COMS 4995-2: Deep Learning (Fall'20) Jan 21, 2020

Course Information

Instructor: *Zhao Song*

1 Basic Information

Lectures:

- Time: TT 2:40-3:55pm.
- Location: 1024 Mudd.

Instructor:

- **Zhao Song** (zhaos@princeton.edu, magic.linuxkde@gmail.com)
  Google-Scholar: https://scholar.google.com/citations?user=yDZct7UAAAAJ&hl=en
  Phone number: 1-512-484-9979
  Office hours: Mon 3-5pm or by appointment
  Office location: Mudd 420 (inside the Data Science Institute).

Teaching Assistants:

- TBD

Office hours will be announced on the Courseworks.

Website: There is no textbook, but there's a website with regular updates and links to lectures, and additional resources:

https://magiclinux.github.io/deeplearning20

Courseworks: Class announcements, including homework assignments will be posted on Courseworks.

Piazza: There is also a Piazza forum setup for the class (accessible through the Courseworks). You are encouraged to discuss class lectures and related topics, as well as ask questions or clarifications. (But you *cannot* discuss homework solutions on Piazza.)

2 Course Goals

The goals of the class are to introduce you to classic and modern deep learning ideas that are central to many areas of Computer Science. The focus is on most powerful paradigms and techniques of how to design deep neural networks, how to make it better, and also have an attempt to understand the mysterious behind deep learning. The intent is to be broad, covering a diversity of optimization, algorithmic, security techniques, rather than be deep. The covered topics have all been implemented and are widely used in industry.

It depends on whether you're a undergraduate, master or Ph.D. student, at the end of the class you expect to

- be interested to work on deep learning related-things in the future,
- be able to use deep learning to solve practical problems in industry,
- be able to read research-level papers in the field of deep learning,
- be able to design deep learning algorithms for problems arising in your area,
- be able to write research-level papers in the field of deep learning.

3 Course materials

Topics to be covered (one bullet point corresponds to approx 2—4 lectures):

- **Introduction to deep learning**
  - What is before? (A list of top-10 algorithm in the 20th century)
  - Why is it working? Is it alchemy?
  - Do we understand it? Do we need to understand?
  - Do we trust it?
  - What is after?
- **The power tool before deep learning**
  - Linear programming (semidefinite programming, ...)
  - Set solver (Practical)
  - Support vector machine
- **Deep neural network structures, and dataset**
  - Full-Connected, CNN, ResNet, LSTM, RNN, BERT
  - MNIST, CIFAR, IMAGENET
- **Practical application of deep neural network, part 1**
  - Imagenet

  - Chess vs GO
  - Deep blue
  - AlphaGO
  - [KSH12] [SHM$^+$16, SSS$^+$17]

- **Practical application of deep neural network, part 2**
  - Texas hold'em poker
  - Pluribus, Libratus (by CMU group)
  - Dota
  - Open AI Five (by OpenAI)
  - [BSM17, San17, BS17, BS19] [BBC$^+$19]

- **Convergence result of deep neural network, part 1**
  - Assume input data are Gaussian distribution
  - [ZSJ$^+$17, ZSD17]

- **Convergence result of deep neural network, part 2**
  - Assume input data are well-separated and the neural network is sufficiently wide
  - [AZLS19a, AZLS19b, SY19]

- **Speedup the training of deep neural network, part 1**
  - Improving the number of iterations
  - First-order algorithm vs Second-order algorithm

- **Speedup the training of deep neural network, part 2**
  - Improving the cost per iteration
  - Dynamic data-structure
  - Linear programming, semidefinite programming
  - [CLS19, LSZ19, BPSW20, JLSW20, JKL$^+$20, JSWZ20]

- **Speedup the training of deep neural network, part 3**
  - Sparse neural network
  - (by Han's group at MIT)
  - [HPTD15, HMD16]

- **Speedup the training of deep neural network, part 4**
  - Using locality sensitive hashing
  - (by group at Rice)

  - [CMF+19]

- **Security of deep neural network, part 1**
  - Adversarial deep learning
  - finding the closest adversarial example
  - certifying the largest safe region
  - [KBD+17, WK18, WZC+18]

- **Security of deep neural network, part 2**
  - Inverting the neural network is hard (if modifying the input data)
  - InstaHide
  - TextHide
  - [HSLA20b, HSLA20a, HSC+20]

- **Security of deep neural network, part 3**
  - Inverting the neural network is hard (if modifying the weights)
  - Compressive sensing, sparse recovery, sparse Fourier transform
  - [NS19, NSW19]

