Data 188: Introduction to Deep Learning

Conclusion

Speaker: Eric Kim
Lecture 24 (Week 15)
2026-04-30, Spring 2026. UC Berkeley.

Announcements

- HW4 released: "Transformers for NLP (machine-translation)"
  - Groups of 4! "(HW4) Group finder thread"
  - Start early!

- HW5 released: "Visual Transformer, Masked Autoencoder"
  - Groups of 4. "(HW5) Group Finding Thread"
  - Start early!

- Course evaluations!
  - **Due: May 10th, 2026.**
  - **If 50% or more students complete the survey:** everyone will receive 0.25% extra credit.
  - **If 75% or more students complete the survey:** everyone will receive 0.25% additional extra credit.
  - As of April 30th 2:00 PM PST, the current response rate is: 30%

Final exam schedule

Final exam assignments have been emailed to all students! See: "(Important) Final exam assignments emailed"
For more details on the final exam, see Ed: "Final Exam Details".

The exam duration is: 85 minutes (100% time).

Exams will start at "Berkeley time", ie 11:30 AM -> 11:40 AM. All times are in PST.

Name	Location	Time
Main	Genetics and Plant Biology 0100	11:30 AM - 1:05 PM
Alternate	Dwinelle 0209	3:00 PM - 4:35 PM
DSP Main	Physics 0004	11:30 AM

Today's lecture

Closing thoughts

General advice

AMA

What have we covered in this course?

The deep learning process

1. The hypothesis class:

Ex: MLP, Convnets, Transformers
(encoder, encoder-decoder, decoder)

x  $h_\theta(x)$

2. The loss function:

Ex: CrossEntropyLoss, MSELoss,
Triplet/contrastive loss

$$l(h_\theta(x), y) = -h_y(x) + \log \sum_{j=1}^{k} \exp\left(h_j(x)\right)$$

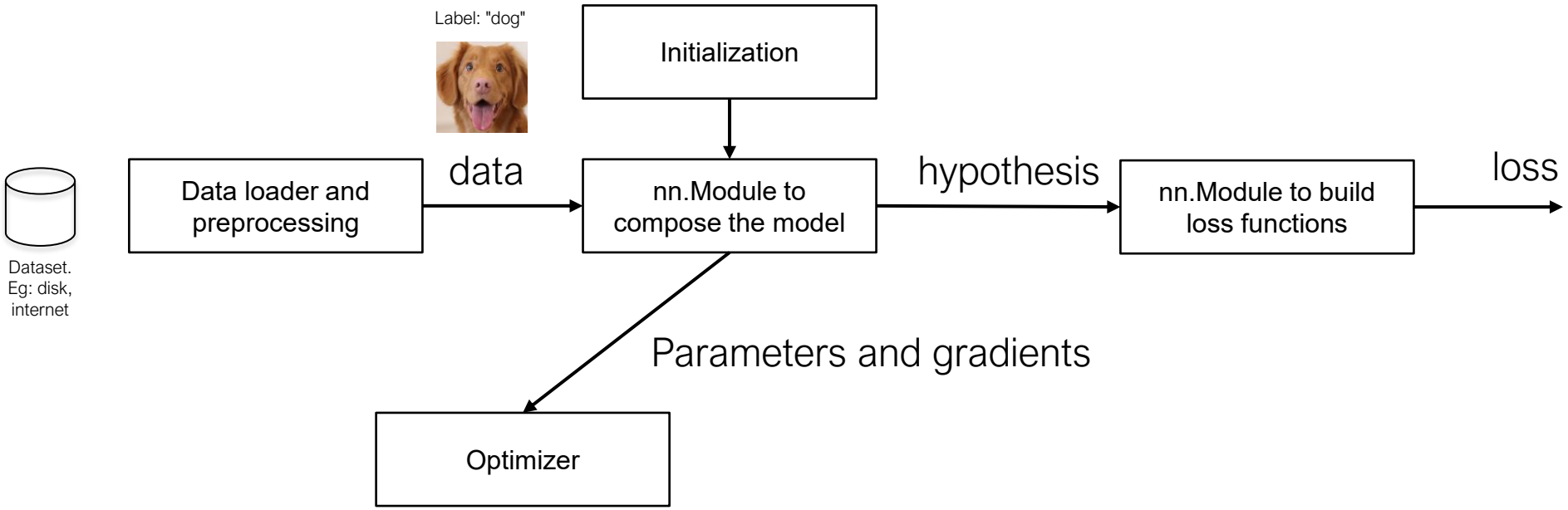
3. An optimization method:

