Computer Science 181 – Spring, 2008

Instructor & Texts

Instructor: **Kim Bruce**
222 Edmunds, x7-1866
kim@cs.pomona.edu
Office Hours: M 10:00-10:50 a.m., TTh 2:45 - 3:45 p.m., W 1:30 - 3:00 p.m., & by appointment.

Lectures: TTh 1:15 - 2:30 p.m., Lincoln 1135

Texts: *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 2nd Edition* by Daniel Jurafsky & James H. Martin, Prentice Hall, draft, and *Natural Language Processing in Python* by Steven Bird, Ewan Klein, & Edward Loper.

Course web page: http://www.cs.pomona.edu/classes/cs181NLP/

Instructor's web page: http://www.cs.pomona.edu/~kim/

Prerequisites: CSC081

Overview

This course is designed to introduce students to the fundamental concepts and ideas in natural language processing, sometimes called computational linguistics. The goals of the field include creating computer programs that can understand, generate, and even learn natural languages. Applications range from text translation and understanding to enabling humans to converse with robots. We will study language processing starting from the word level to syntactic structure to the semantic meaning of text. Approaches include statistical as well as symbolic using logic and the lambda calculus. Students will build and modify systems and will use large existing corpora for validating their systems.

This course will involve extensive reading on your part, both in the (on-line) texts and in outside sources. There will be regular assignments (both programming and paper and pencil). Each student will be expected to do a final project that examines a topic in natural language processing. This project will normally include programs, analysis of resulting data, a written report, and an oral report in the last two weeks of the semester.

Lectures and Readings

The schedule on the following two pages shows the tentative schedule of topics to be covered at each class meeting during the first half of the semester. Consult the on-line version of the course syllabus

(see URL above) regularly to see the most current version of the schedule of topics and readings. The on-line version of this schedule will be revised as the semester progresses.

I expect you to do the reading for a class before the lecture. I will not attempt to cover in lecture all the material in the readings. Instead my goal will be to cover the highlights or particularly difficult material. For this to work, you will need to already be familiar with the simpler aspects of the material. If you keep up your part of the bargain we should be able to have more interesting discussions in class, rather than just listening to me go over the text.

In the table below, JM stands for Jurafsky & Martin, while B stands for Bird *et al.*

Lecture	Date	Topic	Reading
1.	Jan. 22	Introduction	JM 1
2.	Jan. 24	Regular Expressions, Automata, & Python	JM 2, Python reg exp tutorial, B 1, 2, 3.1-3.2, App
3.	Jan. 29	Morphology & Transducers	JM 3.1-3.9, Bird 3.3
4.	Jan. 31	String Edit Distance	JM 3.10-3.11
5.	Feb. 5	Probability & N-grams	JM 4.1-4.4
6.	Feb. 7	N-grams & PoS Tagging	JM 4.5-4.7 (*not 4.7.1 or beyond*)
7.	Feb. 12	Hidden Markov Models & Viterbie Algorithm	JM 5, Bird 4
8.	Feb. 14	Tagging & Hidden Markov Models	JM 6.1-6.4, Bird 4
9.	Feb. 19	Formal Grammars for English, Smoothing code	JM 12, 13.1-3
10.	Feb. 21	CYK and Earley Parsing	JM 13.4
11.	Feb. 26	Statistical Parsing	JM 14
12.	Feb. 28	Features & Unification	JM 16
13.	March 4	Features and Unification	JM 16, Bird 11
14.	March 6	Semantics	JM 17
15.	March 11	Semantics	JM 17
16.	March 13	**No Class**	
	March 17-21	**Spring Break**	

Lecture	Date	Topic	Reading
17.	March 25	Computational semantics	JM 18.1-18.3, Bird 12.1-12.6
18.	March 27	Computational semantics	JM 18.4-18.8, Bird 12.7-12.11
19.	April 1	Lexical Semantics	JM 19.1-19.3
20.	April 3	Lexical semantics	JM 19.4-19.7
21.	April 8	Computational Lexical Semantics	JM 20.1-20.4
22.	April 10	Computational Lexical Semantics	JM 20.5-20.9
23.	April 15	Computational Discourse	JM 21
24.	April 17	Question Answering & Summarization	JM 23
25.	April 22	Dialog	JM 24
26.	April 24	Dialog	JM 24
27.	April 29	Student Presentations	
28.	May 1	Student Presentations	
29.	May 6	Student Presentations	

Homework and Programming Assignments

Programs Programs for this course will be run on the Pomona College Computer Science department's lab facilities, based in Edmunds ???. You are welcome to use other computers to write and test your programs, but they must run on our facilities. You may log in remotely to any of the lab machines using ssh. Please do not log into any of our servers (e.g., linus) to do homework.

