CS221: Artificial Intelligence: Principles and Techniques

Stanford / Autumn 2018-2019

[Calendar] [Coursework] [Schedule]

For last year's page, click here.

Logistics

Time/location:

- Lectures: Mon/Wed 1:30-2:50pm in NVIDIA auditorium (watch online)
- Sections: Fri 1:30-2:20pm in NVIDIA auditorium; see calendar for schedule.
- Office hours: CA office hours are in the Huang basement; see calendar for times; see [Office Hour Logistics] for logistics.

Course staff:

Instructor:

Percy Liang

Guest lecturer:

Kelvin Guu

Course assistants:

Daniel Selsam

Gregory Soh

Ajay Sohmshetty

Anand Dhoot

Anvita Gupta

Benoit Zhou

Chinmayee Shah

Chuma Kabaghe

David Golub

Andrew Han

Jacob Hoffman

Jaebum Lee

Megha Srivastava

Richard Diehl Martinez

Sudarshan Seshadri

Stephanie Dong

Vivekkumar Patel

Hao Wang

William Bakst

Yilong Li

Zhangyuan Wang

Steven Diamond

Kaiyi (Alexander) Fu

Yongshang Wu

Yianni Laloudakis

Benjamin Petit

Vivian Hsu

Aaron Effron

Amita Kamath

How to contact us: Please use Piazza for all questions related to lectures, homeworks, and projects. For SCPD students, email scpdsupport@stanford.edu or call 650-741-1542.

Announcements: see Piazza. NOTE: If you enrolled in this class on Axess, you should be added to the Piazza group automatically, within a few hours. You can also register independently; there is no access code required to join the group.

Calendar: look here for dates/times of all lectures, sections, office hours, due dates.

Grades: click here to check your grades and autograder feedback.

Content

What is this course about? What do web search, speech recognition, face recognition, machine translation, autonomous driving, and automatic scheduling have in common? These are all complex real-world problems, and the goal of artificial intelligence (AI) is to tackle these with rigorous mathematical tools. In this course, you will learn the foundational principles that drive these applications and practice implementing some of these systems. Specific topics include machine learning, search, game playing, Markov decision processes, constraint satisfaction, graphical models, and logic. The main goal of the course is to equip you with the tools to tackle new AI problems you might encounter in life.

Prerequisites: This course is fast-paced and covers a lot of ground, so it is important that you have a solid foundation on both the theoretical and empirical fronts. You should have taken the following classes (or their equivalents):

- Programming (CS 106A, CS 106B, CS 107)
- Discrete math (CS 103)
- Probability (CS 109)

Reading: There is no required textbook for this class, and you should be able to learn everything from the lecture notes and homeworks. However, if you would like to pursue more advanced topics or get another perspective on the same material, here are some books:

- Russell and Norvig. Artificial Intelligence: A Modern Approach. A comprehensive reference for all the AI topics that we will cover.
- Koller and Friedman. Probabilistic Graphical Models. Covers factor graphs and Bayesian networks (this is the textbook for CS228).
- Sutton and Barto. Reinforcement Learning: An Introduction. Covers Markov decision processes and reinforcement learning. Available free online.
- Hastie, Tibshirani, and Friedman. The elements of statistical learning. Covers machine learning. Available free online.
- Tsang. Foundations of constraint satisfaction. Covers constraint satisfaction problems. Available free online.

Bear in mind that some of these books can be quite dense and use different notation terminology, so it might take some effort to connect up with the material from class.

Office Hour Logistics

CA office hours are either in the Huang basement or online. See calendar for times. In both in-person and online office hours, you will need to register an account on QueueStatus. When you wish to join the queue, click "Sign Up" at the CS 221 queue. Be sure to enter your email

when you "Sign Up"; this is a way for the CA to contact you. Look for announcements on the left panel for more information. For online office hours, you will need to install Zoom to video call with the CA. See instructions below.