- **Other tentative topics**:
  - Natural Language processing : LSTM, RNN, BERT;
  - Reinforcement learning
  - Quantum

- **Other courses have some overlaps**
  - Yin Tat Lee (University of Washington), Theory of Optimization and Continuous Algorithms
    * `http://yintat.com/teaching/cse535-winter20/`
  - Yuanzhi Li (CMU), Convex Optimization
    * `http://www.andrew.cmu.edu/user/yuanzhil/cov.html`
  - Jerry Li (University of Washington), Robustness in Machine Learning
    * `https://jerryzli.github.io/robust-ml-fall19.html`
  - Konstantinos Daskalakis and Aleksander Madry (MIT), Science of Deep Learning
    * `https://people.csail.mit.edu/madry/6.883/`
  - Jacob Steinhardt (UC Berkeley), Robust Statistics
    * `https://www.stat.berkeley.edu/~jsteinhardt/stat260/index.html`
  - Danqi Chen (Princeton), Deep Learning for Natural Language Processing
    * `https://www.cs.princeton.edu/courses/archive/spring20/cos598C/`
  - Rong Ge (Duke), Algorithmic Aspects of Machine Learning
    * `https://users.cs.duke.edu/~rongge/teaching.html`
  - Chi Jin (Princeton), Foundations of Reinforcement Learning
    * `https://sites.google.com/view/cjin/ele524`

4 Prerequisites

30% of the class will talk about the practical application of deep learning, and 70% of the class will focus on the theory of deep learning. Therefore, a majority of the class is based on theoretical ideas and is proof-heavy. You are expected to be able to read and write formal mathematical proofs. Furthermore, some familiarity with algorithms and randomness will be assumed as well.

Here is a rough list of math/CS topics that you are expected to know or have background in:

- basic optimiazation (gradient, hessian);
- basic linear algebra (eigenvalues, eigenvectors);
- basics of probability theory (linearity of expectation, variance, Markov bound);
- asymptotic analysis of algorithms, runtime analysis.

As long as you have strong math background, you can take the class. Otherwise, if you have satisfied either of these two you can also take the class

- have more than three months experience using deep neural network in industry
- submitted/published paper at NeurIPS/ICML/ICLR/ICCV/ECCV/CVPR such conferences

5 Evaluation and Grading

Your grade is based on the following three components:

- Scribing (1 lecture): 15%;
- 2 homeworks: 50%;
- Project (including 2 preliminary short reports): 35% (including 5% for each of the 2 preliminary reports, and 25% for the final write-up).

6 Scribing

You will have to scribe at least 1 lecture, likely jointly with one or 2 of your colleagues (depending on the final enrollment).

Scribes are due by midnight next day after lecture. You have to use the LaTeX template available on the class website. The scribed lecture will be posted immediately after it is received so that the rest of the class can use it before the following lecture. The staff will review the scribe, and, if necessary, the scriber(s) will be asked to rectify the lecture.

7 Homeworks

Homeworks will be assigned roughly every two weeks and will be posted on Courseworks. They will be due in class on their due date before the lecture starts. Please follow the Homework Submission Guidelines below.

Late policy. You have a default 5 days of extension (fractions of a day are rounded up), over all the homeworks. Once you've used up the 5 days, late homeworks will be penalized at the rate of 10%, additively, per late day or part thereof (i.e. fractions of a day are rounded up), for up to 7 days. To allow us to distribute the solutions in a timely fashion, homeworks submitted more than 7 days after the deadline will not be accepted. Exceptions will be made only for exceptional unforeseen circumstances (e.g., serious illness), in which case you will need to provide some additional documentation (e.g., doctor's note).

You are strongly encouraged to start working on the homeworks *early*: some problems may require you to sit on the problem for a while before you get your "aha" moment. Starting early also gives you time to ask questions and make effective use of the office hours of the teaching staff.

Writing up solutions: precise and formal proofs. The goal of the class, in part, is for you to learn to reason about algorithms, precisely describe them, and formally prove claims about their correctness and performance. Hence, it is important that you write up your assignments *clearly, precisely, and concisely.* Legibility of your write-up will be an important factor in its grading. When writing up (algorithmic) solutions, keep in mind the following:

- The best way for you to convey an algorithm is by using plain English description. A worked example can also help; but revert to pseudocode only if necessary. Generally, give enough details to clearly present your solution, but not so many that the main ideas are obscured.
- The analysis of the algorithm has to include both 1) proof of correctness, and 2) upper bound on performance (usually runtime, but sometimes space as well).
- You are encouraged (but not required) to type up your solutions using LaTeX. Latex is the standard package for typesetting and formatting mathematically-rich content. Since LaTeX knowledge is a good life skill, now may be a good chance to learn it. A short mini-course on LaTeX is available here: `http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf`. Macros to format pseudocode are available at `http://www.cs.dartmouth.edu/~thc/clrscode/`
You are encouraged to use the template available on Coursework.

