# EECS 545: Machine Learning

# Lecture 27: More ML Advice & Panel Discussions

Honglak Lee and Michał Dereziński

4/18/2022

# Outline

- Resources for further learning
- Advice for research and careers in machine learning
  - How to read papers
  - ML career advice
  - Best practices for research (PhD student perspectives)

**Please ask your question here!**

https://forms.gle/zDnGaNQDYLNnQX7i8

# How to read papers

- Complete list of papers
  - Conferences (NeurIPS / ICML / ICLR / etc.), Twitter, Medium, Reddit, blog posts, friends
- Skip around the list
  - Initially skim and quickly check the high-level idea of the papers
- Go into more depth into more relevant/important papers
- Update paper list (cited papers, follow ups, etc.)
- Go into deeper into more details for relevant ones

Source: Andrew Ng

# Reading a paper

Do multiple passes

- Title / Abstract / Figures
- Intro / Conclusions / Figures + skim the rest
    - (Skim related work)
- Read but skip/skim math
- Read math and try to understand the key technical ideas
- Read the whole paper but skip parts that don’t make sense

Source: Andrew Ng

# Reading a paper

Try to answer these questions while reading paper

- What did the authors try to accomplish?
- What were the key elements of the approach?
- What can you use yourself?
- What other references do you want to follow?

Source: Andrew Ng

# Reading a paper: Technical Details

Math

- Rederive math from scratch

Code

- Run open-source code
- Read the details of the code
- Reimplement from scratch

Source: Andrew Ng

# Reading paper: Consistency is important

Steady, consistent reading N papers a week is much better than reading lots of papers in short bursts.

This is true for any learning process!

Source: Andrew Ng

# ML Career Advice

Short-term Goal:

- Job (big company or startup)
- MS
- PhD

Overall, you want to do important work. A few specific questions:

1. How to get jobs?
2. Selecting a position

Source: Andrew Ng

# ML Career Advice

Recruiters look for:

- Technical skills (ML knowledge and skills: quiz, coding, etc.)
- Meaningful / significant work
    - Making things work
    - Applications
    - Show evidence that you can do important work

Source: Andrew Ng

# ML Career Advice

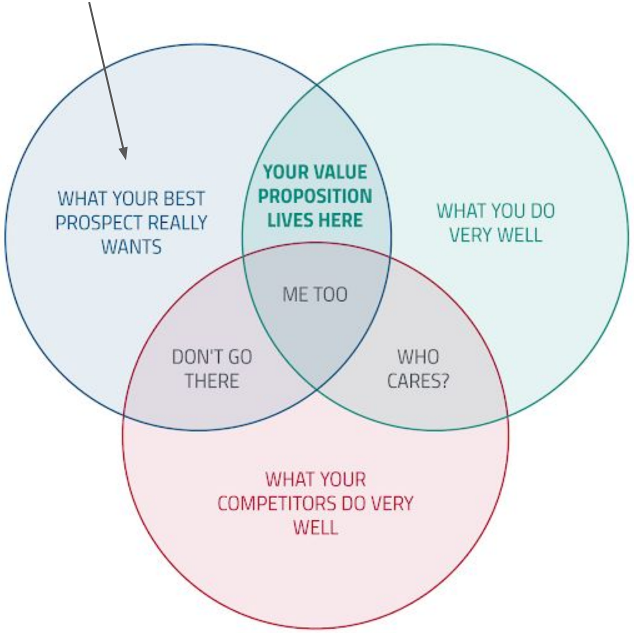
## Unique Value Proposition (UVP)

- a clear statement that describes the benefit of your “expertise and skills”, how you solve your customer's needs and what distinguishes you from others (e.g., “competition”).

- “Magic Power”

Examples of prospects / customers:

- Company
- Research Community
- Academia/Schools




http://duffy.agency/insight/how-to-write-a-compelling-value-proposition/

# ML Career Advice: Breadth vs Depth

Ideally you want to have **T-shaped expertise** (both breadth and depth)

**Breadth**:

- ML Areas: ML, DL, Graphical models, NLP, Computer Vision, etc.
- Breadth within DL: NN, Optimization, CNN, RNN/LSTM, Unsupervised Learning, Generative Models, RL, etc.