SCPD (online) office hours and non-SCPD (on-campus) office hours use different queues:

- On campus office hour: http://queuestatus.com/queues/234.
- Online office hour: http://queuestatus.com/queues/253.

Installing Zoom

- **Linux**:
  - Go to downloads and download appropriate Linux package from 'Zoom Client for Linux'.
  - Open package (e.g. with 'Ubuntu Software Center' or other appropriate application) and install.
- **Windows**:
  - Go to Stanford Zoom and click 'Launch Zoom'.
  - Click 'host meeting'; nothing will launch but there will a link to 'download & run Zoom'.
  - Click on 'download & run Zoom' to download 'Zoom_launcher.exe'.
  - Run 'Zoom_launcher.exe' to install.

Coursework

Grading

- **Homeworks** (60%): There will be weekly homeworks with both written and programming parts. Each homework is centered around an application and will also deepen your understanding of the theoretical concepts. Some homeworks will have a competition component; winners will receive extra credit. Here are all the homework deadlines:

  - Foundations [foundations] (zip) (solutions) (due **Wed Oct 3**)
  - Sentiment classification [sentiment] (zip) (solutions) (due **Tue Oct 9**)
  - Text reconstruction [reconstruct] (zip) (solutions) (due **Tue Oct 16**)
  - Blackjack [blackjack] (zip) (solutions) (due **Tue Oct 23**)
  - Pac-Man [pacman] (zip) (solutions) (due **Tue Oct 30**)
  - Course scheduling [scheduling] (zip) (solutions) (due **Tue Nov 6**)
  - Car tracking [car] (zip) (solutions) (due **Tue Nov 13**)
  - Language and logic [logic] (zip) (due **Fri Dec 7**)

- **Exam** (20%): The exam is a three-hour written exam that will test your knowledge and problem-solving skills on all preceding lectures and homeworks. You cannot use any external aids except one double-sided page of notes. If you have a major conflict (e.g., an academic conference), submit a request to take it at another (earlier) time. Your request must be submitted by **Wed Oct 31**.

  Date: **Tue Nov 27** from 6pm to 9pm
  Location: Last name initial in A-L in Dinkelspiel Auditorium, M-Z in CEMEX Auditorium.

  - 2018 exam [solutions]
  - 2017 exam [solutions]
  - 2016 exam [solutions]
  - 2015 exam [solutions]
  - 2014 exam [solutions]
  - 2013 exam [solutions]

  For SCPD students, your exams will be distributed through the SCPD office once you have set up an exam monitor.

- **Project** (20%): The final project provides an opportunity for you to use the tools from class to build something interesting of your choice. Projects should be done in groups of up to three. The project will be something that you work on throughout the course and we have set up some milestones to help you along the way:

  - Project proposal [p-proposal] (due **Thu Oct 25**)
  - Project progress report [p-progress] (due **Thu Nov 15**)
  - Project poster session [p-poster] (due **Mon Dec 3**)
  - Project poster session (peer review) [p-peer] (due **Thu Dec 6**)
  - Project final report [p-final] (due **Fri Dec 14**)

  See the project page for more details.

  Regardless of the group size, all groups must submit the same basic amount of work as detailed in each milestone. Of course, the experiments may not always be successful, so we will cut the smaller groups more slack, while larger groups are expected to be more thorough in their experiments.

  For inspiration, we have made previous CS221 projects available for viewing.

  Poster Fair Information:
  - **Where:** Tressider Union's Oak Lounge (map)
  - **When:** Monday, December 3rd 2018. 1:00pm - 5:00pm
    - Session A: 1:00pm - 2:20pm
    - Session B: 2:20pm - 3:40pm
    - Session C: 3:40pm - 5:00pm

- **Piazza**: You will be awarded with up to 2% extra credit if you answer other students' questions in a substantial and helpful way.