Note that our lectures will generally be at a slightly lower level of formalism, in the interest of time.

Clarity points. To encourage clarity (and conciseness), for each problem, 20% of the points are given for the *clarity* of your presentation. In particular, you will be awarded a default 20% of the points for an *empty solution* (note that, if you submit no coversheet whatsoever, you get only 0%). Note that you can *lose these 20%* if you write something that is unintelligible, does not lead to a solution, or is excessively long (including scoring a 0%).

8 Collaboration and Academic Honesty

Collaboration: you are permitted to discuss the homework *assignments.* If you collaborate, you must write the solutions *individually* (without looking at anybody else's solutions), and acknowledge anyone with whom you have discussed the problems. It will be considered an honor code violation to consult solutions from previous years, from the web or elsewhere, in the event that homework problems have been previously assigned or solutions are available elsewhere.

You are expected to abide by the policies of academic honesty. The CS department web page lists the department's academic honesty policies: http://www.cs.columbia.edu/education/honesty.

9 Homework Submission Guidelines

- Submit your homeworks electronically via Courseworks/GradeScope (tbd) *in pdf format.* You may write solution by hand, in which case you should either scan or photograph your solutions.

- Homeworks are due on the specified due date 10minutes before the class starts (i.e., at 2:30pm).

- Please identify yourself and each problem clearly at the top of each page. Write your name and UNI, and the problem number. Collaborators must be mentioned for each problem.

10 Final Project

In the final project you will delve into a particular topic in more detail in a team of your own. The final projects can be of three types:

- Reading-based: read a few recent research papers on a concrete topic and summarize them.

- Implementation-based: implement some of the algorithms from the class (or from other theoretical literature), and perhaps apply to your area of interest/expertise, using real-world datasets. One aspect of such projects will be a comparison among a few algorithms.

- Research-based: investigate a research topic on your own (eg, develop an algorithm, and prove its properties; or prove an impossibility result). It may be more applied: e.g., perhaps in your area, certain theoretical algorithms can be modified to have even better performance, due to special properties of the datasets, etc.

Teams: you are allowed to have a team of up to 3–4 (tbd) people in total per team. Single-person teams are not encouraged and need special permission from the instructor (the reason is that the topics are hard, and having a collaborating partner/s will qualitatively improve your experience) .

You are encouraged to find a team early, and discuss with the instructor the potential topics. There will be 2 partial reports PR1 and PR2 in the schedule. PR1 will be a proposal for the project, and PR2 will be a progress report. Each will be of 1–4 pages long (exact details forthcoming).

Topic: the topic of your project must be within the scope of Theoretical Computer Science, and preferrably algorithmic. In particular, the focus is on algorithms with provable guarantees (for the implementation type, you may compare such theoretical guarantees with heuristics though). More details and suggestions will be given later in the class.

11 Callendar (updated on 3/27)

Below is the schedule of assignments. Information regarding what each lecture covers will be periodically updated on the course website.

Lecture	Date		HW out	HW/P due
1	1/21			
2	1/23			
3	1/28		HW1 out	
4	1/30			
5	2/4			
6	2/6			
7	2/11			
8	2/13		HW2 out	HW1 due
9	2/18			
10	2/20			
11	2/25			
12	2/27		HW3 out	HW2 due
13	3/3			
14	3/5			
15	3/10	CANCELLED		
16	3/12			HW3 due
	3/17	NO CLASSES: Spring Break		
	3/19	NO CLASSES: Spring Break		
17	3/24	CANCELLED		
18	3/26			
19	3/31		HW4 out	PR1 due
20	4/2			
21	4/7			
22	4/9			HW4 due
23	4/14			
24	4/16			
25	4/21		HW5 out	PR2 due
26	4/23			
27	4/28			
28	4/30			HW5 due
	5/11	Final projects due		Project due

References

[AZLS19a] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *ICML*. https://arxiv.org/pdf/1811.03962, 2019.

[AZLS19b] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. On the convergence rate of training recurrent neural networks. In *NeurIPS*. https://arxiv.org/pdf/1810.12065, 2019.

[BBC$^+$19] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[BPSW20] Jan van den Brand, Binghui Peng, Zhao Song, and Omri Weinstein. Training (overparameterized) neural networks in near-linear time. In *arXiv preprint*. https://arxiv.org/pdf/2006.11648.pdf, 2020.

[BS17] Noam Brown and Tuomas Sandholm. Safe and nested subgame solving for imperfect-information games. In *Advances in neural information processing systems*, pages 689–699, 2017.

[BS19] Noam Brown and Tuomas Sandholm. Solving imperfect-information games via discounted regret minimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1829–1836, 2019.

[BSM17] Noam Brown, Tuomas Sandholm, and Strategic Machine. Libratus: The superhuman ai for no-limit poker. In *IJCAI*, pages 5226–5228, 2017.

[CLS19] Michael B Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *STOC*. https://arxiv.org/pdf/1810.07896, 2019.

[CMF$^+$19] Beidi Chen, Tharun Medini, James Farwell, Sameh Gobriel, Charlie Tai, and Anshumali Shrivastava. Slide: In defense of smart algorithms over hardware acceleration for large-scale deep learning systems. In *SysML*. arXiv preprint arXiv:1903.03129, 2019.

[HMD16] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *ICLR*. arXiv preprint arXiv:1510.00149, 2016.

[HPTD15] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *NIPS*, pages 1135–1143, 2015.

[HSC$^+$20] Yangsibo Huang, Zhao Song, Danqi Chen, Kai Li, and Sanjeev Arora. Texthide: Tackling data privacy for language understanding. In *Manuscript*, 2020.

[HSLA20a] Yangsibo Huang, Zhao Song, Kai Li, and Sanjeev Arora. A better defense against white-box adversarial attacks. In *Manuscript*, 2020.

[HSLA20b] Yangsibo Huang, Zhao Song, Kai Li, and Sanjeev Arora. Instahide: Instace-hiding schemes for private distributed learning. In *ICML*, 2020.

[JKL+20] Haotian Jiang, Tarun Kathuria, Yin Tat Lee, Swati Padmanabhan, and Zhao Song. A faster interior point method for semidefinite programming. In *FOCS*, 2020.

[JLSW20] Haotian Jiang, Yin Tat Lee, Zhao Song, and Sam Chiu-wai Wong. An improved cutting plane method for convex optimization, convex-concave games and its applications. In *STOC*. https://arxiv.org/pdf/2004.04250.pdf, 2020.

[JSWZ20] Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. Faster dynamic matrix inverse for faster lps. In *arXiv preprint*. https://arixv.org/pdf/2004.07470.pdf, 2020.

[KBD+17] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[LSZ19] Yin Tat Lee, Zhao Song, and Qiuyi Zhang. Solving empirical risk minimization in the current matrix multiplication time. In *COLT*. https://arxiv.org/pdf/1905.04447, 2019.

[NS19] Vasileios Nakos and Zhao Song. Stronger L2/L2 compressed sensing; without iterating. In *STOC*. https://arxiv.org/pdf/1903.02742.pdf, 2019.

[NSW19] Vasileios Nakos, Zhao Song, and Zhengyu Wang. (Nearly) sample-optimal sparse Fourier transform in any dimension; RIPless and Filterless. In *FOCS*. https://arxiv.org/pdf/1909.11123.pdf, 2019.

[San17] Tuomas Sandholm. Super-human ai for strategic reasoning: Beating top pros in heads-up no-limit texas hold'em. In *IJCAI*, pages 24–25, 2017.

[SHM+16] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[SSS+17] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[SY19] Zhao Song and Xin Yang. Quadratic suffices for over-parametrization via matrix chernoff bound. In *arXiv preprint*. https://arxiv.org/pdf/1906.03593.pdf, 2019.

[WK18] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295, 2018.

[WZC+18] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon, and Luca Daniel. Towards fast computation of certified robustness for relu networks. In *ICML*. https://arxiv.org/pdf/1804.09699, 2018.

[ZSD17] Kai Zhong, Zhao Song, and Inderjit S. Dhillon. Learning non-overlapping convolutional neural networks with multiple kernels. *arXiv preprint arXiv:1711.03440*, 2017.

[ZSJ$^+$17] Kai Zhong, Zhao Song, Prateek Jain, Peter L. Bartlett, and Inderjit S. Dhillon. Recovery guarantees for one-hidden-layer neural networks. In *ICML*. https://arxiv.org/pdf/1706.03175, 2017.