Computational Linguistics (Syllabus)

Course	Info
Course#	LIN 220/425
Time	MF 1:00-2:20pm
Location	SBS S-218
Website	on CoCalc
Instructor	Thomas Graf
Email	lin220@thomasgraf.net
Office hours	tbd
Office	SBS N-249

1 Bulletin description

An introduction to computational linguistics for students with previous programming experience. This course explores the models, algorithms, and techniques that dominate modern-day language technology, and it evaluates them from a linguistically informed perspective. Topics include corpus-based methods, finite-state approaches, machine learning, and model evaluation techniques. Great emphasis is put on discussing the limitations of existing techniques and how they might benefit from linguistic insights. Students will also hone their programming skills and develop familiarity with state-of-the-art software packages for computational linguistics.

2 Teaching goals

- expand existing programming skills
- master essential concepts and techniques in computational linguistics
- critically evaluate computational models from a linguistically informed perspective
- translate abstract computational models into fully functional source code

- develop learning autonomy and the ability to deepen your programming knowledge through self-study
- Students must use the skills expected from their Versatility courses to study and practice them in greater depth, with further study applied to the area in which they are certified.

3 Prerequisites

3.1 Formal prerequisites

C or better in at least one of the following courses:

- LIN 120
- CSE 110
- CSE 114
- ISE 108

Students who do not meet this requirement may be allowed to enroll with permission of the instructor.

3.2 What's required?

This course assumes that you have already mastered essential programming concepts, including

- variables
- basic data types (strings, integers, arrays/lists)
- control flow (`if-else`)
- loops (`while`, `for`)
- custom functions

Even though the course uses Python as its programming language, you do not need to have prior experience with that specific language. The basic programming techniques are virtually identical across all popular languages: Python, Java, C, Perl, Ruby, Haskell, and so on.

3.3 What's NOT required?

You do **not** need any of the following:

1. Experience with intermediate or advanced programming techniques, e.g. hash maps, recursive functions, or object oriented programming
2. Programming experience with Python; the first two weeks of the course give a review of all the Python fundamentals you need to succeed
3. Knowledge of advanced mathematics, e.g. calculus or linear algebra

4. A general linguistics background; students who have taken LIN 101 or some other introductory course will recognize many concepts, but we discuss them from a very different perspective.
5. Having taken LIN 120; while this course continues LIN 120, it is also suitable for students who have previously taken a programming class on campus.

3.4 Resources

We will be using *Cocalc* for this class, a cloud-based service for collaborative programming and data science. All class materials will be distributed on Cocalc, including slides and homework. Your assignments will also be automatically collected and graded on Cocalc after the deadline.

All you need for Cocalc is a device with a recent browser (Windows, OS X, or Linux computer, Chromebook, Android tablet; Safari does not work well, so iOS devices may run into some issues). While Cocalc can be used for free, we will upgrade to a premium tier for this class, which costs **$14 per semester**.

4 Grading

This course can only be taken for 3 credits. Student grades are determined by the following components:

1. **Weekly homework (35%)**
   Students have to solve weekly programming assignments in Python. One homework may take the form of

   - several minor practice exercises, or
   - a single, more open ended coding task (e.g. implementing a simple part-of-speech tagger for English).

   Assignments are unlocked each Wednesday at 11:59pm on Cocalc and, unless stated otherwise, are due the following Wednesday at 11:59pm. Each homework is graded on a P/F basis (P = 1 point, F = 0 points) depending on whether is passes specific *unit tests*, which will be shared with students in advance.

2. **Project (20%)**
   After spring break, students have to form groups of 2 or 3 and pick one programming project from a curated list. Each group must collaborate on this project for the rest of the semester. Projects require students to apply the theoretical concepts they have learned to a concrete problem and implement a solution in Python. The completed project is due by the end of week 13 (Friday). Each project receives between 0 and 5 points, based on specific criteria that will be explained in a separate rubric.