**Depth**:

- Projects
- Open-source
- Research
- Internship

Source: Andrew Ng

# ML Career Advice: Breadth vs Depth

Ideally you want to have **T-shaped expertise** (both breadth and depth)

**Non-ideal cases:**

- Only breadth but no depth
- Only depth but no breadth
- Breadth + many small projects without significant results

Source: Andrew Ng

# ML Career Advice: Breadth vs Depth

**Breadth**: Foundational skills

- Coursework
- Reading papers

**Depth**:

- Focused work on relevant projects
- Try your best to achieve significant results
    - solving problems, new applications, improving performance, open-source code, publishing papers, etc.

Source: Andrew Ng

# Selecting a job

**Key factor**: Work with great people on important projects

- Focus on the team that you will interact with (10-30 people)
- Manager
- Not on “brand”

Ultimately, your ideal end-goal (career-wise) should be making impact.

**Other tips:**

- Talk to people; do internships

Source: Andrew Ng

# Selecting a job: decision making

- List key aspects/criteria of the job (some examples below but not limited to)
    - Team (colleagues, manager, etc.)
    - Projects
    - Potential impact
    - Growth potential (company, team, etc.)
    - Learning (personal growth)
    - Compensation
    - Family
    - Work-life balance
    - Culture
- Rate the importance of these criteria (1-10 scale)
- Rate individual scores for different job options
- Calculate the weighted average

# Kevin, Aabhaas, Sudeep – more resources

- At Michigan:
    - Check out Michigan AI to find profs and PhD students in AI
    - Other ML classes include 542 (computer vision), 543 (knowledge-based systems), 567 (robotics), 576 (data mining), 592 (foundations of AI), 595 (NLP)
        - as well as many special topics courses!
    - Reading groups:
        - Computer vision: Computer Vision Reading Group
        - NLP: NLP Reading Group | Language Information and Technologies
        - RL: RL Reading Group
        - Theory: ML theory reading group
    - Student orgs (mostly undergrad-focused):
        - Michigan Student AI Lab: MSAIL
        - Michigan Data Science Team: Michigan Data Science Team
        - AI Teas (UMich AI Lab)

# Kevin, Aabhaas, Sudeep – more resources

- At other universities:
  - Stanford: List of AI courses, including CS230: Deep Learning, CS330: Deep Multi-Task and Meta Learning
  - Carnegie Mellon: Machine Learning Core Courses (you can often look these up and find course info including lectures)
  - Berkeley: Deep Unsupervised Learning, Simons Institute programs (this one on deep learning is supposedly pretty good)
  - MIT: Machine Learning at MIT -- Classes; there are also 1-month seminars like this one on NTKs that can be useful
- These universities are good about “open-sourcing” their classes, but there are classes from other universities that are also good

# Kevin, Aabhaas, Sudeep – more resources

- Non-academic:
  - Awesome Machine Learning, a curated list of ML resources in all different programming languages
  - For the casual explorer, browsing Medium articles is useful too
- New in Machine learning:
  - Youtube: Yannic Kilcher, Machine Learning Street Talk, StatQuest, etc.
  - Conferences: ComputerVisionFoundation Videos, ICML, ICLR
  - Blogs: Lil'Log, inFERENCe, Off the convex path
- But devil is in the details:
  - Code things up to understand better.
  - You can contribute to open source projects of your choice.

# Advice / Best practices for research (PhD student perspectives) - Junghwan

- Learn the standards used in your area as early as possible
  - Software Frameworks and Tools
  - Model and data
  - Mathematical techniques, Technical standards
- Pick up the non-standards consistently
  - Unexpected synthesis of ideas
- Learn to recognize significance and relevance of results
- Talk to people and learn from them

# Advice / Best practices for research (PhD student perspectives) - Yijie

- Keep track of random ideas, related papers, experimental results, codes in an organized way
  - Docs, slides, github repositories
  - Back and forth, change of directions in the research process
  - Accumulated materials are helpful in the paper writing
- Start a project with simple sanity-check experiments and move to more complicated scenarios step by step
  - Sanity-check experiments may help avoid wasting time in the long term
  - It's hard to analyze or find the errors if we move too aggressively
- Practice writing and presentation skills ("deliberate practice")
  - Take every opportunity to improve the communication skills (e.g. regular meetings, informal discussions, oral presentations, etc)
  - Clear explanation about research work
  - Deep discussions with the colleagues and feedback from audience are super helpful