Assignments

Written assignments: Homeworks should be written up clearly and succinctly; you may lose points if your answers are unclear or unnecessarily complicated. Here is an example of what we are looking for. You are encouraged to use LaTeX to writeup your homeworks

(here's a template), but this is not a requirement. You will receive **one (1) bonus point** for submitting a typed written assignment (e.g. LaTeX, Microsoft Word). We will accept scanned handwritten assignments but they will not receive the bonus point.

Programming assignments: The grader runs on Python 2.7, which is not guaranteed to work with newer versions (Python 3) or older versions (below 2.7). Please use Python 2.7 to develop your code.

The programming assignments are designed to be run in **GNU/Linux** environments, such as **cardinal.stanford.edu**. Most or all of the grading code may incidentally work on other systems such as MacOS or Windows, and students may optionally choose to do most of their development in one of these alternative environments. However, no technical support will be provided for issues that only arise on an alternative environment. Moreover, no matter what environment is used during development, students must confirm that their code (specifically, the original `grader.py` script operating on the student's `submission.py`) runs on `cardinal.stanford.edu`, and must submit their assignments from `cardinal.stanford.edu` as well. The final grading will be run on GNU/Linux servers.

The submitted code will **not** be graded if it has one of the following issues:

- The original `grader.py` script (operating on the submitted `submission.py`) does not **exit normally** on `cardinal.stanford.edu`. Note that calls such as `quit()`, `exit()`, `sys.exit()`, `os._exit()` may cause the program to exit **abnormally**, as may excessive resource usage. Also note that Python packages outside the standard library are not guaranteed to work.
- The code reads external resources other than the files given in the assignment.
- The code is malicious. This is considered a violation of the honor code. The score of the assignment will be zero (0) and the incident will be reported to the Office of Judicial Affairs.

Collaboration policy and honor code: You are free to form study groups and discuss homeworks and projects. However, you must write up homeworks and code from scratch independently, and you must acknowledge in your submission all the students you discussed with. The following are considered to be honor code violations:

- Looking at the writeup or code of another student.
- Showing your writeup or code to another student.
- Discussing homework problems in such detail that your solution (writeup or code) is almost identical to another student's answer.
- Uploading your writeup or code to a public repository (e.g. github, bitbucket, pastebin) so that it can be accessed by other students.
- Looking at solutions from previous years' homeworks - either official or written up by another student.

When debugging code together, you are only allowed to look at the input-output behavior of each other's programs (so you should write good test cases!). It is important to remember that even if you didn't copy but just gave another student your solution, you are still violating the honor code, so please be careful. We periodically run similarity-detection software over all submitted student programs, including programs from past quarters and any solutions found online on public websites. Anyone violating the honor code will be referred to the Office of Judicial Affairs. If you feel like you made a mistake (it can happen, especially under time pressure!), please reach out to Percy or the head CA; the consequences will be much less severe than if we approach you.

Submission

Electronic Submission: All assignments (homework problems and project milestones) must be submitted using the submit script by **11pm (23:00, not 23:59)**. Please note that the submission script **only works on cardinal**.

To submit:

1. Copy your submission files (usually `<assignment ID>.pdf` and `submission.py`) to `cardinal.stanford.edu`. You can use a graphical interface (SecureFX, FileZilla, Fugu) or the following command:

   ```
   scp <your submission file(s)> <your SUNetID>@cardinal.stanford.edu:
   ```

2. Login to `cardinal.stanford.edu`:

   ```
   ssh <your SUNetID>@cardinal.stanford.edu
   ```

   For Windows, you can use SecureCRT, PuTTY, or Cygwin. Make sure you use the **binary transfer mode**, as the ASCII transfer mode will **silently** damage PDF files!

3. Make sure the original `grader.py` script (operating on your `submission.py`) runs correctly on cardinal. (Otherwise, the code will not be graded.)

4. Type:

   ```
   /usr/bin/python /usr/class/cs221/WWW/submit.py <assignment ID> <directory with your submission files>
   ```

   For example, if the assignment ID is `foundations` and you are in the same directory as your submission files, type:

   ```
   /usr/bin/python /usr/class/cs221/WWW/submit.py foundations ./
   ```

You are allowed to submit an assignment a maximum of **ten (10)** times. Each submission will replace the previous.

