Comprehensive Information Gatherings

1. April 2021 Deep Dive: Transformers and GPT-3
2. Alex Turner's Research, Comprehensive Information Gathering

April 2021 Deep Dive: Transformers and GPT-3

Introduction

I know very little about a staggering number of topics that would be incredibly useful for my research and/or navigating the field. Part of the problem is the sheer number of choices -- I can't make myself study one thing very long because I always feel like I need to learn 20 other things.

To solve this problem, I started to implement monthly deep dives into a topic. A month is short enough that even I can stay relatively on track for that long, while still being enough time to actually learn something. The goal is not to master the topic completely (which would be impossible); it's to get a finer map of the territory, and to be able to discuss relevant ideas on this topic.

This first month was dedicated to transformers and the GPT-3 model, a topic I felt like I had to do, but which actually kind of grew on me.

Note that this post is a very quickly written summary of what I did and how it went (a bit like TurnTrout's sequence, with probably less insights). This is not a distillation post, and if you read it, you will not learn that much about the subject. That being said, it might prove useful if you want to go learn it by yourself.

Thanks to Jérémy, Flora, Connor, Kyle and Laria for great discussions that helped me understand this topic further.

The Plan

I based myself quite loosely on the structure advocated in Scott Young's Ultralearning. Which only means that I made a planning week by week, checked what resources were recommended beforehand, and tried to focus on the direct applications I wanted to make of my learning, which is explaining it to people and having interesting discussions on the subject.

My idea was that in order to understand GPT-3, I needed first to understand the Transformer architecture, then the GPT family, then play with GPT-3 directly. Since I started this deep dive on the 8th of April, the planning was broadly:

- Week of the 8th: Study transformers. Here are the resources I had in mind
  - Original paper (Attention is all you need)
  - Annotated version of original paper
  - Stack Overflow explanation of an apparently difficult point (Queries, keys and values)
  - Explanatory blog post 1
  - Explanatory blog post 2
  - Blog post on the historical significance of transformers (nostalgebraist)
  - Survey of attention mechanisms

    - Survey of transformer family
- Week of the 15th: Study the GPT family of models. Resources once again
    - Original papers
        - GPT
        - GPT-2
        - GPT-3
    - History
        - https://medium.com/walmartglobaltech/the-journey-of-open-ai-gpt-models-32d95b7b7fb2
        - https://lilianweng.github.io/lil-log/2019/01/31/generalized-language-models.html
- Week of the 22th: Play with GPT-3
    - AI Dungeon
    - Gwern's page on GPT-3 abilities
- Week of the 29th: Buffer week

I expected to take one hour per day, tentatively placed between 1pm and 2pm, just after my lunch.

The Reality

Week 1: Following the Plan

The first week I actually did my weekly hour, at the time scheduled, and I pretty much learned what I wanted to learned about how transformers worked.

My biggest hurdle in groking the original paper was the self-attention mechanism itself. With hindsight, I had two issues:

- Not having thought about NN architectures for so long, I had forgotten that the whole point was not the encode the information or even the specific mechanism that you wanted, but to create an affordance for the NN to learn the kind of things you care about. Remembering this helped me with getting that the NNs learned how to parameterize the queries, keys and values as was most useful for it.
- I couldn't find where the inputs where used in the mechanism. The phrasing of the original paper only talked about matrices of parameters, and most of the other content didn't help on that front (even the specific Stack Overflow question on it). In the end, I read the great illustrated transformer blogpost, which was incredibly clear, and told me that the inputs are just multiplied by the matrices of learned parameters to give the matrices used in the self-attention mechanism. Pretty obvious afterwards, but I was properly confused for a couple of days.

In the course of this first week, I also ended up changing my mind on the usefulness of some resources. For example, I found no use whatsoever for the annotated transformer, even if I assume that people who think in TensorFlow would find that helpful. On the other hand, the aforementionned illustrated transformer that cleared up so many parts of the attention mechanism for me was pretty low in my original list, as it just looked like any other blog post.

So here would be my recommendation for studying transformers, if you are like me a theoretically minded person (albeit with some programming chops):

- Start with the original paper.
- Once you have tried to get the self-attention mechanism, move on to the Illustrated Transformer to get an actual explanation of it.
- To round things up, read this great blog post (nostalgebraist) by nostalgebraist on the history of attention and the history of different types of neural nets.
- (If you want to go a bit further, this survey on the transformer family and extensions contains some good discussions of the limitations of the original architecture and how to deal with them).

