Course Missive

Spring 2017

Time and Location: MWF 2:00-2:50 PM, Smith-Buonano Hall 106

Course Staff

What	Who	Where	When
Professor	Eugene Charniak (`ec`)	CIT 419	Hours by appointment
Head TA	Sidd Karamcheti (`skaramch`)	TBA	TBA
Undergraduate TAs	Arun Drelich (`adrelich`)	TBA	TBA
	Beverly Tai (`btai`)	TBA	TBA
	Nick McKenna (`nmckenna`)	TBA	TBA
	Raphael Kargon (`rkargon`)	TBA	TBA

Introduction

Welcome to CS146! Have you ever wondered how a computer program can complete such amazing tasks as:

- distinguishing good English from bad,
- translating English texts into exotic foreign tongues,
- identifying the subject, verb, and object of a sentence, and
- grouping articles by their topic,

all with little or no human instruction? Do you want to take the first step on the road to becoming Professor Eugene Charniak? If so, this is the course for you!

This course satisfies the AI requirement for the Sc.B. track in Computer Science.

Prerequisites

There are no formal prerequisites for CS146, but students should at least have a solid background in basic programming (equivalent to CS15/16, CS17/18, or CS19). Knowledge of basic probability and taking CS141 are also helpful.

Assignments

There are five programming projects of moderate difficulty, plus a small warmup project. There will also be five written homework assignments, which will provide more practice with the topics

covered in class. The exercises for the written homeworks and the assignments for the projects can be found at the end of each chapter of the textbook. These **must** be typed. We will not accept handwritten assignments.

The programming projects together will make up 75% of your course grade; the homeworks will make up the remaining 25%. Each homework, thus, will be worth 5% of your final grade, and each project (not counting the warmup project) will be worth 15% thereof. **Note, however, that you must complete all of the programming projects to receive credit for this course.**

Each of the programming projects and homeworks is due by midnight on the date listed on the handout — thus, you have until the end of the day listed as the due date to complete the assignment. You will have seven (7) free late days to use towards all but the last project. After your late days are expended you will lose 10% of your project grade for each extra day your handin is delayed. At the end of the term, we will calculate how to best divvy out your late days to best help your final grade. There is no need for you to do anything special for this, other than keep track of submission dates for your own records. **Projects handed in after 11:59pm on Tuesday, May 10th will not be accepted.** Late homeworks will not be accepted without a legitimate excuse, and you may not use your late days on homework assignments.

Additionally, if you want a TA to regrade an assignment, you need to ask that TA within a month of recieving your assignment back. The exception to this is the last project; regrades most be asked for within a week.

Coding

Since this is not a software engineering course, we won't be enforcing stringent style guidelines, but you should write so that someone who isn't a wizard with your langauge of choice will be able to undersand what your program is doing (add plenty of comments, break up code into smaller functions, i.e. apply basic common sense). If you turn in a partially-functional assignment and we can't tell what you were trying to do, we'll probably be very grumpy about giving partial credit.

As that translates to an official policy, so long as your code produces the expected output(s) and adheres to any specific project restrictions (runtime, etc.) then you will not lose points for poor design or coding practices. *However*, as this is not a software design course, it is not the responsibility of the TAs to attempt to understand the intentions underlying confusing code. If it is not fully clear what you were trying to do in the implementation of a paritally-functinoal assignment (i.e. not all of the output is as expected) then partial-credit will be given sparingly, and at our discretion.

Theoretically, you may write the programming assignments in whatever language you choose, so long as the code is able to be executed on department machines. However, keep in mind two points. One, you will be working with fairly large data sets, so you shouldn't use any language that has a high computational overload. Python probably represents the bearable upper limit in that regard; Perl would certainly be too slow. Two, as mentioned above the TA staff needs to be able to read and well understand your code in order to assign *any* partial-credit. If you submit an assignment in a language that the TA staff is not well versed in, then you might not be able to receive partial-credit for errors. Our current TA staff is well versed in Java, Python, and C. **You are allowed to write programming assignments in whatever language you chose, but we strongly recommend using either Java or Python.**

We will provide you with template shell scripts which will invoke your program on arbitrary input files. You will have to fill these in - this a trivial task but *don't forget to do so*, as this is the means by which we will be testing your handins. Failure to fill in these scripts will cause you to lose 15% of your grade on that project, even if you provide full instructions for running your code.

Note: The deep learning portions of this class will involve performing matrix operations on data (primarily matrix multiplication). While it wouldn't be too tricky to implement those matrix operations on your own, We recommend you use a matrix manipulation library for the language of your choice.

Here are some recommended matrix manipulation libraries for different programming languages:

- Java -
- Python - NumPy / Tensorflow

Collaboration Policy

Discussion of material with your classmates is both permitted and encouraged. However, **showing, copying or other sharing of actual code is forbidden**. This **will** be enforced. Furthermore, no collaboration is allowed on the homework assignments.

Course Schedule

Day	Topic	Out	Due
Wed. Jan 25th	Introduction	`warmup`	
Fri. Jan 27th	Language Modeling (Introduction)	`langmod`, `hw1`	
Mon. Jan 30th	Language Modeling		`warmup`
Wed. Feb 1st	Language Modeling (Conclusion)		
Fri. Feb 3rd	Machine Translation (Introduction)	`mt`, `hw2`	`hw1`
Mon. Feb 6th	Machine Translation (EM Algorithm)		`langmod`
Wed. Feb 8th	Machine Translation (EM Algorithm)		
Fri. Feb 10th	Machine Translation		
Mon. Feb 13th	Machine Translation		
Wed. Feb 15th	Machine Translation		
Fri. Feb 17th	Machine Translation (Conclusion)		
	Long Weekend		
Wed. Feb 22nd	Part-of-speech Tagging (Introduction)	`pos`, `hw3`	`hw2`
Fri. Feb 24th	Part-of-speech Tagging		
Mon. Feb 27th	Part-of-speech Tagging		`mt`
Wed. Mar 1st	Part-of-speech Tagging		
Fri. Mar 3rd	Part-of-speech Tagging		
Mon. Mar 6th	Part-of-speech Tagging (Conclusion)		
Wed. Mar 8th	Parsing (Introduction)	`parsing`, `hw4`	`hw3`
Fri. Mar 10th	Parsing		
Mon. Mar 13th	Parsing		`pos`
Wed. Mar 15th	Parsing		
Fri. Mar 17th	Parsing		
Mon. Mar 20th	Parsing (Conclusion)		
Wed. Mar 22nd	Topic Modeling (Introduction)	`topicmod`, `hw5`	`hw4`
Fri. Mar 24th	Topic Modeling		
	Spring Break		
Mon. Apr 3rd	Topic Modeling (Conclusion)		`parsing`
Wed. Apr 5th	Deep Language Modeling (Introduction)	`deep_langmod`	`hw5`
Fri. Apr 7th	Deep Language Modeling		
Mon. Apr 10th	Deep Language Modeling		`topicmod`
Wed. Apr 12th	Deep Language Modeling		
Fri. Apr 14th	Deep Language Modeling (Conclusion)		
Mon. Apr 17th	Deep Parsing (Introduction)	`deep_parsing`	`deep_langmod`
Wed. Apr 19th	Deep Parsing		
Fri. Apr 21st	Deep Parsing		
Mon. Apr 24th	Deep Parsing (Conclusion)		
Wed. Apr 26th	SLACK DAY		
Fri. Apr 28th	SLACK DAY		`deep_parsing`
	Reading Period		
Tue. May 10th	**Absolute Deadline**		