# Advice / Best practices for research (PhD student perspectives) - Yijie

- Read relevant papers
  - Avoid reading the literatures aimlessly
  - Focus on a research problem and read the relevant literatures
  - Organize papers in groups
  - Find relations among papers (e.g. C is combination of A and B, E is extension of D)
  - Big picture of the research topic
- Set up automation for experiments
  - Run a large number of experiments for hyper-parameter search
  - Data visualization and analysis

# Advice / Best practices for research (PhD student perspectives) - Anthony

Plan by making ***small tangible goals*** that make progress towards your larger goal (research, startups, etc.)

- Research is complex, and your plan is likely to change along the way
- Making many small steps every amounts to large steps over time

# Advice / Best practices for research (PhD student perspectives) - Anthony

Plan by making ***small tangible goals*** that make progress towards your larger goal (research, startups, etc.)

- Research is complex, and your plan is likely to change along the way
- Making many small steps every amounts to large steps over time

Do this by solving ***simpler versions of your problem***

- Make prototypes! You can make prototypes of any idea
- E.x.
    - Use a smaller dataset / environment
    - Assume more things about the problem that make it easier
    - Use a simpler model architecture at first

# Advice / Best practices for research (PhD student perspectives) - Anthony

***Code quality matters!*** Coding for ML is still software engineering.

- Your methods are going to change!
  - Different architectures / hyperparameters
  - Different datasets
  - Different assumptions about the problem
- Need to change your code *frequently*
- Learn to use good software practices --- Version Control, comments, modular code, etc.
- How to run and analyze your code is also important to keep clean
  - Job scheduling your model training
  - Automatically visualizing your model results

# Advice / Best practices for research (PhD student perspectives) - Jongwook

One of my favorite lessons that I learned:

***"Research is an iterative process."***

- Rome wasn't built in a day. Same for your research (and your engineering work)
  - The problem, the method, the main claim, etc. may all change. There are a lot more things that end up not being included in the final manuscript.
- Recommendations:
  - In the early stages, try iterating quick and dirty – it usually helps.
    First make things work, and then refactor/restructure/improve as needed.
  - Fast iterations and try out as many things as possible,
    because time is limited in research. We should conquer the *easiest* settings first!
  - Build your own accustomed, efficient workflow of experimentation.
  - Talk to your mentors, collaborators, peers, friends (and even rubber ducks!)
    much and frequently, so you can collect feedback and helpful inputs.

# Advice / Best practices for research (PhD student perspectives) - Jongwook

***Leverage tools efficiently and effectively:***

- Learn how to use your favorite editor (vim, emacs, etc.), IDE (pycharm, VSCode, etc.), command line tools, etc. so as to minimize redundant and repetitive jobs, and do the cognitively simple thing as quickly as possible.

- Practice how to write good codes and how to use the right tool for the job
  - Often overlooked are how to do interface-driven design, object-oriented programming, principles for separating concerns and responsibility, even in programming for research.
  - Maintain a good, sustainable codebase so you can focus solely on your idea and research. See also: Hidden Technical Debt in Machine Learning Systems
- Make a deliberate practice to improve such skills.

# Advice / Best practices for research (PhD student perspectives) - Jongwook

***Leverage tools efficiently and effectively:***

- Keep a good habit of maintaining your own journal, archive, scrapbook, memo about great papers/articles/resources.
  - Write your own summary, notes, comments, critical thoughts about the paper while reading, so you can find, recall, and recap them quickly in the future.
  - Make them maintained and searchable (but w/ minimal effort).
    - E.g. note taking -- Obsidian, Dynalist, Notion, Notable, Evernote, etc.
    - E.g. library: Mendeley, Zotero, Readcube, Papers, …

Thank you for taking this course
We hope that you learned a lot!

We wish you the best of luck
for your future!

Please ask your questions: