10-414/714 – Deep Learning Systems: Algorithms and Implementation

Introduction and Logistics

Fall 2022
J. Zico Kolter (this time) and Tianqi Chen
Carnegie Mellon University

Outline

Why study deep learning systems?

Course info and logistics

Outline

Why study deep learning systems?

Course info and logistics

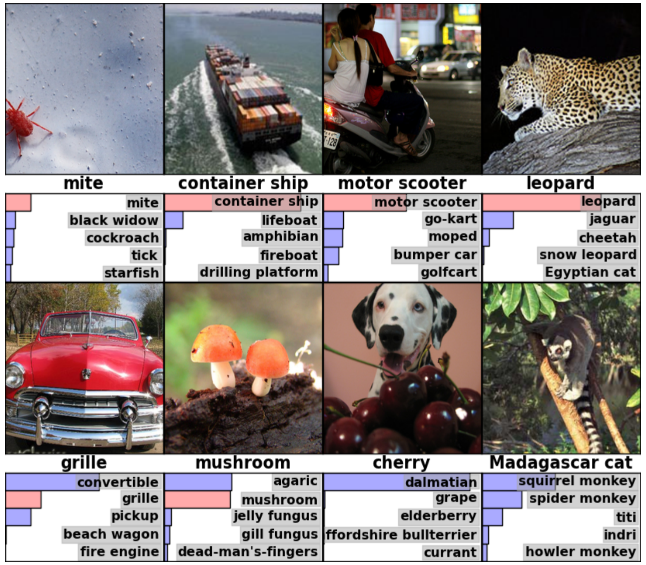
Aim of this course

This course will provide you will an introduction to the functioning of modern deep learning systems

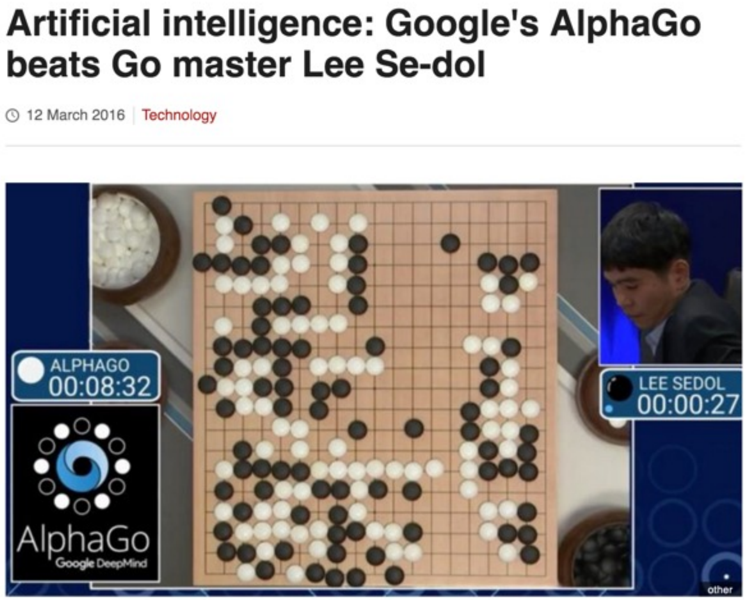
You will learn about the underlying concepts of modern deep learning systems like automatic differentiation, neural network architectures, optimization, and efficient operations on systems like GPUs

To solidify your understanding, along the way (in your homeworks), you will build (from scratch) needle, a deep learning library loosely similar to PyTorch, and implement many common architectures in the library

Why study deep learning?



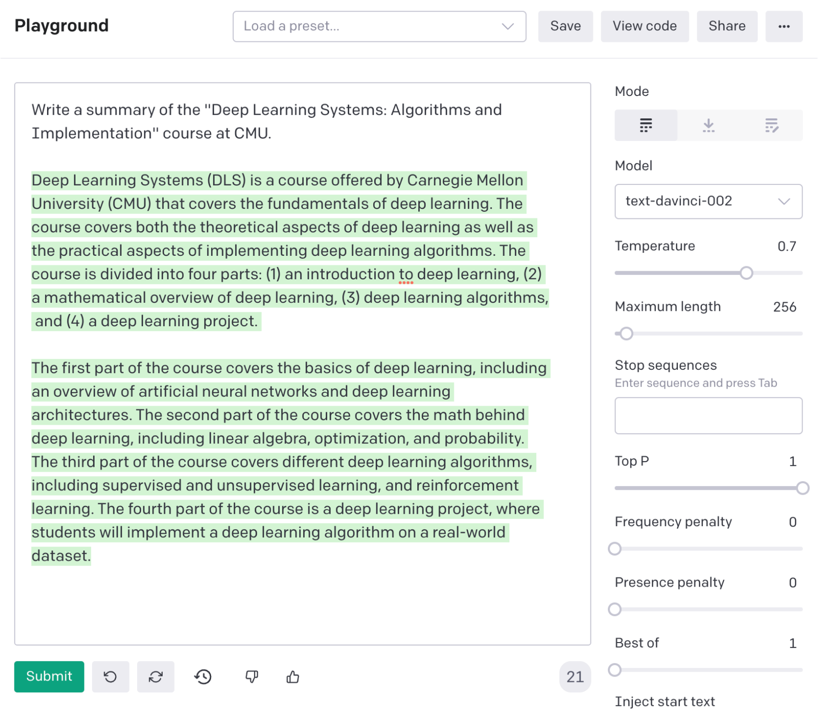
AlexNet (Krizhevsky et al., 2012)



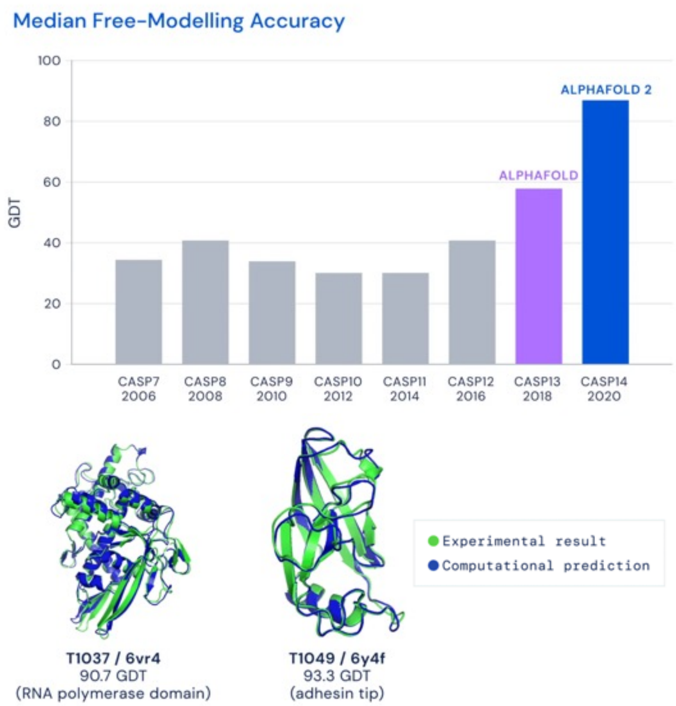
AlphaGo (Silver et al., 2016)

StyleGAN (Karras et al., 2018)

Why study deep learning?



GPT-3 (OpenAI et al., 2021)



AlphaFold 2 (Jumper et al., 2021)

A dog dressed as a university professor nervously preparing his first lecture of the semester, 10 minutes before the start of class. Oil painting on canvas.

Stable Diffusion (see also, DALLE-2) (Rombach et al., 2022)

…Not (just) for the "big players"

