1 Course Description

Natural language processing (NLP) is a subfield of AI focused on solving problems that involve dealing with human language in a sophisticated way: these include information extraction, machine translation, automatic summarization, conversational dialogue, syntactic analysis, and many others. Much of the progress on these problems over the last 25 years has been driven by statistical machine learning and, more recently, deep learning. One distinctive feature of language compared to other types of data is its structured nature: modeling language involves understanding the linguistic phenomena it exhibits and grappling with it as a sequentially-structured, tree-structured, or graph-structured entity.

This class is intended to be a survey of modern NLP in two respects. First, it covers the main applications of NLP techniques today, both in academia and in industry, as well as enough linguistics to put these problems in context and understand their challenges. Second, it covers a range of models in structured prediction and deep learning including classifiers, sequence models, statistical parsers, neural network encoders, encoder-decoder models, convolutional neural networks, and more. We study the models themselves, examples of problems they are applied to, inference methods, parameter estimation (both supervised and unsupervised approaches), and optimization. Programming assignments involve building scalable machine learning systems for various NLP tasks and seeing how these models can be put into practice.

Overlap with CS395T: Structured Models for NLP: This class will have roughly 70% overlap with the CS395T: Structured Models for NLP taught in Fall 2017. It is not recommended that students take both classes.

2 Prerequisites

- 391L - Machine Learning, 343 - Artificial Intelligence, or equivalent AI/ML course experience
- Familiarity with Python (for programming assignments)
- Additional prior exposure to probability, linear algebra, optimization, linguistics, and NLP useful but not required

3 Lectures

Lectures are 9:30-11:00am Tuesday and Thursday in GDC 4.304. A complete schedule of lectures and assignments, complete with readings, is on the course website:

There is no required textbook for this course. Readings from book chapters and papers will be posted on the course website.
4 Assignments

There are five assignments in the course: two “mini” assignments, two projects, and a final project.

- Mini 1: 10 points
- Project 1: 20 points
- Mini 2: 10 points
- Project 2: 20 points
- Final project: 40 points
- Total: 100 points

The timeline of these assignments is on the course calendar. Assignment specifications, code, and data will be made available on the course website and Canvas.

4.1 Mini

The mini assignments are designed to be relatively straightforward programming assignments. In each one, you will implement a simple system and run it on some data. The main goal is to gain familiarity with the techniques you’ll be using in the following project and get accustomed to coding up ML systems that perform well.

Grading 10 points total: 8 points for code/results and 2 points for a minimal writeup that describes what you did and gives results. Code performance requirements for getting full credit will be described in each assignment. The writeup should include a table/graph of your results and any accompanying description necessary to understand that table. For example, if you’re comparing two different classification techniques, at least briefly specify what features the models consider, what optimization techniques you used, whether you used regularization, etc.

4.2 Projects

The projects are more substantial programming assignments. Each project centers around an NLP task on a standard dataset, with part of the project being an open-ended extension where you’ll have options for exactly what you want to implement or explore further.

Grading 20 points total: 12 points for code/results, 4 points for writeup, 4 points for the extension. Code performance requirements for getting full credit will be described in each assignment. Getting full credit on the extension requires going above and beyond: your extension should really demonstrate some improvement, or you should have some particularly insightful analysis. See the website for examples of projects with successful extensions.

Writeup Your project writeup should be 2-3 pages (excluding references, if you have any). Your report should restate the core problem and what you are doing, describe relevant details of your implementation, present results, describe your extension, and optionally discuss error cases addressed by your extension or describe how the system could be further improved. Your report should be written in the tone and style of an ACL/NIPS conference paper. Any format with reasonably small (≈1” margins) is fine, including the ACL style files\(^1\) or any one- or two-column format with similar density.

\(^1\)Available at http://acl2017.org/calls/papers/
4.3 Final Project

The final project is an opportunity for open-ended exploration of concepts in the course. This project should constitute novel work beyond directly implementing concepts from lecture and should result in a report that roughly reads like an NLP/ML conference submission in terms of presentation and scope. You may work on the final project either individually or in groups of two; however, groups of two are preferred from the standpoint of enabling more substantial projects.

Proposal You will write a brief proposal (<1 page) explaining your idea, which the course staff will provide feedback on.

Writeup Your final project report should be 4-8 pages—use your discretion about the length. Groups of two should have reports closer to 8 pages. The scope should be similar to that of an ACL paper: you should present a novel idea, discuss related work, describe your implementation or what you did, give results, and provide discussion or error analysis.

Presentation You will give lightning talks (around 5 minutes) about your project on the last two class days.

4.4 Final grades

Your final grade is computed based on the total points earned across all assignments. The final grade is mapped to a letter as follows, with grades on the boundary receiving the higher grade:

<table>
<thead>
<tr>
<th>Grade</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>100 − 93.3</td>
</tr>
<tr>
<td>A−</td>
<td>93.3 − 90.0</td>
</tr>
<tr>
<td>B+</td>
<td>90 − 86.6</td>
</tr>
<tr>
<td>B</td>
<td>86.6 − 83.3</td>
</tr>
<tr>
<td>B−</td>
<td>83.3 − 80</td>
</tr>
<tr>
<td>C+</td>
<td>80 − 76.6</td>
</tr>
<tr>
<td>C</td>
<td>76.6 − 73.3</td>
</tr>
<tr>
<td>C−</td>
<td>73.3 − 70</td>
</tr>
<tr>
<td>D+</td>
<td>70 − 66.6</td>
</tr>
<tr>
<td>D</td>
<td>66.6 − 63.3</td>
</tr>
<tr>
<td>D−</td>
<td>63.3 − 60.0</td>
</tr>
<tr>
<td>F</td>
<td>&lt; 60</td>
</tr>
</tbody>
</table>

4.5 Collaboration Policy

You are free to discuss the homework assignments with other students and work towards solutions together. However, all of the code you write must be your own! We will be using Moss and any copied code will be treated as a violation of academic honesty\(^2\) and may result in a failing grade. In addition, your writeup must be entirely your own and your extension cannot duplicate those of your collaborators\(^1\).

\(^2\)See more about UT CS academic honesty at https://www.cs.utexas.edu/academics/conduct
4.6 Project Submission, Slip Days, Late Assignments

Projects will be submitted on Canvas. Submissions should include your writeup, a gzipped tar or zip file of code, and any requested system output (e.g., model predictions on the blind test set).

Each student is given 5 “slip days” to use throughout the term. Any number of these days can be applied to any project excluding the final project to extend the deadline for that project. E.g., you can turn the first project in 2 days late and the fourth project 3 days late. After your slip days are exhausted, each day of lateness will incur a 15% penalty to that assignment’s grade. Plan your slip day budget accordingly, e.g., be sure to save them up if you know you’ll be traveling for a conference around a due date for a later project.

4.7 Compute Resources

The assignments are designed to be doable on personal computers (assuming you write your code efficiently!). However, for extensions and for your final project, you may wish to run longer experiments. We encourage you to do so using the department’s Condor pool. An overview of Condor can be found here: http://www.cs.utexas.edu/facilities/documentation/condor and some documentation can be found here: https://www.cs.utexas.edu/~ml/faq/mastodon_readme.html

Be aware that your jobs may be terminated by Condor if they are competing for resources and plan ahead for this if you choose to use Condor.

For the final project, the course will have an instructional allocation on TACC that you may use. More details about this forthcoming.

5 Miscellaneous

Disabilities Students with disabilities may request appropriate academic accommodations from the Division of Diversity and Community Engagement, Services for Students with Disabilities3 at 471-6259.

---

3On the web at http://ddce.utexas.edu/disability/