Ex: SGD (momentum), Adam

$$\theta := \theta - \frac{\alpha}{B} \sum_{i=1}^{B} \nabla_\theta \ell\left(h_\theta\left(x^{(i)}\right), y^{(i)}\right)$$

Also needed (perhaps most importantly!): a large, high-quality training and evaluation dataset.

The deep learning pipeline: training



Deep learning libraries like pytorch, tensorflow (and needle!) own this entire system. Neat!

Course takeaways: technical skills

On one hand, I hope that you walk away with this course with:

Expanded mathematical toolbox. Appreciation for how we can apply mathematical techniques such as multivariable calculus, linear algebra, and optimization to solve interesting problems (ex: training exciting deep learning models).

Deep learning wisdom. How deep learning works under the hood (model architecture design, backpropagation, dataset curation). Its strengths (and weaknesses!). And: maybe when NOT to use deep learning!

Hands-on modeling experience. Experience on how to effectively apply deep learning to solve nontrivial problems

Course takeaways: ..and more

There are other non-technical things I hope you can take with you too!

AI/ML, deep learning "literacy". In a world increasingly affected by ML/AI, it's good to be informed about what exactly deep learning's capabilities are.

Confidence and resilience. We covered some very technically complex and challenging topics in this course. I hope that you feel more confident in your ability to tackle challenging problems in whatever field/discipline you choose to pursue!

General advice

The following slides are some general, broad advice that I wish I could tell my past self

Advice on working on your own ML/DS projects

(Ideally) Identify a project that you personally find interesting

- Example: if you enjoy music, work on an ML+music project!

(Or) do whatever project your manager/advisor tells you to do

These days, many fields are interested in applying ML/DS to their problems

- Example: medicine/health tech. According to a doctor I know: "It's easier for a ML/DS person to learn the medicine stuff for a modeling project, than it is for a doctor to learn the ML/DS/coding stuff."

The importance of a good story

Before embarking on a project, ask yourself: "What story am I trying to tell here?"

The best projects (academic, industry, or creative) are those that have a clear, simple, and captivating story to tell.

Tip: when you start a project, try to have a vision of how you want the project to go.

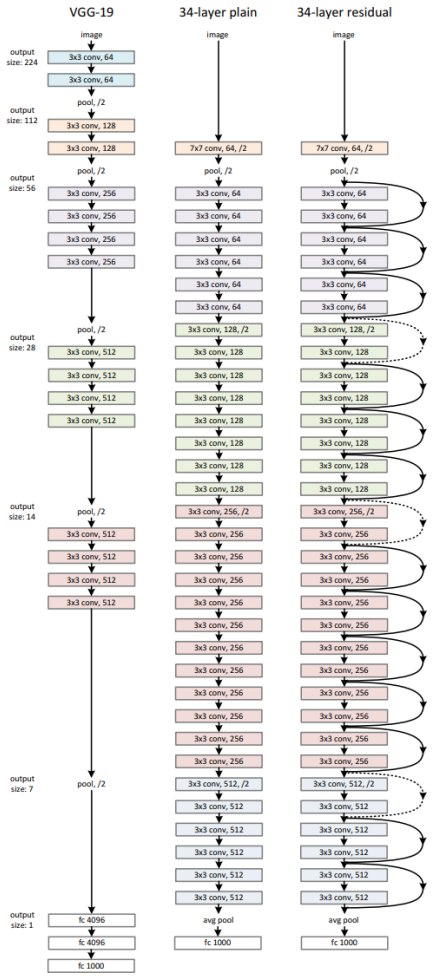
Things might not work out the way you originally envisioned (and it's good to be adaptable/flexible!), but it's immensely valuable to have an initial "guiding light".