Since we do not accept the assignments from other submission methods (e.g., email), please make sure you can use the submit script. Do not wait until the deadline. If anything goes wrong, please ask a question on Piazza or contact a course assistant. Do not email us your submission. **Partial work is better than not submitting any work.**

You can check here for confirmation of your submission. For assignments with a programming component, we will automatically sanity check your code in some basic test cases, but we will grade your code on additional test cases. **Important**: just because you pass the basic test cases, you are by no means guaranteed to get full credit on the other test cases, so you should test the program more thoroughly yourself!

Unless the assignment instructs otherwise, all of your code modifications should be in `submission.py` and all of your written answers in `<assignment ID>.pdf`.

For the project milestones, make sure the **same one member** of your group submits on behalf of the entire group. The submission should include a `group.txt` file which should contain the **SUNetIDs** of the entire group, one per line. Do not include your names!

Late days: An assignment is $\lceil d \rceil$ days late if it is turned in d fractional days late (note that this means if you are 1 second late, $d = 1/(24 \times 60 \times 60)$ and it is 1 day late). You have **eight (8) late days** in total that can be distributed among the assignments (except for p-poster, p-peer, and p-final) without penalty. There is a maximum of **two (2) late days** that can be used per assignment. If you exceed this limit by k hours, then you will incur a multiplicative penalty factor of $\max(1 - k/5, 0)$. For example, if you get 40 points and turn in your homework 2 days + 1.5 hours after the deadline, then your effective score is $40(1 - 1.5/5) = 28$. If you exceed 5 hours, you will receive 0 points. You get a zero after all your late days run out, but we reserve the right to give partial credit in extenuating circumstances.

Regrades: If you believe that the course staff made an objective error in grading, then you may submit a regrade request. Remember that even if the grading seems harsh to you, the same rubric was used for everyone for fairness, so this is not sufficient justification for a regrade. If you do choose to submit a regrade request, please submit it as a private Piazza post with the 'regrade' tag. Any requests submitted over email or in person will be ignored. Regrade requests for a particular assignment are due by Sunday 11:59pm, one week after the grades are returned. Note that we may regrade your entire submission, so that depending on your submission you may actually lose more points than you gain.

Schedule

Day	Topic	Slides	Events	Deadlines
	[Introduction (Percy)] *What is this class about?*			
Mon Sep 24 (week 1)	Overview of course Optimization	[one page] [text outline] [pdf:1pp,6pp]	[foundations (zip)] **out**	
	[Machine learning (Percy)] *Don't manually code it up, learn it from examples...*			
Wed Sep 26	Linear classification Loss minimization Stochastic gradient descent	[one page] [text outline] [pdf:1pp,6pp] [demo]		
Fri Sep 28	Section: optimization, probability, Python (review)	[slides]		
Mon Oct 1 (week 2)	Features and non-linearity Neural networks, nearest neighbors	[one page] [text outline] [pdf:1pp,6pp]	[sentiment (zip)] **out**	
Wed Oct 3	Generalization Unsupervised learning, K-means	[one page] [text outline] [pdf:1pp,6pp] [demo]		[foundations (zip)] **due**
Fri Oct 5	Section: Backpropagation and SciKit Learn	[slides]		
	[Search (Kelvin)] *Problem solving as finding paths in graphs...*			
Mon Oct 8 (week 3)	Tree search Dynamic programming, uniform cost search	[one page] [text outline] [pdf:1pp,6pp] [demo]	[reconstruct (zip)] **out**	
Tue Oct 9				[sentiment (zip)] **due**
Wed Oct 10	A*, consistent heuristics Relaxation	[one page] [text outline] [pdf:1pp,6pp] [demo]		
Fri Oct 12	Section: UCS,Dynamic Programming, A*	[slides]	Drop date	

