John Schulman's Homepage (/index.html)

An Opinionated Guide to ML Research

Posted on 2020/01/24
← back to blog index (/blog.html)

I originally wrote this guide in back in December 2017 for the OpenAI Fellows program (https://openai.com/blog/openai-fellows/)

In this essay, I provide some advice to up-and-coming researchers in machine learning (ML), based on my experience doing research and advising others. The advice covers how to choose problems and organize your time. I also recommend the following prior essays on similar topics:

- *You and Your Research* by Richard Hamming (http://www.cs.virginia.edu/~robins/YouAndYourResearch.html)
- *Principles of Effective Research* by Michael Nielsen (http://michaelnielsen.org/blog/principles-of-effective-research)

My essay will cover similar ground, but it's more tuned to the peculiar features of ML.

The keys to success are working on the right problems, making continual progress on them, and achieving continual personal growth. This essay is comprised of three sections, each covering one of these topics.

Exercise. Before continuing, it's useful to spend a few minutes about which findings and achievements in ML have been most interesting and informative to you. Think about what makes each one stand out—whether it's a groundbreaking result that changed your perspective on some problem; or an algorithmic idea that's reusable; or a deep insight about some recurring questions. You should aspire to produce results, algorithms, and insights of this caliber.

Choosing Problems

Honing Your Taste

Your ability to choose the right problems to work on is even more important than your raw technical skill. This taste in problems is something you'll develop over time by watching which ideas prosper and which ones are forgotten. You'll see which ones serve as building blocks for new ideas and results, and which ones are ignored because they are too complicated or too fragile, or because the incremental improvement is too small.

You might be wondering if there's a way to speed up the process of developing a good taste for what problems to work on. In fact, there are several good ways.

1. Read a lot of papers, and assess them critically. If possible, discuss them with others who have a deeper knowledge of the subject.

2. Work in a research group with other people working on similar topics. That way you can absorb their experiences as well as your own.
3. Seek advice from experienced researchers on what to work on. There's no shame in working on ideas suggested by other people. Ideas are cheap, and there are lots of them in the air. Your skill comes in when you decide which one to work on, and how well you execute on it.
4. Spend time reflecting on what research is useful and fruitful. Think about questions like
    a. When is theory useful?
    b. When are empirical results transferable?
    c. What causes some ideas to get wide uptake, whereas others are forgotten?
    d. What are the trends in your field? Which lines of work will make the other ones obsolete?

Items 1-3 relate to optimizing your environment and getting input from other researchers, whereas item 4 is something you do alone. As empirical evidence for the importance of 1-3, consider how the biggests bursts of impactful work tend to be tightly clustered in a small number of research groups and institutions. That's not because these people are dramatically smarter than everyone else, it's because they have a higher density of expertise and perspective, which puts them a little ahead of the rest of the community, and thus they dominate in generating new results. If you're not fortunate enough to be in an environment with high density of relevant expertise, don't despair. You'll just have to work extra-hard to get ahead of the pack, and it's extra-important to specialize and develop your own unique perspective.

Idea-Driven vs Goal-Driven Research

Roughly speaking, there are two different ways that you might go about deciding what to work on next.

1. Idea-driven. Follow some sector of the literature. As you read a paper showing how to do X, you have an idea of how to do X even better. Then you embark on a project to test your idea.

2. Goal-driven. Develop a vision of some new AI capabilities you'd like to achieve, and solve problems that bring you closer to that goal. (Below, I give a couple case studies from my own research, including the goal of using reinforcement learning for 3D humanoid locomotion.) In your experimentation, you test a variety of existing methods from the literature, and then you develop your own methods that improve on them.

Of course, these two approaches are not mutually exclusive. Any given subfield ML is concerned with some goals (e.g., object detection). Any "idea-driven" project will represent progress towards the subfield's goals, and thus in a sense, it's an instance of goal-driven research. But here, I'll take goal-driven research to mean that your goal is more specific than your whole subfield's goal, and it's more like *make X work for the first time* than *make X work better*.

I personally recommend goal-driven research for most people, and I've mostly followed this strategy myself.

One major downside of idea-driven research is that there's a high risk of getting scooped or duplicating the work of others. Researchers around the world are reading the same literature, which leads them to similar ideas. To make breakthroughs with idea-driven research, you need to develop an exceptionally deep understanding of your subject, and a perspective that diverges from the rest of the community—some can do it, but it's difficult.

On the other hand, with goal-driven research, your goal will give you a perspective that's differentiated from the rest of the community. It will lead you to ask questions that no one else is asking, enabling you to make larger leaps of progress. Goal driven research can also be much more motivating. You can wake up every morning and imagine achieving your goal—what the result would look like and how you would feel. That makes it easier to stick to a long-running research program with ups and downs. Goals also make it possible for a team of researchers to work together and attack different aspects of the problem, whereas idea-driven research is most effectively carried out by "teams" of 1-2 people.