Corollary: have strong communication skills!

Example: ResNet's story

The ResNet paper wasn't just: "We tried a bunch of Convnet architecture tweaks, and found that ResNet worked really well".

Instead, the story was: "Learning deep models is hard due to vanishing/exploding gradients. By adding residual (skip) connections, we can now train extremely deep models (>150 layers) and achieve state-of-the-art performance on ImageNet."



Training/eval dataset collection

This is the most important part!

Ideally: you can find an existing dataset online

If not - ideally you can collect a dataset by, say, scraping some website or calling some API

- Ex: scraping Google reviews/website-text

Worst case: manually collect data via human annotations. Painful!

New Trend: have ChatGPT help collect a dataset for you?

Feature engineering

Important question: is there enough signal in the training data and input features for the task?

- If not - you should rethink your approach to the problem.
- This is where your human intuition (and ML wisdom) plays an important role.

Modeling + evaluation

Formulate the task. Is this a classification task? Regression? Recommendation?

- Ideally you should be able to identify a reasonable loss function

Formulate the evaluation pipeline. Which metrics will you (or can you) compute?

- Ex: accuracy/precision-recall-f1/MSE
- Eval metrics should be
  - (1) Easy to compute
  - (2) Easy to interpret

Model architecture

Advice: start with the simplest model (eg logistic regression for classification) to establish a baseline.

Using your baseline, run the entire train+eval pipeline end-to-end. Do things look correct?

- Aka a "sanity check" for your ML pipeline
- Ex: can you overfit on a single training example? If you can't, something's wrong with your pipeline.

Once you're convinced things look right, then proceed with model architecture iteration (ie training fancier models)

Model experiment advice

Learning how to debug ML models is a skill gained by practice and experience

Standard software bugs are obvious (eg Exceptions/errors)

ML bugs are more subtle: rather than outright errors, they instead manifest as poor performance or unstable training

- This is an art + skill!

Google has decent guides: "The Rules for ML" [link], "Tuning Playbook" [link]

Industry vs Academia?

When I was an undergrad (2007-2012), the two routes seemed completely disjoint and incompatible.

Academia: work on exciting state-of-the-art research + interesting problems. Get to teach and mentor others.

Industry: work on projects (that may not be exciting to me) to make CompanyX more money.

- Aka feel like a tiny cog in a giant machine.

Industry vs Academia?

Now (2026): I feel strongly that there is a lot of interesting, exciting, impactful work happening in industry, especially in AI/ML/DS.

Ex: tech companies need good recommendation systems, which often boils down to learning good representations by training large models on large datasets in scalable ways.

- Ex: Google/Meta/Pinterest/Amazon/etc

The big tech companies often have research labs doing state-of-the-art research

- Ex: Google DeepMind, Facebook AI Research ("FAIR"), Open AI, Pinterest ATG
- ...and more!

And: you get paid much more in industry (ha)

Industry AND academia?

Remember: you don't have to choose one or the other. You can try out both!

You can always apply to grad school after working for a few years!

Industry: IC vs Manager

Two tracks:

- IC: "Individual Contributor". Ex: software developers, data scientists/analysts, etc (aka "not manager").
- Management
  - Engineering manager ("EM")
  - Program/project manager ("PM, TPM")

Managers

Manager's goal: effectively utilize their team to work on the "right" projects that delivers the best impact to the company.

Lots of planning to ensure that everyone is working on projects that aligns with what the company needs.