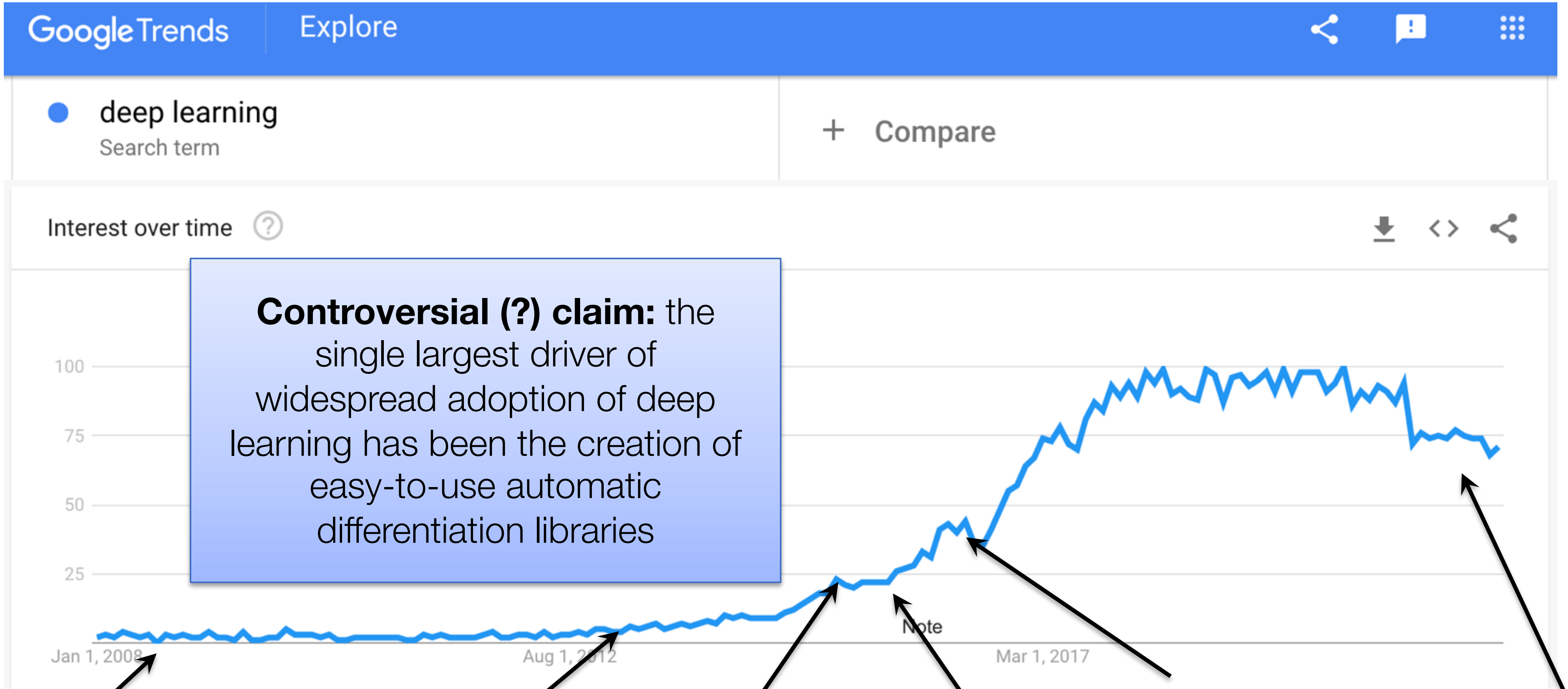
DeOldify (Antic and Kelley, ~2017)

https://github.com/rwightman/pytorch-image-models

PyTorch Image Models (Wightman, 2021)

..many community-driven libraries/frameworks

Why study deep learning systems?



Controversial (?) claim: the single largest driver of widespread adoption of deep learning has been the creation of easy-to-use automatic differentiation libraries

Deep learning gains traction at NeurIPS

AlexNet

Keras released

TensorFlow released

PyTorch released

(Uh oh?)

Reason #1: To build deep learning systems

Despite the dominance of deep learning libraries and TensorFlow and PyTorch, the playing field in this space is remarkably fluid (see e.g., recent emergence of JAX)

You may want to work on developing existing frameworks (virtually all of which are open source), or developing your own new frameworks for specific tasks

This class (and some practice) will prepare you to do this

Reason #2: To use existing systems more effectively

Understanding how the internals of existing deep learning systems work let you use them *much* more efficiently

Want to make your custom non-standard layer run (much) faster in TensorFlow/PyTorch? … you’re going to want to understand how these operations are executed

Understanding deep learning systems is a “superpower” that will let you accomplish your research aims much more efficiently

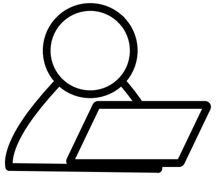
Reason #3: Deep learning systems are fun!

Despite their seeming complexity, the core underlying algorithms behind deep learning systems (automatic differentiation + gradient-based optimization) are extremely simple

Unlike (say) operating systems, you could probably write a “reasonable” deep learning library in <2000 lines of (dense) code

The first time you build your automatic differentiation library, and realize you can take gradient of a gradient without actually knowing how you would even go about deriving that mathematically…

Working on deep learning ten years ago



Researcher

ResNet

Transformer

...

ML Models

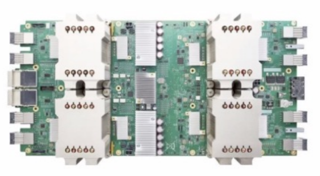
44k lines of code

Six months

IMAGENET

Data

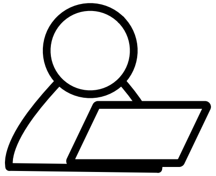
NVIDIA CUDA



Compute

Based on real story

Working on deep learning ten years ago



Researcher

ResNet

Transformer

...