Turning in Homework Programs will generally be due at the beginning of class periods. Please hand in paper copies of all solutions and the answers generated. Runnable copies of programs for homework should also be copied into directory `/common/cs/cs181/dropbox` on these computers. Files should be labelled with your name and the assignment number. If you have more than one file for an assignment, please put all associated files in a directory and turn in the directory. You can either drag your file or directory into `dropbox` or use sftp if you are running remotely. *You are responsible for making sure that your program, as turned in, will run successfully without any extra work on my part. Include instructions on how to run your programs either at the top of the program file or in a separate README.*

An important criterion in grading homework will be clarity of solution. Thus you should attempt

to explain your solutions, program or not, as clearly as possible. This also means that programs should be carefully documented so that I can understand them. At a minimum, each function defined should include a comment on what it does. The comment should explain what each input parameter stands for and how the output depends on the input.

Late policy Each student may use a maximum of three late days during the course of the semester (note that weekend days count), though at most two days may be taken for any one assignment. Once those late days are used up, late homework will not be accepted.

Exams and Grading

I am tentatively planning to have a take-home midterm but no final exam. The midterm will be handed out after spring break. Student grades will be determined as follows: Midterm: 25%, Final Project: 35%, Homework and programs: 30%, in-class participation: 10%.

Collaboration & Academic Honesty Policy

I highly encourage students to get together in small groups to go over material from the lectures and text, work problems from the text, study for exams, and to discuss the general ideas and approaches to material in the course.

Homework (including programs) may be done either by yourself or with one other person. If you decide to work with another person then you should only turn in one solution. Put both students' names on the assignment. If you collaborate then you must include a statement describing the contributions of each collaborator. For example, it might be *"We used pair programming for all aspects of this assignment. X took the lead on documenting the program, while Y was primarily responsible for the accompanying write-up."* or *"W wrote the parser, while Y wrote the lexical scanner. We each looked over the other's code in order to improve its clarity and efficiency.* If you collaborate with someone, then you will both generally receive the same grade on the assignment. However, I reserve the right to assign different grades if there is evidence that one person contributed significantly more to the solutions. Both parties to the collaboration are responsible for understanding all parts of the solution.

Aside from this explicitly claimed collaboration, all work to be turned in, including programming assignments, must be done independently. As explained in the student handbook, this means that the work you turn in must represent only your (or your team's) own work. It must not be based on help from others or information obtained from sources other than those approved by the instructor (e.g., the text, web pages linked from the course web page, and materials provided in lecture). Effective learning is compromised when this is not the case.

Accordingly, you should never read or copy another student's code or solutions, exchange computer files, or share your code or solutions with anyone else in the class until after the assignment is due. However, students may collaborate or receive help from each other on an occasional basis as long as all parties contributing are given explicit credit for their contributions to the homework. I will inform students if I believe they are collaborating too much. Uncredited collaborations will be considered a violation of college policies and will be handled appropriately. If there is any doubt about whether a collaboration is legal, you should cite it or ask me. I will let you know if it is within the rules.

An important exception to the above rules has to do with learning to program in Python. Students are explicitly allowed to help each other with difficulties in setting up, running compilers or using libraries for Python programs, and to help explain error messages. However, any collaboration beyond that point should be cited as explained in the prior paragraph.

Failure to abide by these rules is considered plagiarism, and will result in severe penalties. The first offense typically results in failure in the course and referral to the appropriate college office or committee. See the Academic Honesty Policy in the Student Handbook for further information. Please do not put me, yourself, or anyone else in this unpleasant situation.