program (`ask`) whose input is a web page P and whose output is a set of questions about the content in P that a human could answer if she read P, and
- a **answering** program (`answer`) whose input is a web page P and a question Q about P and whose output is an intelligent answer A.

[1] Students should bring a device to class so they can acces Canvas.

Projects will be pitted against each other in a competition at the end of the course. Here's how the competition works:

1. Questions will be generated manually by students in the course (this happens early in the course as an exercise to start thinking about how to build an `ask` program). These will be rated by student judges in a blind setup, for reasonableness and difficulty.

2. Questions will be generated by each team's `ask` program. These will be rated by student judges, for reasonableness and fluency, in a blind setup.

3. Human-generated and reasonable automatically-generated questions will be provided as input to the `answer` programs, producing answers. These answers will be rated for correctness and fluency by student judges, in a blind setup.

The project will be primarily graded based on documentation your team submits describing how the programs work, and a brief video presentation at the end of the semester.

5 Textbook

The textbook for the course will be the second edition of *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, by Daniel Jurafsky and James H. Martin. The course will cover roughly sections I, III, IV, and parts of V.

6 Lectures

The lecture plan is subject to change. Readings from Jurafsky and Martin are given in brackets.

- Course overview; what does it mean to "know language"? [1]

(Part 1 is "shallow" NLP.)

- Words, morphology, and lexicons‡ [3.1, 3.9]
- Discussion of the project
- Information extraction, question answering, and NLP in information retrieval† [22.0, 23.0, 22.1–2, 23.1–2]
- Probability and language models [4.0–2]
- Language model evaluation and smoothing [4.3–8]
- Noisy channel models, edit distance, and spelling correction [3.10–11, 5.9]
- Classification*
- Word categories and parts of speech [5.0–3]
- Hidden Markov models and part-of-speech tagging [6.0–4]

- Chomsky hierarchy and natural language‡ [16]

(There will be an in-class midterm in the week before spring break. Part 2 is “deep” NLP.)

- Syntactic representations of natural language‡ [12.0–3]
- Parsing algorithms [13]
- Treebanks and parsing evaluation [12.4, 14.7]
- Probabilistic context-free grammars and statistical parsing [14.0–4]
- Word Embeddings and Dense Word Vectors
- Beyond context-free parsing‡
- Lexical semantics‡ [19.0–3]
- Semantic disamiguation problems: word-sense and coreference [20.0-2, 21.3, 21.7]
- Semantic role labeling [20.9]
- Compositional semantics‡ [17.2–3, 18.0–3]
- Clustering and Expectation Maximization*
- Machine translation† [25.0–1, 25.9]

(The classes closes by synthesizing and looking forward.)

- Current NLP research at CMU
- Wrap-up and discussion

*These lectures are essentially stand-alone lectures on important topics in *machine learning*, a subfield of CS that is central to current NLP.
†These lectures focus on *applications* that companies you’ve heard of are currently working on.
‡These lectures explore ideas from *linguistics*, but with a computational spin.