3. **Project presentation (20%)**
   Each group has to give a brief ($\sim$ 15 min) presentation of their project in class during

week 14 or week 15. The presentation should cover design principles, what problems were encountered and how they were addressed, as well as a short demo session. Each presentation receives between 0 and 5 points, based on specific criteria that will be explained in a separate rubric.

4. **Class participation (25%)**
   Class participation is highly encouraged and can take various forms:

   - asking questions in class
   - participating during in-class discussions
   - pointing out problems with the lecture notes on Cocalc (e.g. typos, confusing wording, broken code)
   - posting in the course's Help and Support forum on Cocalc (this includes both asking questions of your own and answering fellow students' questions)
   - suggesting topics for lectures
   - posting links to relevant online material (e.g. tutorials or news paper articles)
   - participating in optional surveys

5 Schedule by week

The weekly schedule consists of two lectures. There is no recitation. Friday lectures focus on theory and general discussion, whereas Monday lectures cover specific ways of implementing the discussed models in Python. So when reading the overview below, keep in mind that Friday of week n belongs to Monday of week $n+1$, not week n.

The approximate schedule for topics and assignments is as follows.

- Homework is assigned each Wednesday and due a week later.
- In week 9, students form groups of 2 or 3 and pick a topic for the final project.
- The final project is due the Friday of week 13.
- Student presentations will be given during the last 2 weeks of the semester.

Week	Monday	Friday
1	syllabus & Python recap	big picture
2	Python recap	n-grams, word prediction
3	units tests with hypothesis, zip, counters	corpora and model evaluation
4	working with corpora, regular expressions	spellcheckers, edit distance
5	dynamic programming	tagging, stemming, lemmatization
6	nltk taggers, stemmers, lemmatizers	nothing scheduled (buffer)
7	nothing scheduled (buffer)	interim summary
8	Spring break	
9	Project discussion	finite-state morphology
10	object oriented programming	hand-written models VS machine learning
11	scikit learn	syntactic analysis, parsing
12	data structures for trees	semantics (lexical & compositional)

Week	Monday	Friday
13	word2vec in Tensorflow	machine translation
14	seq2seq in Tensorflow	nothing scheduled (buffer)
15	nothing scheduled (buffer)	summary & final discussion

6 Policies

6.1 Contacting me

- Emails should be sent to lin220@thomasgraf.net Disregarding this policy means late replies and is a sure-fire way to get on my bad side.
- Reply time < 24h in simple cases, possibly more if meddling with bureaucracy is involved.
- If you want to come to my office hours and anticipate a longer meeting, please email me so that we can set apart enough time and avoid collisions with other students.

6.2 Disability Support Services

If you have a physical, psychological, medical or learning disability that may impact your course work, please contact Student Accessibility Support Center, ECC (Educational Communications Center) Building, Room 128, (631)632-6748. They will determine with you what accommodations, if any, are necessary and appropriate. All information and documentation is confidential.

Students who require assistance during emergency evacuation are encouraged to discuss their needs with their professors and Student Accessibility Support Center. For procedures and information go to the following website: http://www.stonybrook.edu/ehs/fire/disabilities.

6.3 Academic Integrity

Each student must pursue his or her academic goals honestly and be personally accountable for all submitted work. Representing another person's work as your own is always wrong. Faculty is required to report any suspected instances of academic dishonesty to the Academic Judiciary. Faculty in the Health Sciences Center (School of Health Technology & Management, Nursing, Social Welfare, Dental Medicine) and School of Medicine are required to follow their school-specific procedures. For more comprehensive information on academic integrity, including categories of academic dishonesty please refer to the academic judiciary website at http://www.stonybrook.edu/commcms/academic_integrity/index.html.

6.4 Critical Incident Management

Stony Brook University expects students to respect the rights, privileges, and property of other people. Faculty are required to report to the Office of University Community Standards any disruptive behavior that interrupts their ability to teach, compromises the safety of the learning environment, or inhibits students' ability to learn. Faculty in the HSC Schools and the School of Medicine are required to follow their school-specific procedures. Further information about most academic matters can be found in the Undergraduate Bulletin, the Undergraduate Class Schedule, and the Faculty-Employee Handbook.