Lots of communication across teams to ensure that projects are running smoothly ("cross-functional")

One philosophy I've heard: "The manager's main job is to keep their direct reports happy, and to get them promoted."

- Aka "People manager"

(I personally don't know as much about this world, how they are evaluated, etc)

Switching tracks

It's possible to switch from, say, an IC to a Manager (and back again!)

- Some people enjoy one or the other. Both are valid!

Note: traditionally (ie before tech companies like Google), the wisdom was that one needed to become a manager to "climb the ladder"

However: in many newer tech companies, it's perfectly fine to remain an IC, as there's "plenty of ladder" to climb as an IC.

Industry: advice

Always aim to learn and grow (especially in your first 5 years!)

Many people advise job-hopping every 3-4 years to maximize compensation ($). Instead, I have a different philosophy: maximize learning + growth.

If you feel like you're stagnant in a role, proactively speak to your manager about this. If nothing changes and you feel that there isn't a good road forward, strongly consider looking for another role (either at the same company, or at a different company)

PhD advice: should I do a PhD?

You should do a PhD only if you love your field, you have done research before and really enjoyed it, and you are OK with working on a single project in a single field for 4-6 years.

IF you're ok with the above, then my single biggest advice is: make sure that you are compatible with your PhD advisor!

- Your PhD advisor will make or break your PhD experience. There are many horror stories online of PhD student's lives made miserable by an advisor that was a poor fit.

Cautionary tale: the PhD attrition rate is ~50%.

How did I get into AI/ML? A brief bio

(2007) Entered UC Berkeley as "Undeclared - Physics" (L&S)

- Fun fact: my high school college advisor told me UC Berkeley would be a "far reach" school for me (I didn't have stellar grades in HS)
- I took Physics 7A, disliked it, and decided not to go further in Physics
- Considered majoring in Music, took several Music classes (shoutout to Prof. David Pereira, his harmony / music theory courses were great)
- Frankly, I was an unfocused student that didn't do very well my first 2 years, as (other than Music) I wasn't super inspired by my courses.

My intro CS journey

(2nd year, Fall 2008) On a whim, I took my first CS programming course, "CS 3L", aka a pre-CS61A course. I liked it enough to consider taking more CS courses.

(2nd year, Spring 2009) I took CS 61A (w/ Prof. Brian Harvey). This class was HARD! I didn't do that great in the course, but it intrigued me a little.

(2nd year, Summer 2009) I took CS 61B (w/ Prof. Colleen Lewis), and this was the class that "clicked" for me and sparked my love for CS.

- The teaching staff was incredible also, and sparked my initial interest in teaching.

CS: TA-ing

(3rd year Summer 2010) I TA'd for my first time for CS 61BL. Great experience! I went on to TA for: CS61A and CS3L.

Fun fact: I TA'd for Prof. John DeNero the first time he taught at UC Berkeley! That semester, he redesigned cs61A from Scheme to Python.

- The "Ants" project was created that semester!

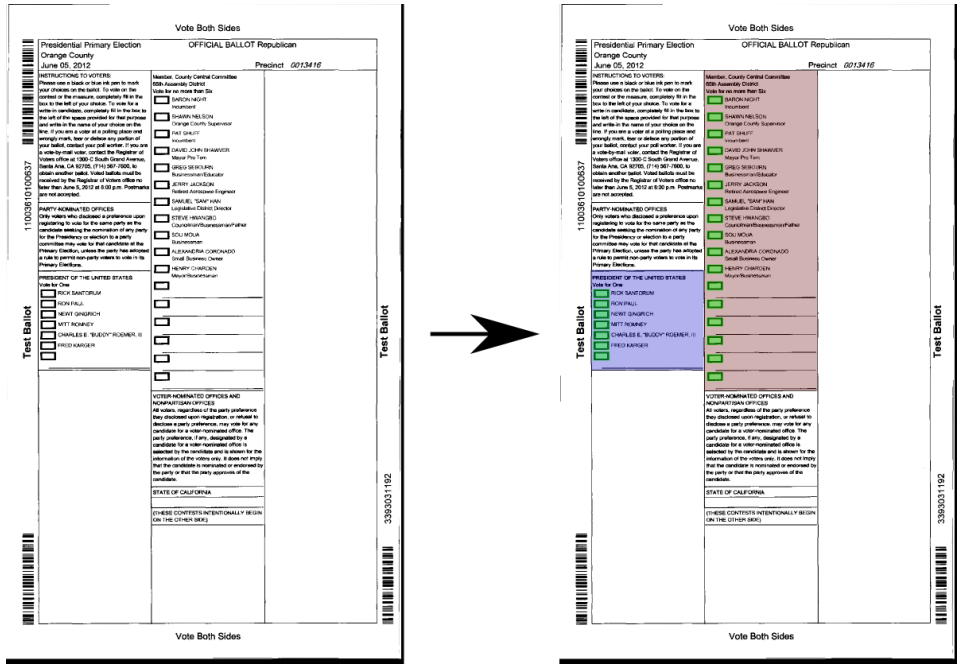
CS: undergrad research

(3rd year Fall 2010) I started undergraduate research work with Prof. David Wagner and PhD student Arel Cordero + Kai Wang.

I took Prof. Wagner's CS 161 (Computer Security), really liked it, and emailed him about undergrad research opportunities. He referred me to Arel!

Topic: election technology. Build an open-source ballot counter to streamline ballot counting in an open, verifiable way. ("OpenCount")

- Image processing / computer vision algorithms, ex: image alignment, filled-in mark detection, template matching, etc



CS: introduction to ML

(~4th year 2011) Through my ugrad research, I got exposed to computer vision and image processing. I loved the idea of teaching computers how to "see" the world. My grad student mentor showed me a few neat computer vision papers, and I was hooked!

(5th year Fall 2011) I took the CS department's first version of CS189, "Introduction to Machine Learning" (then known as CS194-10). I loved it!

(5th year Fall 2011) I graduated as a "super senior". Hooray!

PhD applications: Attempt 1

(Fall 2011) At the time, my attitude was "PhD or bust"

- I wanted to teach, and I wanted to do neat research in computer vision

Problem: while I had a few research publications, my undergrad GPA was not good, and my Math/Quant GRE scores were not quite high enough (oops).

First attempt: No acceptances!

I was very demoralized and extremely disappointed. This was the tough time.

Fortunately, I was able to continue working with Prof. Wagner as a "full time researcher". I stayed in Berkeley for another year, and I planned to apply for next year's application cycle.

PhD applications: Attempt 2

(Fall 2012) I again applied for PhD programs for computer vision programs

Second attempt: No acceptances!*

But: while I didn't get into UCLA's PhD program, I did get into their MS program.

I visited UCLA, loved the campus + computer vision people, and chose UCLA!

After a (challenging, yet rewarding!) 3 year MS, I joined Pinterest's Visual Search team as an ML engineer.

As of 2026 I'm still at Pinterest and still loving it!

I also teach part-time (usually 1 class a year), which I find very fulfilling.

Takeaway: regarding challenges

Not everyone's academic/professional journey is straightforward, clean, and easy. Mine certainly wasn't!

Success (or failure!) is never a sure thing!

You will face difficult times. There will be ups and downs, flagging motivation and morale, etc.

Being able to effectively (and healthily) navigate these difficult periods will help you enormously!

Takeaway: keep learning and growing!

If you prioritize learning, growth, and doing good work, you will end up in a good place.

It may not be where you originally intended, and that's 100% OK.

Thank you course staff!

TA's

Andria Xu SHE/HER
andria.xu@berkeley.edu
About Me: Hello! I'm Andria, a current junior from the Bay Area. I'm very excited to be teaching Data 188 this semester! Previously, I taught Data C88C. I also do research in reinforcement learning! In my free time, I enjoy hiking, playing board games, and trying random new things. Looking forward to getting to know you all! :D

Rebecca Dang SHE/HER
rdang@berkeley.edu
About Me: Hey there! I'm a 5th Year MS EECS student and I'm excited to teach DATA 188 for the first time! In the past, I've taught DATA C88C and DATA 101. In my free time, I love fangirling over the most recent book/TV show/movie I'm obsessed with, hiking, and playing guitar. Come talk to me about anything :)

Terry (Taehan) Kim
terry.kim@berkeley.edu
About Me: Hi! I'm Terry (Taehan) Kim, a senior majoring in Computer Science and a 🐵 and 🍔lover. Super excited for this semester!

Zekai Wang HE/HIM/HIS
zekai.wang@berkeley.edu
About Me: Hi! I'm Zekai, a 4th year CS and Applied Math major. I am interested in machine learning and robotics. Looking forward to meeting you all!

Tutors

Ryan Michael Tom HE/HIM
ryantom@berkeley.edu
About Me: Hi everyone! I'm Ryan, a 3rd year EECS major who enjoys playing basketball, running, raving, and adding new places to Beli! Very excited to meet you all :)

Yihang Chen HE/HIM
yhc0720@berkeley.edu
About Me: Hi! I'm Yihang, a senior majoring in Computer Science and Data Science. I'm thrilled to be part of the Data 188 course staff and look forward to working with students throughout the semester. In my free time, I enjoy snowboarding and spending time outdoors.

Learn more about your TA's + Tutors here: https://data-188-berkeley.github.io/sp26/staff/

AMA

Questions Ed post: [link]

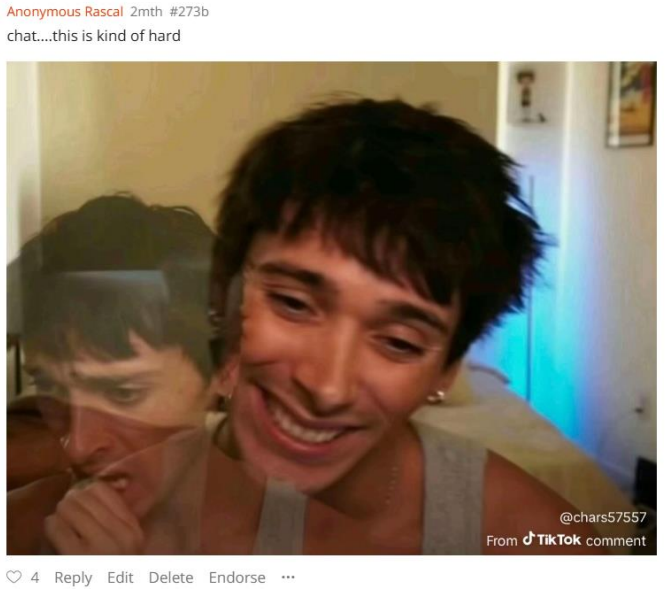
How did you decide to stay on the individual contributor track instead of moving into engineering management? Have you had any moments where you second-guessed that decision?

What's the biggest professional risk you've taken, and what made you decide to take the leap despite the uncertainty? Looking back, how did it shape where you are today?

(Anonymous Diva)

which meme of mine is your favorite so far:

Runner up:



Hi professor! Against the backdrop of the rapid development of artificial intelligence, how should we explore our future career paths and avoid being replaced by AI?

In future iterations of this course, would you consider dedicating more time to other relevant dimensions of the LLM/generative AI conversation (e.g. the sidelining of critical thinking in favor of "efficiency/productivity", the deeply questionable ethics of scraping the "entire Internet", the shortsightedness of eliminating certain job positions in favor of AI, the "AI Ouroboros", the exploitation of "ghost workers" and the general public's perception of how AI functions, etc)?

What made you want to go into education and teach ML?

What would you want to pursue in the future--continue in teaching, industry, or both?