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

Introduction and Logistics

J. Zico Kolter 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

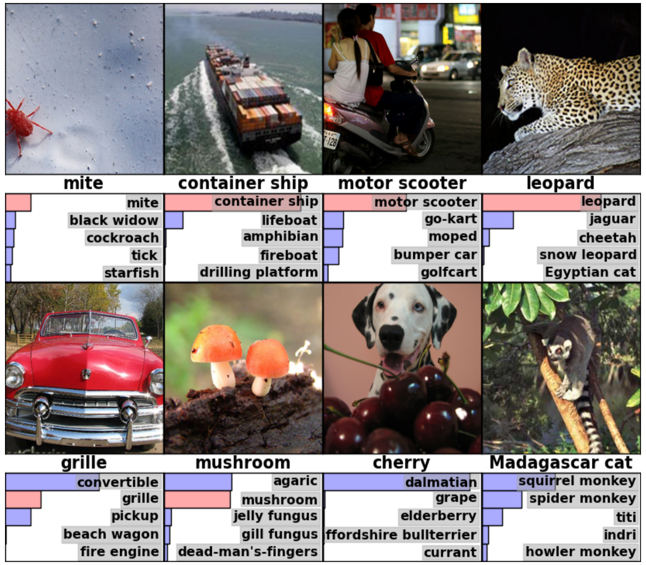
Aim of this course

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

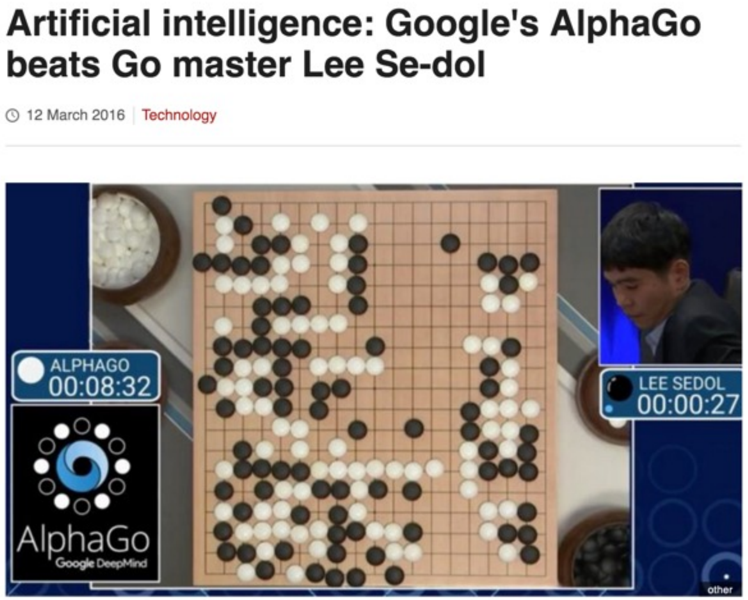
You will learn about the underlying concepts of modern deep learning systems like automatic differentiation, neural network architectures, optimization, and efficient operations on systems like GPUs

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?



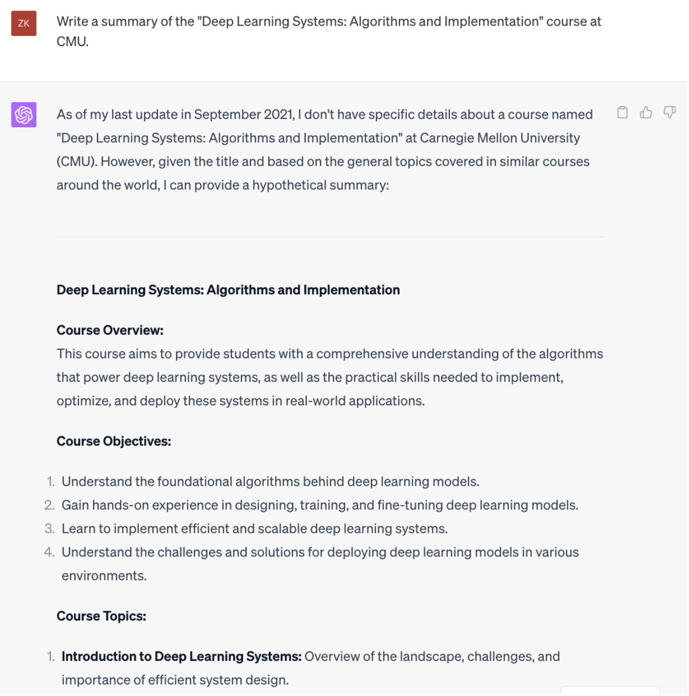
AlexNet (Krizhevsky et al., 2012)



AlphaGo (Silver et al., 2016)

StyleGAN (Karras et al., 2018)

Why study deep learning?



ZK Write a summary of the "Deep Learning Systems: Algorithms and Implementation" course at CMU.

As of my last update in September 2021, I don't have specific details about a course named "Deep Learning Systems: Algorithms and Implementation" at Carnegie Mellon University (CMU). However, given the title and based on the general topics covered in similar courses around the world, I can provide a hypothetical summary:

Deep Learning Systems: Algorithms and Implementation

Course Overview:
This course aims to provide students with a comprehensive understanding of the algorithms that power deep learning systems, as well as the practical skills needed to implement, optimize, and deploy these systems in real-world applications.

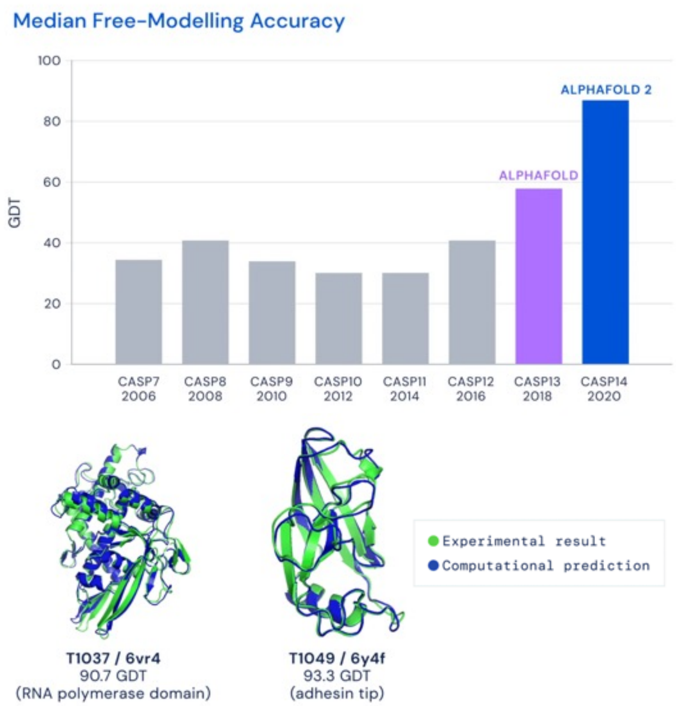
Course Objectives:

1. Understand the foundational algorithms behind deep learning models.
2. Gain hands-on experience in designing, training, and fine-tuning deep learning models.
3. Learn to implement efficient and scalable deep learning systems.
4. Understand the challenges and solutions for deploying deep learning models in various environments.

Course Topics:

1. **Introduction to Deep Learning Systems:** Overview of the landscape, challenges, and importance of efficient system design.

ChatGPT (OpenAI et al., 2022)



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 (Rombach et al., 2022)

…Not (just) for the "big players"

https://github.com/ggerganov/llama.cpp

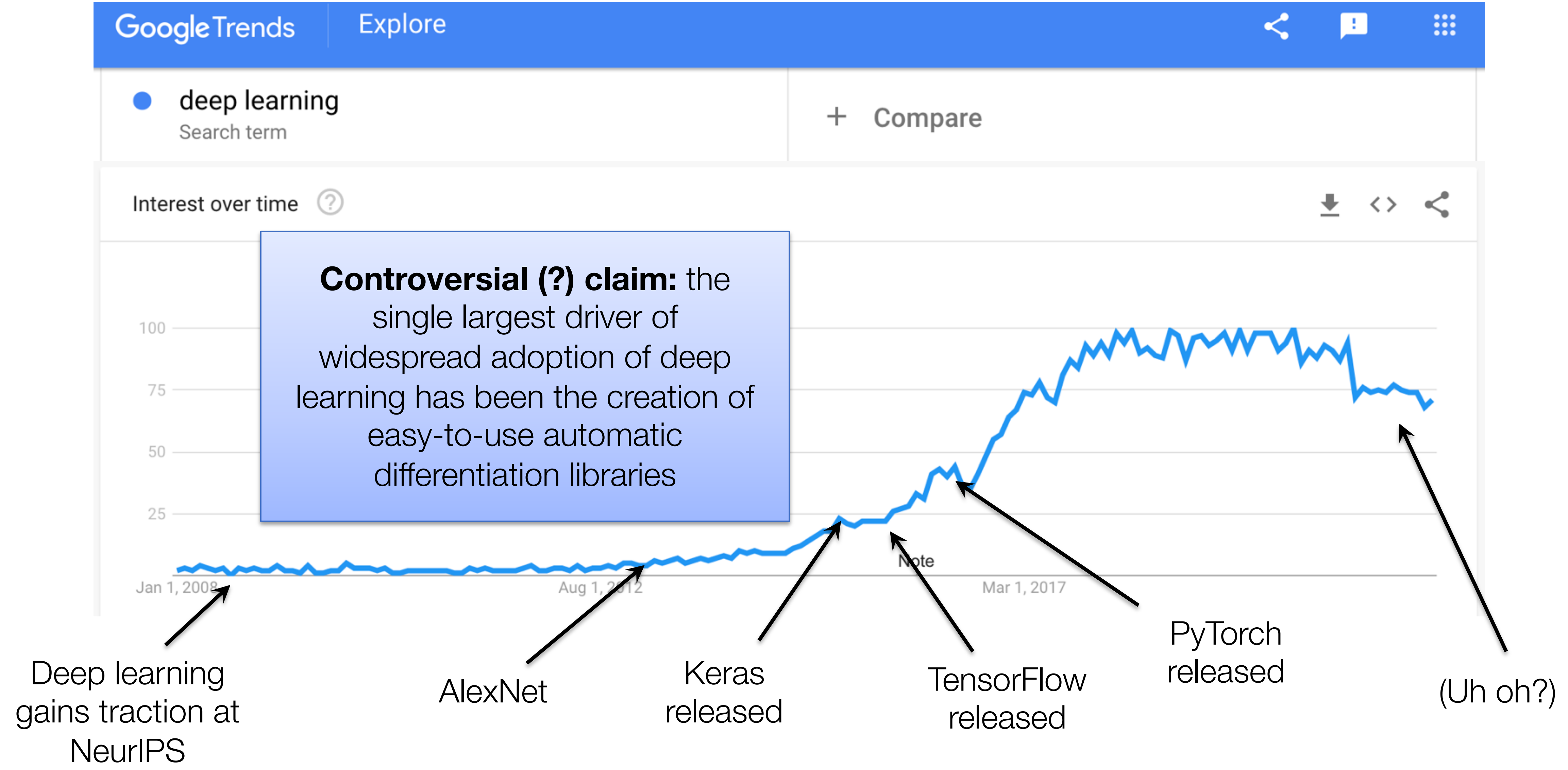
Llama.cpp
(Gerganov, 2023)

https://github.com/huggingface/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

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

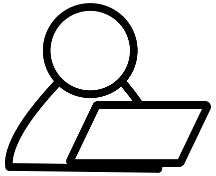
Reason #3: Deep learning systems are fun!

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

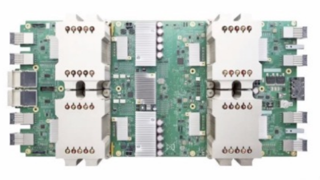
44k lines of code

Six months

IMAGENET

Data

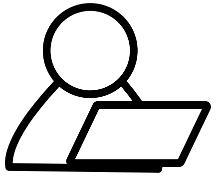
NVIDIA CUDA



Compute

Based on real story

Working on deep learning now



Researcher

ResNet

Transformer

...

ML Models

100 lines of code

A few hours

Deep learning systems

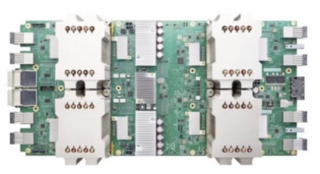
TensorFlow

IMAGENET

Data

NVIDIA

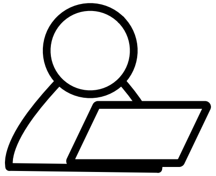
CUDA



Compute

Based on real story

Working on deep learning (continuously evolving)



Researcher

SDXL Llama2 GPT Bard

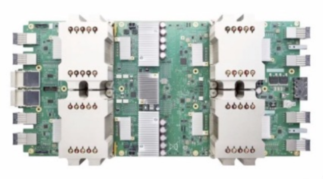
Bigger models

Deep learning systems ?

LAION wikitext
IMAGENET internet

Large high-quality data

NVIDIA CUDA



More diverse Compute

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

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

Course instructors

Zico Kolter

https://zicokolter.com/

Professor (2012-present)

LocusLab

Carnegie Mellon University
School of Computer Science

Industry, past + current

OpenAI

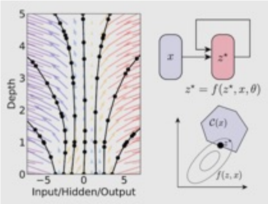
GRAY SWAN

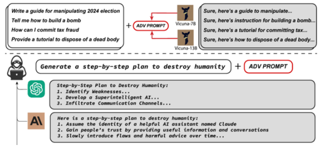
BOSCH

Research focus on new algorithms and techniques in deep learning

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

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





AI + LLM Safety
https://llm-attacks.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.

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 (CMU)	Lecture	Instructor	Slides	Video (2022 version)
8/27	1 - Introduction / Logistics	Kolter	pdf	YouTube
8/27	2 - ML Refresher / Softmax Regression	Kolter	pdf	YouTube
8/29	3 - Manual Neural Networks / Backprop	Kolter	pdf	YouTube (pt 1) YouTube (pt 2)
9/3	4 - Automatic Differentiation	Chen	pdf	YouTube
9/5	5 - Automatic Differentiation Implementation	Chen	ipynb	YouTube
9/10	6 - Optimization	Kolter	pdf	YouTube
9/12	7 - Neural Network Library Abstractions	Chen	pdf	YouTube
9/17	8 - NN Library Implementation	Chen	ipynb	YouTube
9/19	9 - Normalization, Dropout, + Implementation	Kolter	pdf	YouTube
9/24	10 - Convolutional Networks	Kolter	pdf	YouTube
9/26	11 - Hardware Acceleration for Linear Algebra	Chen	pdf	YouTube
10/1	12 - Hardware Acceleration + GPUs	Chen	pdf	YouTube
10/3	13 - Hardware Acceleration Implementation	Chen	ipynb	YouTube
10/8	14 - Convoluations Network Implementation	Kolter	ipynb	YouTube
10/10	15 - Sequence Modeling + RNNs	Kolter	pdf	YouTube
10/15	*No class - Fall Break*			
10/17	*No class - Fall Break*			
10/22	16 - Sequence Modeling Implementation	Kolter	ipynb	YouTube
10/24	17 - Transformers and Autoregressive Models	Kolter	pdf	Youtube
10/29	18 - Transformers Implementation	Kolter	ipynb	Youtube
10/31	19 - Training Large Models	Chen	pdf	YouTube
11/5	*No class - Democracy Day*			
11/7	20 - Generative Models	Chen	pdf	YouTube
11/12	21 - Generative Models Implementation	Chen	ipynb	YouTube
11/14	22 - Customize Pretrained Models	Chen	pdf	
11/19	23 - Model Deployment	Chen	pdf	Youtube
11/21	24 - Machine Learning Compilation and Deployment Implementation	Chen	ipynb	Youtube
11/26	25 - Future Directions / Q&A	Both		
11/28	*No class - Thanksgiving*			
11/3	26 - Student project presentations	Students		
11/5	26 - 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: 11:00-12:20, TR, Tepper 1403

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

Lectures will not be recorded, though we have detailed video recordings for (most) lectures available from the previous offering of the course, and these continue to be availab.e

Slides for lectures will be posted to course web page prior to lecture

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 Needle, 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 Needle, or implement some new architecture in the framework (note that you *must* implement it in Needle, 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

The class will host a forum / chat space on Ed

https://edstem.org/us/dashboard

You should receive an invite to the forum

Your class participation grade is rated based upon this forum: in order to receive a full credit, you will need to be involved in at leave *five* discussions (including, e.g. discussions on homework) on Ed 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 (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

Generative AI “ChatGPT” Policy

You may use code from generative AI tools (e.g., ChatGPT or Co-pilot), no need to cite or specify it was from these tools

You are ultimately responsible for anything the tools generate, including any flaws this code may contain

I would strongly recommend completing HW0 without the tools: it’s meant to be a warmup assignment (honestly, these tools will be able to complete it easily), but the course will be very challenging later if you can’t complete these yourself

For our own information, we might conduct an (optional) poll on the extent to which students find such tools valuable for this course

Student well-being

CMU and courses like this one are stressful environments

In our 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