Mysteries of Deep Learning

Zach Furman

Recap

- We asked “what if AGI is a learning machine?” How can we start to reason about such systems, with an eye towards eventually making them safe?
- We covered the problems of approximation, generalization and optimization, and how learning machines can navigate these challenges
- Solomonoff induction provides an optimal answer to approximation/generalization, but leaves optimization unsolved

Motivation

- **What if AGI is based on *deep learning* specifically?**
- These systems seem to be remarkably capable learning machines; from a safety perspective we could potentially bypass hard problems like having to specify human values exactly, etc
- On the other hand we have little understanding of these systems and often are forced to treat them as black boxes - state-of-the-art safety methods are often based more on vibes than rigorous understanding
- Can we build a rigorous understanding?

Overview

- There is not much consensus as to how deep learning is even able to perform so well; in fact existing theoretical frameworks (e.g. bias-variance tradeoff) often actively *counterpredicted* the success of deep learning
    - People do not agree on explanations for how deep learning solves the approximation/generalization/optimization problems
- Further, there are additional empirical mysteries specific to deep learning in particular, such as the emergence of interpretable “mechanistic” structures or convergence of representations across architectures and learning setups, which there are not established consensus explanations for
- Nevertheless there are emerging explanations here which we think are plausible
- We expect solving these mysteries to directly lead to safety applications

Today

- We will read and discuss papers investigating and proposing explanations for these mysteries
- We will explore explanations for how deep learning solves the approximation, generalization, and optimization problems
    - Does deep learning solve approximation/generalization similarly to Solomonoff induction? What does deep learning appear to be doing to solve the optimization problem?
- We will discuss the empirical phenomena of representational alignment and in-context learning. (Mech interp already covered in previous days, but worth keeping in mind)

Learning goals

- This content is designed to support and provide broader context for the other days in the deep learning theory module
- There's a tradeoff between breadth and depth in what we cover: these first two learning theory days are something like a broad but shallow overview, whereas the next three days will be more in-depth "case studies" on specific research directions
- Trying to avoid the failure mode where we just focus on particular hobby horses of existing researchers in the safety community - we want you to be aware of at least the basic phenomena the field is trying to explain
- Explicitly, the goal here is "awareness," not deep knowledge, unlike the "case study" days later in the sequence. What we cover will connect directly to later days, which explore particular solutions in depth

Approximation

- The approximation problem asks: why does the hypothesis class of a learning machine even necessarily *contain* any hypothesis which gets low loss?
- In the context of deep learning: why does there even exist any parameter in a neural network's parameter space which gets low loss?
- The universal approximation razor
  - Approximating generic (K-Lipschitz) functions *requires* (provable lower bound) exponentially many parameters. Therefore, since neural networks do not need ridiculously many parameters to solve realistically large problems, they must be leveraging some non-generic structure of target functions

Generalization

- The generalization problem asks: how can you ensure that you *find* good hypotheses (which generalize to unseen observations) given only *finite data* from observations?
- In deep learning, this becomes: why does training select parameters that perform well on unseen data (get low test loss), given only performance on seen data (the training loss)?
  - It's obvious why SGD would select points with low training loss, so the question refines to: why does training select parameters with a small gap between test and train loss?
- (Again, this is in-distribution generalization, distinct from out-of-distribution generalization)

Optimization

- The optimization problem asks: “how (and when) can my learning method find a solution *efficiently*, assuming a good one exists in my parameter space and I just need to find it?”
- In deep learning, we use variants of SGD for optimization, and this reduces to: how (and when) does SGD find solutions quickly, rather than getting stuck or taking a prohibitively large number of steps?
- Historically, (pre deep learning) machine learning tried to ensure that the loss function was convex in order to avoid this problem, and so this is sometimes phrased as: “why does SGD not get stuck in local minima even though the loss is non-convex?”

Representational alignment

- Deep learning algorithms develop latent *representations* of data, either as an explicit goal or in the process of solving their task
  - Explicit goal: see autoencoders, embedding models
  - Implicitly: think about the kinds of structures found by mechanistic interpretability, like edge-detecting neurons in an image classifier
- Naively, you would expect these representations to be a complex function of many different variables, including the architecture, optimizer, random SGD noise, etc. Surprisingly, these representations often converge across different learning setups, in fact even sometimes between ANNs and humans/animals!

In-context learning

- When GPT-2 was first released, researchers noticed that it had the ability to solve new problems it had never been trained on, given just a few examples at runtime (“few-shot in-context learning”) or even just a description of the problem with no examples (“zero-shot in-context learning”)
- This has since expanded to become the primary mechanism for how LLMs function in practice - they are rarely trained on the exact task users ask them to complete
- Aside: this was a massive break from almost all previous machine learning, which presumed some amount of task-specific training. The terminology we still use reflects this: “pretraining,” “foundation models,” “fine-tuning”

Questions?