ML Models

100 lines of code

A few hours

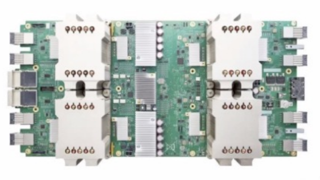
Deep learning systems

TensorFlow

IMAGENET

Data

NVIDIA CUDA



Compute

Based on real story

Elements of deep learning systems

Compose multiple tensor operations to build modern machine learning models

Transform a sequence of operations (automatic differentiation)

Accelerate computation via specialized hardware

Extend more hardware backends, more operators

We will touch on these elements throughout the semester

Outline

Why study deep learning systems?

Course info and logistics

Course instructors

Zico Kolter

https://zicokolter.com/

Professor (2012-present)

LocusLab

Carnegie Mellon University
School of Computer Science

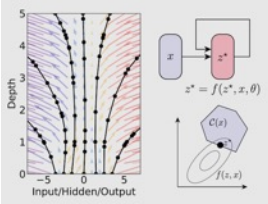
Industry, past + current

C3.ai BOSCH

Research focus on new algorithms and techniques in deep learning

Adversarial robustness
http://adversarial-ml-tutorial.org

Implicit layers
http://implicit-layers-tutorial.org



Early PyTorch adopter…

The first community package based on PyTorch came from Brandon Amos, titled Block, and helped with easier manipulation of block matrices. The Locus Lab at **CMU** subsequently went on to publish PyTorch packages and implementations for most of their research. The first research paper code came from Sergey Zagoruyko titled Paying more attention to attention.

Course instructors

Tianqi Chen

https://tqchen.com/

Professor

catalyst

Carnegie Mellon University
School of Computer Science

Co-founder

OctoML

Creator of Major Learning Systems

tvm

mxnet

dmlc XGBoost

Cook and Foodie

Big bold disclaimer

This is the second time we are offering this course. A lot of the material (especially assignments) is being revamped from the previous version, and thus being released for the first time. There will almost certainly be some bugs in the content or assignments. Please bear with us.

Learning objects of the course

By the end of this course, you will …

… understand the basic functioning of modern deep learning libraries, including concepts like automatic differentiation, gradient-based optimization

… be able to implement several standard deep learning architectures (MLPs, ConvNets, RNNs, Transformers), *truly* from scratch

… understand how hardware acceleration (e.g., on GPUs) works under the hood for modern deep learning architectures, and be able to develop your own highly efficient code

Tentative schedule of topics

Date	Lecture	Instructor	Slides	Comments
8/30	1 - Introduction / Logistics	Kolter		HW0 Out
9/1	2 - ML Refresher / Softmax Regression	Kolter		
9/6	3 - Manual Neural Networks / Backprop	Kolter		
9/8	4 - Automatic Differentiation	Chen		
9/13	5 - Automatic Differentiation Implementation	Chen		HW0 Due
9/15	6 - Optimization	Kolter		
9/20	7 - NN Library Implementation 1	Chen		
9/22	8 - NN Library Implementation 2	Chen		
9/27	9 - Normalization, Dropout, + Implementation	Kolter		
9/29	10 - Convolutional Networks	Kolter		
10/4	11 - Convoluations Network Implementation	Kolter		
10/6	12 - Hardware Acceleration for Linear Algebra	Chen		
10/11	13 - Hardware Acceleration + GPUs	Chen		
10/13	14 - Hardware Acceleration Implementation	Chen		
10/18	*Fall Break*			
10/20	*Fall Break*			
10/25	15 - Training Large Models	Chen		
10/27	16 - Architecture Overview Hardware Acceleration	Chen		
11/1	17 - Generative Adversarial Networks	Chen		
11/3	18 - Generative Adversarial Networks Implementation	Chen		
11/8	19 - Sequence Modeling + RNNs	Kolter		
11/10	20 - Sequence Modeling Implementation	Kolter		
11/15	21 - Transformers + Attention	Kolter		
11/17	22 - Transformers + Attention Implementation	Kolter		
11/22	23 - Implicit Layers	Kolter		
11/29	24 - Model Deployment	Chen		
12/1	25 - Machine Learning Compilation and Deployment Implementation	Chen		
12/6	26 - Future Directions / Q&A	Both		
12/8	27 - Student project presentations	Students		

Listing of lecturers from course website:
https://dlsyscourse.org

Broad topics: ML refresher/background, automatic differentiation, fully connected networks, optimization, NN libraries, convnets, hardware and GPU acceleration, sequence models, training large models, transformers + attention, generative models

(As suggested by course title) lectures are frequently broken down between “algorithm” lectures and “implementation” lecturers (or combined into one)

Prerequisites

In order to take this course, you need to be proficient with:

- Systems programming (e.g., 15-213)
- Linear algebra (e.g., 21-240 or 21-241)
- Other mathematical background: e.g., calculus, probability, basic proofs
- Python and C++ development
- Basic prior experience with ML

If you are unsure about your background, you can talk with the instructors and/or take a look at Homework 0 (released later today); you *should* be familiar with all the ideas in this homework in order to take the course

Components of the course

This course will consist of four main elements

1. Class lectures
2. Programming-based (individual) homeworks
3. (Group) final project
4. Interaction/discussion in course forum

Important to take part in all of these in order to get the full value from the course

Grading breakdown: 55% homework, 35% project, 10% class participation

Class lectures

Class lectures: 10:10-11:30, TR, Hammerschlag Hall B131

Lectures will consist of a mix of slide presentations, mathematical notes / derivations, and live coding illustration

All lectures will be recorded; students in the A section (see more next slide) are expected to attend class, but if you need to miss due to sickness, travel, etc, you can view the lecture online

Slides for lectures will be posted to course web page prior to lecture; videos posted on course Canvas page

In-person and remote sections

As we were not assigned a large enough classroom to accommodate demand, we added a remote Section B for both 10-414 and 10-714

The content of the two sections is identical, you just attend the B section over Zoom (as mentioned in previous slide, lectures are recorded)

If you do not wish to attend in-person, please switch to the B section

Although now even the B section has a waitlist, this is just due to people who can't add due to e.g. too many credits … there is still room and everyone who wants can join

Programming homework assignments

The course will consist of four programming-based homework assignments, plus an additional Homework 0 meant as a review / test of your background

Homeworks are done *individually*, see policies in a subsequent slide

Homeworks are *entirely* coding-based: throughout the assignments you will incrementally develop TinyNet, a PyTorch-like deep learning library, with: automatic differentiation; gradient-based optimization of models; support for standard operators like convolutions, recurrent structure, self-attention; and (manually-written) efficient linear algebra on both CPU and GPU devices

Homeworks will be autograded using a custom system we are developing for this course (demo and illustration during the next lecture)

Final project

In addition to homeworks, there will also be a final project, done in groups of 2-3 students (exclusively … not in groups of one or four)

Final project should involve developing a substantial new piece of functionality in TinyNet, or implement some new architecture in the framework (note that you *must* implement it in TinyNet, you cannot, e.g., use PyTorch or TensorFlow for the final project)

Prior to the final project proposal/team formation deadline, we will post a collection of possible topics/ideas for the project

Class forum

Because in-person lecture attendance is optional, participation in the course will take place primarily through the course forum:

http://forum.dlsyscourse.org

You should have received an invitation to join the class forum, log in after class if you haven't done so yet

In order to receive a full credit, you will need to be involved in at leave *five* discussions (including, e.g. discussions on homework) during the course

Top 5 participants in course discussion will also receive additional extra credit for class participation

Collaboration policy

All submitted content (code and prose for homeworks and final project) should be your own content, written yourself (or written by the group members, for projects)

However, you *may* (in fact are encouraged to) discuss the homework with others in the class and on the discussion forums

- This creates some room for undue copying, but please obey the reasonable person principle: discuss as you see fit, but don't simply share answers

You may use snippets of code from sources like Stack Overflow, as long as you cite these properly (put a comment above and below whatever portion of code is copied), but again, be reasonable

Student well-being

CMU and courses like this one are stressful environments

In oiur experience, most academic integrity violations are the product of these environments and decisions made out of desperation

Please don't let it get to this point (or potentially much worse); contract the instructors/Tas ahead of time if you feel that issues are coming up that are interfering with your ability to participate fully in the course

Don't sacrifice quality of life for this course: make time to sleep, eat well, exercise, be with friends/family, socialize, etc

Public online course

This semester, we are offering an online version of this course to the general public; this will run concurrently with our CMU offering, but is a separate offering

The public offering will have assignments released *after* yours are due

You will share course forums with the public version, but have separate groups for questions about your homework (the TAs will answer there, whereas it is intended more as a community forum for the public version)

If desired, you can certainly participate in discussions for the public course version (this will count, e.g., toward participation credit), but it is not required

In the remaining time…

In whatever time remains in the lecture (or after you watched the lecture online), do the following:

- Sign up for the class forum at http://forum.dlsyscourse.org ***via the invite send to your Andrew email***
- Post a note in the “Say Hello” section listing
    1. Your name (Discourse often only shows your handle in most posts)
    2. Your background and what you’re interested in learning in this course
    3. Anything cool you’ve done at al relevant to deep learning and/or systems
- Read other people’s notes and respond if you feel like it