As part of my learning, I also had quite a lot of discussions trying to explain the technical details to a friend who knew about transformers but had forgotten the nitty-gritty; and I spent an hour and a half retracing the history of neural nets architectures (based on nostalgebraist post) and a bit of self-attention to my girlfriend, which has a background in chemistry and neuroscience. That was definitely helpful, and left me with the impression that I could explain the ideas decently well, or look for the shaky parts of my explanation pretty quickly.

Week 2: the Loss of Hope

Honestly, the first week went better than I expected, so I was pretty hopeful coming into the second week. Then, I started reading the GPT papers.

Now, the first GPT paper was fine. I didn't learn that much about the architecture (given that it's not that different from the original transformer architecture), but I found out about some fun new NLP related ideas: perplexity and BPEs. So in itself, the paper was pretty interesting.

But GPT-2 ... well my take on the GPT-2 paper is that it's an attempt to make a respectable-sized paper (24 pages!) out of the content "we removed fine-tuning and made the model bigger, and it's now better!". Very important here: I'm not saying that the results are not important, interesting and impressive. But the paper in itself has not that much to say, I feel.

The mistake I did was to force myself to finish it. Which means that I stopped studying, because it was a real bore. I thus lost almost all momentum in the second week. Only when I started reading the GPT-3 paper in the week-end did I get a bit excited. This last paper actually tried to present a conceptual framework to think about the results (zero-shot vs few-shots), which I didn't know in details, and so was pretty exciting. I actually never finished it, but it was enough to push me back into rails for the third week.

Although I didn't really study much this week, I still can give some tentative recommendations:

- Read the GPT paper to get the basic changes to the architecture and a bit of the NLP setting
- Skim the GPT-2 paper really quickly
- Read the GPT-3 paper, with a focus on the framing of zero vs few-shots tasks.
- Probably read on the history of the GPT models and others like BERT (no pointer here, I ended up reading nothing on this).

Week 3 and 4: Fascinating GPT-3

My original plan for toying with GPT-3 was to use AI Dungeon, a storytelling service that uses a fine-tuned version of GPT-3 under the hood (with the premium membership, but there's a free one week-trial). But I unexpectedly found a way to have access a bit to the actual GPT-3 API.

Followed a flurry of experiments and prompt writing, right? Not exactly. After toying a bit with it, I quickly realized that I didn't really knew what I wanted to experiment on. The lack of understanding of why it sometimes worked and other times it didn't also quite frustrated me, coming from a more programming languages perspective.

So I spent a couple of days after that looking at interesting discussions online in the EleutherAI discord, where people have a lot of hands-on knowledge about LMs and GPT-3. This lead to me discovering this great blog. What was so great about it is that it supplied both a detailed explanation of the kind of strategies that work when using GPT-3 (in this post), and a mental model of how to think about Language Models that gave me new alignment ideas (presented in this post, but I also like the intro of the previous post for a quick overview).

I thus ended thinking and discussing far more about GPT-3 than one hour a day, but with a quite unstructured approach. I tried some of the experiments in the methods of prompt-programming blogpost, I wrote rants about how the model of LMs seems to imply interesting directions for alignment, and I discussed with the authors of the blog.

Another surprise was that I didn't spend that much time on Gwern's post about GPT-3. I expected this to be one of the most fun part of the deep dive, but it didn't go that way. But here, contrary to what happened with the GPT-2 paper, I think it's mostly on me. I had already read a lot of the conceptual framing sections, and I'm not that excited by a long list of results, even if they are all individually pretty cool. I'd still want to go through it eventually, and still thinks it's a valuable resource for someone wanting to get a grasp of what GPT-3 can do.

Here are my recommendation for studying GPT-3 more specifically, especially if you're not that hands on:

- Finding a way to toy with the model is probably a great idea. If you can't get direct access (which is pretty hard), alternatives like AI Dungeon are probably good, although they tend to be fine-tuned for specific tasks.
- Read Methods of Prompt-programming and try to apply these methods to see what it gives you.
- If you want a broad view of what GPT-3 can do, Gwern's post is the way to go.
- If instead you want to go deeper in the tricks of prompt programming, I recommend Parsing by counterfactual and List sorting does not play well with few-shot
- If you want to think about how the model works for alignment purposes, I recommend Language models are multiverse generators.

(Note that here even more than in the previous sections, the recommendations are super biased to what I found exciting. There's probably hundreds of great resources on GPT-3 that I don't know).