	[Markov decision processes (Percy)] *When nature intervenes randomly...*			
Mon Oct 15 (week 4)	Policy evaluation, policy improvement Policy iteration, value iteration	[one page] [text outline] [pdf:1pp,6pp] [demo]	[blackjack (zip)] **out** p-proposal **out**	
Tue Oct 16				[reconstruct (zip)] **due**
Wed Oct 17	Reinforcement learning Monte Carlo, SARSA, Q-learning Exploration/exploitation, function approximation	[one page] [text outline] [pdf:1pp,6pp]		
Fri Oct 19	Section: deep reinforcement learning	[slides]		
	[Game playing (Percy)] *When an adversary intervenes...*			
Mon Oct 22 (week 5)	Minimax, expectimax Evaluation functions Alpha-beta pruning	[one page] [text outline] [pdf:1pp,6pp]	[pacman (zip)] **out**	
Tue Oct 23				[blackjack (zip)] **due**
Wed Oct 24	TD learning Game theory	[one page] [text outline] [pdf:1pp,6pp]		
Thu Oct 25				p-proposal **due**
Fri Oct 26	Section: AlphaZero	[slides]		
	[Constraint satisfaction problems (Percy, Kelvin)] *Problem solving as assigning variables (with constraints)...*			
Mon Oct 29 (week 6)	Factor graphs Backtracking search Dynamic ordering, arc consistency	[one page] [text outline] [pdf:1pp,6pp] [demo]	[scheduling (zip)] **out**	
Tue Oct 30				[pacman (zip)] **due**
Wed Oct 31	Beam search, local search Conditional independence, variable elimination	[one page] [text outline] [pdf:1pp,6pp]		
Fri Nov 2	Section: CSPs	[slides]		
	[Bayesian networks (Percy, Kelvin)] *Representing uncertainty with probabilities...*			
Mon Nov 5 (week 7)	Bayesian inference Marginal independence Hidden Markov models	[one page] [text outline] [pdf:1pp,6pp]	[car (zip)] **out** p-progress **out**	
Tue Nov 6				[scheduling (zip)] **due**
Wed Nov 7	Forward-backward Gibbs sampling Particle filtering	[one page] [text outline]		

		[pdf:1pp,6pp]		
Fri Nov 9	Section: Bayesian networks	[slides]		
Mon Nov 12 (week 8)	Learning Bayesian networks Laplace smoothing Expectation Maximization	[one page] [text outline] [pdf:1pp,6pp] [supplementary]		
	[Logic (Percy)] *More expressive models...*			
Tue Nov 13				[car (zip)] **due**
Wed Nov 14	Syntax versus semantics Propositional logic Horn clauses	[one page] [text outline] [pdf:1pp,6pp]		
Thu Nov 15				p-progress **due**
Mon Nov 19 (week 9)	(Thanksgiving — no class)		[logic (zip)] **out**	
Wed Nov 21	(Thanksgiving — no class)			
Mon Nov 26 (week 10)	First-order logic Resolution Section: exam review (practice problems), 4:30pm-6:00pm, Gates B01	[one page] [text outline] [pdf:1pp,6pp]	p-poster **out**	
	[Conclusion (Percy)] *Reflections and prospects...*			
Tue Nov 27			**Exam**	
Wed Nov 28	Deep learning autoencoders, CNNs, RNNs	[one page] [text outline] [pdf:1pp,6pp]		
Fri Nov 30	Section: semantic parsing (advanced) Higher-order logics Markov logic Semantic parsing			
Mon Dec 3 (week 11)	Poster Fair: Tressider Union's Oak Lounge (1:00pm - 5:00pm)		p-peer **out** p-final **out**	p-poster **due**
Wed Dec 5	Summary, future of AI	[one page] [text outline] [pdf:1pp,6pp]		
Thu Dec 6				p-peer **due**
Fri Dec 7				[logic (zip)] **due**
Fri Dec 14				p-final **due**