Case Study of Goal-Driven Research: My Work During Graduate School

For the first half of my PhD, my goal was to enable robots to manipulate deformable objects—including surgical robots tying knots, and household robots folding clothes. While this goal was determined by my advisor, Pieter Abbeel, as the main goal for his lab, I developed my own opinion on how to achieve this goal—my approach was based on learning from human demonstrations, and I was going to start with the problem of getting the PR2 to tie knots in rope. Various unexpected subproblems arose, one of which was trajectory optimization, and my work on that subproblem ended up being the most influential product of the knot-tying project.

For the second half of my PhD, I became interested in reinforcement learning. While there are many problem domains in which reinforcement learning can be applied, I decided to focus on robotic locomotion, since the goal was concrete and the end result was exciting to me. Specifically, my goal was to get a 3D robot to learn how to run from scratch using reinforcement learning. After some initial exploration, I decided to focus on policy gradient methods, since they seemed most amenable to understanding and mathematical analysis, and I could leverage my strength in optimization. During this period, I developed TRPO and GAE and eventually achieved the original goal of 3D humanoid locomotion.

While I was working on locomotion and starting to get my first results with policy gradient methods, the DeepMind team presented the results using DQN on Atari. After this result, many people jumped on the bandwagon and tried to develop better versions of Q-learning and apply them to the Atari domain. However, I had already explored Q-learning and concluded that it wasn't a good approach for the locomotion tasks I was working on, so I continued working on policy gradient methods, which led to TRPO, GAE, and later PPO—now my best known pieces of work. This example illustrates how choosing a different problem from the rest of the community can lead you to explore different ideas.

Goal Driven Research: Restrict Yourself to General Solutions

One pitfall of goal-driven research is taking your goal too literally. If you have a specific capability in mind, there's probably some way to achieve it in an uninteresting way that doesn't advance the field of machine learning. You should constrain your search to solutions that seem general and can be applied to other problems.

For example, while working on robotic locomotion, I avoided incorporating domain information into the solution—the goal was to achieve locomotion in simulation, *in a way that was general and could be applied to other problems*. I did a bit of feature engineering and reward shaping in order to see the first signs of life, but I was careful to keep my changes simple and not let them affect the algorithm I was developing. Now that I am using videogames as a testbed, I make sure that my algorithmic ideas are not specific to this setting—that they equally well could be applied to robotics.

Aim High, and Climb Incrementally Towards High Goals

Sometimes, people who are both exceptionally smart and hard-working fail to do great research. In my view, the main reason for this failure is that they work on unimportant problems. When you embark on a research project, you should ask yourself: how large is the potential upside? Will this be a 10% improvement or a 10X improvement? I often see researchers take on projects that seem sensible but could only possibly yield a small improvement to some metric.

Incremental work (those 10% improvements) are most useful in the context of a larger goal that you are trying to achieve. For example, the seminal paper on ImageNet classification using convolutional neural networks (Krizhevsky, Sutskever, & Hinton, 2012) does not contain any radically new algorithmic components, rather, it stacks up a large number of small improvements to achieve an unprecedented result that was surprising to almost everyone at the time (though we take it for granted now). During your day-to-day work, you'll make incremental improvements in performance and in understanding. But these small steps should be moving you towards a larger goal that represents a non-incremental advance.

If you are working on incremental ideas, be aware that their usefulness depends on their complexity. A method that slightly improves on the baseline better be very simple, otherwise no one will bother using it—not even you. If it gives a 10% improvement, it better be 2 lines of code, whereas if it's a 50% improvement, it can add 10 lines of code, etc. (I'm just giving these numbers for illustration, the actual numbers will obviously depend on the domain.)

Go back and look at the list of machine learning achievements you admire the most. Does your long-term research plan have the potential to reach the level of those achievements? If you can't see a path to something that you'd be proud of, then you should revise your plan so it does have that potential.

Making Continual Progress

To develop new algorithms and insights in machine learning, you need to concentrate your efforts on a problem for a long period of time. This section is about developing effective habits for this long-term problem solving process, enabling you to continually build towards great results.

Keep a Notebook, and Review It

I strongly advise you to keep a notebook, where you record your daily ideas and experiments. I have done this through 5 years of grad school and 2 years at OpenAI, and I feel that it has been tremendously helpful.

I create an entry for each day. In this entry, I write down what I'm doing, ideas I have, and experimental results (pasting in plots and tables). Every 1 or 2 weeks, I do a review, where I read all of my daily entries and I condense the information into a summary. Usually my review contains sections for *experimental findings*, *insights* (which might come from me, my colleagues, or things I read), *code progress* (what did I implement), and *next steps / future work*. After I do my week in review, I often look at the previous week to see if I followed up on everything I thought of that week. Also, while doing this review, I sometimes transfer information into other sources of notes. (For example, I keep a list of backburner ideas and projects, separate from my notebook.)

What's the value in keeping this notebook and doing the regular reviews?