What I Would Have Done Differently

I'm quite happy with the transformer week; pretty unhappy with the second week; and excited about the rest, while knowing that it was a lot more due to luck than planning. Mostly, there are three things I would change:

- **Try to ankify some of the things I learned.** I half planned to do that, which resulted in me never implementing it. That would be mostly useful for the transformer stuff, I think.
- **Prepare backup plans in case some part of the planning is too hard and/or too boring.** I definitely feel like reading a lot more on the history of GPT models and the controversies (as well as people's reactions) would have been a better use of my second week, if I didn't fixated on reading the GPT-2 paper completely.
- **When possible, force myself to do more hands on experiments.** Even if my investigation of GPT-3 is satisfying, I still feel like I should have tried more prompts, more variants, more experiments. That's not my natural tendancy, but there's some knowledge that can only be gotten like that.

Conclusion

For a first try, I think I did decently well. I managed to focus on one main topic for a month, and learned a lot around it that is already informing some of my alignment research and perspectives on AI and AI risk.

Alex Turner's Research, Comprehensive Information Gathering

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

Introduction

This is the second post in a sequence where I try to focus on one topic for some time (the first two were for a month, but I'm changing with the one I'm currently doing). Initially I called this deep dives, but John's Comprehensive Information Gathering seems more fitting. My goal is not to master the topic; instead I want to learn enough to be able to have a constructive conversation on it, and to follow future work on it.

For the month of May, I focused on Alex Turner's research. Namely Power-seeking and Attainable Utility. The way I went about it was a bit different from my dive into Transformers, because now I had access to the main author. I thus basically read stuff and tried to understand it (while asking question to Alex via discord), and then had calls with him to check that I got most of it right and correct my mistakes.

Concretely, I studied the last version of the Power-seeking paper (which Alex was rewriting at the time) and Reframing Impact. I didn't read the proof (except in one or two cases), but I tried to understand in a lot of details the theorems and lemmas themselves.

The power-seeking I didn't know I needed.

My first surprise came from the power-seeking work, and just how interesting it was. Reading the AF had biased me towards thinking that Alex's main work was on impact measures and Attainable Utility, but I actually find power-seeking more exciting.

Intuitively, the power of a state captures the expected optimal value of this state for a given distribution of reward functions (with some subtleties to make it cleaner) -- how "many" reward functions have an optimal policy passing by this state. An action is then power-seeking compared to another one if it leads in expectation to more powerful states.

The trick that makes this great is Alex's insight that how "many" reward functions have an optimal policy passing by this state boils down to questions of symmetry of the MDP. Especially in the stronger version that only cares about the final cycles of optimal policies (instead of the full trajectory), there is an obvious sense in which a state s with more power than a state s' leads to more such final cycles, which means there is an injection from the final cycles from s' to the final cycles from s (with some final cycles of s left out of the image). This in turns leads to one of the strongest result of Alex's paper: for any "well-behaved" distribution on reward functions, if the environment has the sort of symmetry I mentioned, then for at least half of the permutations of this distribution, at least half of the probability mass will be on reward functions for which the optimal policy is power-seeking.

I've been obsessed with deconfusion lately, and Alex's formalization of power-seeking is a great example of good deconfusion:

- It reduces the confusion to a simple definition with an intuitive meaning
- It only uses very basic mathematical notions to build this definition and its implications: MDPs, state-visit distribution, permutations, orbits...
- It unlock the solution to the initial application of understanding whether competence necessarily involves power-seeking.
- It leads to very interesting conjectures. For example, that as the difference in options becomes larger, maybe the proportion of the orbit that has a majority of power-seeking reward functions grows too.

For those who don't want to read the full paper, I recommend this new post hot out of the oven which gives the intuitions and even some spicy new results.

As for whether I consider this part of the information gathering a success, I want to say yes. I'm able to reasonably explain the core of the result and why it is interesting, and I can follow new development, as was shown to me when Alex sent me his new result about the simplicity prior and power-seeking and I could follow it enough to give some feedback.

I'm also personally excited about trying to link power-seeking with my own deconfusion of goal-directedness, and ask questions about the power-seeking tendencies of competent goal-directed policies instead of optimal policies.

Reframing reframing impact's impact

I ended up being less excited by rereading Reframing Impact. Part of the problem was the rereading, obviously. Also because the sequence is really well-written and paced, I think I had gotten most of the insight from my first readthrough. But there were some new discoveries.

Notably, I hadn't read this post on how to pick a level of impact that is large enough to actually do something, but low enough to keep the benefit of impact measures and AUP.

I still think Attainable Utility Preservation is valuable and a great deconfusion work too, but it is less relevant to my own research interest, and I didn't feel like I got that much out of rereading it.

Conclusion

I'm pretty satisfied with this comprehensive information gathering. I feel like I could have done better, maybe by searching for other angles on AUP instead of just rereading Reframing Impact. That being said, I got what I wanted out of the exercise, and I am quite comfortable now with reading almost anything Alex publishes.