CSE 517: Natural Language Processing

University of Washington

Winter 2016

The syllabus is subject to change; always get the latest version from the class website.

Website:	http://courses.cs.washington.edu/courses/cse517/16wi
Lectures:	Smith 102, Mondays and Wednesdays 1:30–2:50 pm
Instructor:	Noah A. Smith (nasmith@cs.washington.edu)
Instructor office hours:	CSE 532, Fridays 1:30–2:30 pm or by appointment
Teaching assistants:	Jesse Dodge (dodgejesse@gmail.com)
	Eunsol Choi (eunsol@cs.washington.edu)
	Kenton Lee (kentonl@uw.edu)
TA office hours:	CSE 220, Mondays 11:30 am–12:30 pm or by appointment

Natural language processing (NLP) seeks to endow computers with the ability to intelligently process human language. NLP components are used in conversational agents and other systems that engage in dialogue with humans, automatic translation between human languages, automatic answering of questions using large text collections, the extraction of structured information from text, tools that help human authors, and many, many more. This course will teach you the fundamental ideas used in key NLP components. It is organized into four parts:

1. **Probabilistic language models**, which define probability distributions over text passages.
2. **Text classifiers**, which infer attributes of a piece of text by "reading" it.
3. **Analyzers**, which map texts into **linguistic representations** that in turn enable various kinds of understanding.
4. **Generators**, which produce natural language as output.

1 Course Plan

Table 1 shows the planned lectures, along with readings.

2 Evaluation

Students will be evaluated as follows:

- Approximately four assignments, completed individually (40%), due 1/20, 2/3, 2/17, and 3/2
- A project, completed in teams of 1–3 (35%), due 3/9
- An oral exam (15%), to take place at the end of the quarter
- Participation (10%)

Participation points are earned by submitting proposed oral exam questions. You are expected to submit one per week, between Monday at 1:30 pm and Friday at 5:00 pm, through the appropriate catalyst link for each week: 1/4–8; 1/11–15; the rest are through the canvas site for this course.

1/4		**introduction**		[1]
1/4–6	**probabilistic language models**	generative	$\mathcal{V}^* \rightarrow \mathcal{V}$	[2, 3]
1/11		featurized		[4] §2, 7.4
1/13		neural		[5] §0–4, 10–13
1/20		cotext: topic models		[6] §1–4
1/25		cotext and bitext		[7]
1/27	**text classifiers**	methods & applications	$\mathcal{V}^* \rightarrow \mathcal{L}$	[8, 9]
2/1		(continued)		
2/1	**linguistic representations and analyzers**	methods for sequences	$\mathcal{V}^* \rightarrow \mathcal{L}^*$	[10]
2/3		parts of speech		[11]
2/8		supersenses, entities, chunking		[12]
2/8–10		graphical models		[13]
2/17–22		phrase-structure trees		[14]
2/24		syntactic dependencies	$\mathcal{V}^* \rightarrow \mathcal{Y}$	[15]
2/29		semantic roles and relations		[16]
3/2		logical forms		[17]
3/7	**text generators**	translation between languages, summarization	$\mathcal{V}^* \rightarrow \bar{\mathcal{V}}^*$	[18]

Table 1: Course structure and lecture topics. Blue links are to lecture slides, and green links are to references. In this notation, $\mathcal{V}$ is a vocabulary of discrete symbols—most commonly words—in a (natural) language. $\mathcal{V}^*$ is a sequence of symbols of an arbitrary length. We use $\mathcal{L}$ to denote a smaller vocabulary of labels, $\mathcal{Y}$ to denote a constrained set of discrete structures (e.g., trees or directed graphs), and $\bar{\mathcal{V}}$ to denote a vocabulary possibly in another natural language.

References

[1] Julia Hirschberg and Christopher D. Manning. Advances in natural language processing. *Science*, 349 (6245):261–266, 2015. URL https://www.sciencemag.org/content/349/6245/261.full.

[2] Michael Collins. Course notes for COMS w4705: Language modeling, 2011. URL http://www.cs.columbia.edu/~mcollins/courses/nlp2011/notes/lm.pdf.

[3] Daniel Jurafsky and James H. Martin. N-grams (draft chapter), 2015. URL https://web.stanford.edu/~jurafsky/slp3/4.pdf.

[4] Michael Collins. Log-linear models, MEMMs, and CRFs, 2011. URL http://www.cs.columbia.edu/~mcollins/crf.pdf.

[5] Yoav Goldberg. A primer on neural network models for natural language processing, 2015. URL http://u.cs.biu.ac.il/~yogo/nnlp.pdf.

[6] Peter D. Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, 2010. URL https://www.jair.org/media/2934/live-2934-4846-jair.pdf.

[7] Michael Collins. Statistical machine translation: IBM models 1 and 2, 2011. URL http://www.cs.columbia.edu/~mcollins/courses/nlp2011/notes/ibm12.pdf.

[8] Daniel Jurafsky and James H. Martin. Classification: Naive Bayes, logistic regression, sentiment (draft chapter), 2015. URL https://web.stanford.edu/~jurafsky/slp3/7.pdf.

[9] Michael Collins. The naive Bayes model, maximum-likelihood estimation, and the EM algorithm, 2011. URL http://www.cs.columbia.edu/~mcollins/em.pdf.

[10] Michael Collins. Tagging with hidden Markov models, 2011. URL http://www.cs.columbia.

edu/~mcollins/courses/nlp2011/notes/hmms.pdf.
[11] Daniel Jurafsky and James H. Martin. Part-of-speech tagging (draft chapter), 2015. URL https://web.stanford.edu/~jurafsky/slp3/9.pdf.
[12] Daniel Jurafsky and James H. Martin. Information extraction (draft chapter), 2015. URL https://web.stanford.edu/~jurafsky/slp3/20.pdf.
[13] Daphne Koller, Nir Friedman, Lise Getoor, and Ben Taskar. Graphical models in a nutshell, 2007. URL http://www.seas.upenn.edu/~taskar/pubs/gms-srl07.pdf.
[14] Michael Collins. Probabilistic context-free grammars, 2011. URL http://www.cs.columbia.edu/~mcollins/courses/nlp2011/notes/pcfgs.pdf.
[15] Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT-EMNLP*, 2005. URL http://www.aclweb.org/anthology/H/H05/H05-1066.pdf.
[16] Daniel Jurafsky and James H. Martin. Semantic role labeling (draft chapter), 2015. URL https://web.stanford.edu/~jurafsky/slp3/22.pdf.
[17] Mark Steedman. A very short introduction to CCG, 1996. URL http://www.inf.ed.ac.uk/teaching/courses/nlg/readings/ccgintro.pdf.
[18] Michael Collins. Phrase-based translation models, 2013. URL http://www.cs.columbia.edu/~mcollins/pb.pdf.

3 Academic Integrity

We understand that most students would never consider cheating in any form. There is, however, a fraction of students for whom this is not the case. In the past, when we have caught students cheating, they have often insisted that they did not understand the rules and penalties. For this reason, we require that each student read this document and sign and return the second page.

- You may verbally collaborate on homework assignments. On each problem and program that you hand in, you must include the names of the people with whom you have had discussions concerning your solution. Indicate whether you gave help, received help, or worked something out together. The names should include anyone you talked with, whether or not they're taking the class, and whether or not they attend or work at UW. The only people you don't need to acknowledge are the instructor and TA(s).
- The course project allows you to work in a team. No one on your team is to discuss any aspect of your specific solution to the project with any other team or anyone not currently taking the course.
- You may get help from anyone concerning programming issues which are more general than the specific assignment (e.g., "what does a particular error message mean?").
- You may not share written work or programs (on paper, electronic, or any other form) with anyone else.
- If you find an assignment's answer, partial answer, or helpful material in published literature or on the web, you must cite it appropriately. Don't claim to have come up with an idea that wasn't originally yours; instead, explain it in your own words and make it clear where it came from.
- On the course project, you are allowed to use existing NLP tools. You must acknowledge these appropriately in all documentation, including your final report. If you aren't sure whether a tool or data resource is appropriate for use on the project, because it appears to solve a major portion of the assignment or because the license for its use is not clear to you, or if you aren't sure how to acknowledge a tool appropriately, you must speak with the course staff.

Clear examples of cheating include (but are not limited to):

- Showing a draft of a written solution to another student.
- Showing your code to another student.
- Getting help from someone or some resource that you do not acknowledge on your solution.
- Copying another someone else's solution to an assignment.
- Receiving exam related information from a student who has already taken the exam.
- Attempting to hack any part of the course infrastructure.
- Looking at someone else's work stored on disk, even if the file permissions allow it.
- Lying to the course staff.

By signing below, you indicate that you are also aware of UW's policies on student academic responsibility (https://depts.washington.edu/grading/pdf/AcademicResponsibility.pdf and the Student Conduct Code).

I, _____, have read and understood the CSE 517 course policy on cheating. I agree to honor the rules which the policy describes.

_____ (sign)

_____ (date)