First, the notebook is a good place to write down ideas as soon as you have them, so you can revisit them later. Often, when I revisit my journal entries during the week in review, I'll fill in a missing piece in a puzzle, which didn't occur to me at the time.

Second, the notebook helps you keep your experimental results in a unified place, so you can easily find the results later. It's easy to forget about your conclusions, e.g., which hyperparameters made a difference, and you'll want to revisit your old notebook entries.

Third, the notebook lets you monitor your use of time. You might wonder "where did last week go?", and the notebook will help you answer that question. You might be disappointed with your throughput and realize you need to work on your time management. You also might look back at several months and realize that you've been jumping around between ideas too much—that you have a few half-finished projects but you didn't follow any of these threads long enough to yield a notable result.

When to Switch Problems

To solve a challenging problem, you need to spend a sufficient amount of time on it. But in empirical machine learning research, it's hard to know if you've tried an idea hard enough. Sometimes the idea has the potential to work, but if you get one detail wrong, you'll see no signs of life. But other ideas are simply doomed to fail no matter how hard you work on them.

In my experience, switching problems too frequently (and giving up on promising ideas) is a more common failure mode than not switching enough. Often, while you're engaged in the long slog towards getting your current idea to work, another promising idea will come along, and you'll want to jump to that idea. If your idea is quick to try and the potential upside is large, then go ahead and do it. But more commonly, your initial results on the new idea will be disappointing, and it'll take a more sustained effort to yield significant results.

As a rule of thumb, when you look back at which projects you've been working on over a period of months, you should find that there have been lots of small dead ends, but the majority of your time has been directed towards projects that yielded a deliverable such as a paper or a blog post. If you look back at your time and see that a substantial fraction was spent on half-finished projects—which were not definite failures, but which you abandoned in favor of some newer idea—then you should make a stronger effort towards consistency and follow-through in the future.

One strategy, which I haven't tried personally but makes a lot of sense upon reflection, is to devote some fixed time budget to trying out new ideas that diverge from your main line of work. Say, spend one day per week on something totally different from your main project. This would constitute a kind of epsilon-greedy exploration, and it would also help to broaden your knowledge.

Personal Development

No matter how you allocate your time during your research journey, you are bound to learn a lot. Each project will present new challenges, and you can pick up the background material and skills as you go along. However, you can significantly improve your chances to do great work in the long term by regularly setting aside time for your personal development. Specifically, you should allocate some fraction of your time towards improving your general knowledge of ML as opposed to working on your current project. If you don't allocate this time, then your knowledge is likely to plateau after you learn the basics that you need for your day-to-day work. It's easy to settle into a comfort zone of methods you understand well—you may need to expend active effort to expand this zone.

The main ways to build your knowledge of ML are to read textbooks, theses and papers; and to reimplement algorithms from these sources. Early on in your career, I recommend splitting your time about evenly between textbooks and papers. You should choose a small set of relevant textbooks and theses to gradually work through, and you should also reimplement the models and algorithms from your favorite papers.

Most students of machine learning don't spend time reading textbooks after they finish their school courses. I think this is a mistake, since textbooks are a much more dense way to absorb knowledge than papers. Each conference paper typically contains one main new idea, along with a background section that's too concise to learn anything from. There's a lot of overhead, since you typically need to spend more time understanding the notation and terminology than the idea itself. On the other hand, good textbooks collect decades of ideas and present them in the proper order with the same notation. Besides reading the introductory machine learning textbooks, read other books in your areas of interest. A couple of my favorites were *Numerical Optimization* by Nocedal & Wright, and *Elements of Information Theory* by Cover & Thomas.

Besides textbooks, I recommend reading PhD theses of researchers whose work interests you. PhD theses in ML usually are ordered as follows: (1) introductory and background material, (2) several papers that were previously published at conferences (it's said that you just have to "staple together" your papers to write your thesis), and (3) a conclusion and outlook. You're likely to benefit most from parts (1) and (3), since they contain a unifying view of the past and future of the field, written by an expert. Recent theses are often the best place to find a literature review of an active field, but older theses also often contain valuable gems of insight.

Textbooks and theses are good for building up your foundational knowledge, but you'll also need to read a lot of papers to bring your knowledge up to the frontier. When you are just starting your research career, I recommend spending a lot of time reimplementing ideas from papers, and comparing your results to the published ones. First of all, this gives you a much deeper understanding of the topic than you'd get by passively reading. Second, you'll gain experience running experiments, and you'll get much quicker feedback by reimplementing existing work (where the desired level of performance is known) than by doing original research. Once you can easily reproduce the state-of-the-art, you'll be ready to go beyond it.

Besides reading seminal papers and reimplementing them, you should also keep track of the less exceptional papers being published in your field. Reading and skimming the incoming papers with a critical eye helps you notice the trends in your field (perhaps you notice that a lot of papers are using some new technique and getting good results—maybe you should investigate it). It also helps you build up your taste by observing the dependency graph of ideas—which ideas become widely used and open the door to other ideas.

Go forth and do great research!