Meta-reasoning CSC2547 Presentation

Supervising Strong Learners by Amplifying Weak Experts

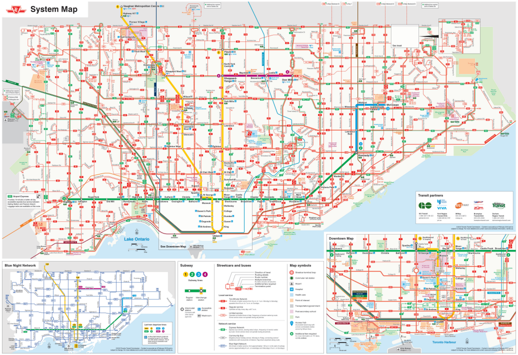
Michal Malyska
Shawn S. Unger

University of Toronto

November 27, 2019

Complex Problems

- We need some kind of training signal for our ML model
- What happens if our problem is too complex for us to have either labeled data or a proxy for a reward?
- What if we are not able to even easily evaluate the answer given by the model?



Example of Complex Task Decomposition

Comparing two designs of a Transit System

- Could train an AI to emulate human judgements but those are often quite bad
- Can try to collect information about the transit systems but this will have a ten year delay.
- It is easy for humans to define sub-tasks that are informative (not necessarily efficient) for the main task:
  - Compare the cost of the two designs
  - Compare the usefulness of the designs
  - Compare the potential risks associated with the designs

Decomposing the Decomposition

- Compare the cost of the two designs:
  - Estimate the likely construction cost:
    - Identify comparable projects and estimate their costs.
    - Figure out how this project differs and how it's cost is likely to differ.
  - Compare the maintenance costs over time
    - Identify categories of maintenance cost and estimate each of them separately.
    - Compare maintenance for similar projects.
    - ...
- Compare the usefulness of the designs:
  - ...
    - ...

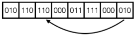
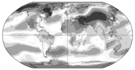
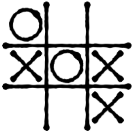
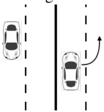
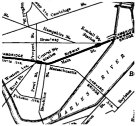
Supervising Strong Learners by Amplifying Weak Experts
Paul Christiano, Buck Shlegeris, Dario Amodei

Paper Overview:

- The Goal is to provide an algorithm to train on tasks for which signals we do not know how to evaluate
- Propose a framework in which they decompose tasks into simpler tasks for which we have a human or algorithmic training signal, in order to build up a training signal to solve the original more complex task
  - Kinda like Karate Kid, you might be better of being taught how to do a few moves which are simple on their own, and then you can learn how to put them all together and kick some butt.

Basic Problem

Table 1: Example problems which require different kinds of training signal.

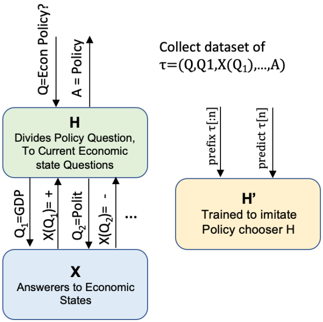
Training signal:	Algorithmic	Human	Beyond human
Supervised learning	Learning data structures 	Image classification 	Long-term prediction 
Reinforcement learning	Playing games 	Driving “well” 	Designing transit system 

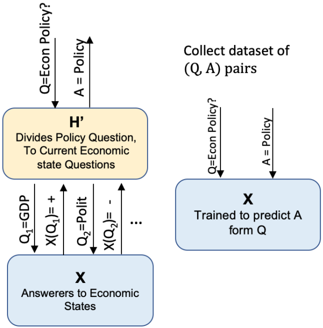
Goal

1. Allow for tasks that can be solved using Supervised and Reinforcement Learning to be greater then current limitations allows
2. Avoid using proxy rewards which can lead to pathological limitations to solve problems
   - Short term behaviour as Proxy for long term effects
   - Related rewards that are calculable as proxy for actual goal of task

Example

Example Implementation for Economic Policy





Thinking about the Problem

1. The Context
   - Usually complex questions come from complex contexts
   - However, if we split down question to subset questions with their our contexts, might be able to more easily solve those questions referring only to the small contexts that they correspond to
2. Solving Problems
   - Solving problems within context sometimes just means understanding it
   - Hence we can change the problem solver to a two step approach

Proposed Approach

- "Our goal is for X to learn the goal at the same time that it learns to behave competently. This is in contrast with the alternative approach of specifying a reward function and then training a capable agent to maximize that reward function."

Algorithm

Training H'

1. Sample $Q \sim D$
2. Run $Amplify^H(X)$ by doing the following for $i \in \{1, \ldots, n\}$
    1. H gets Q_i from Q
    2. $A_i = X(Q_i)$

    then $A = H(A_1, \ldots, A_k)$ to get $\tau = (Q, Q_1, \ldots, Q_k, A_1, \ldots, A_k, A)$
3. Train H' to imitate H

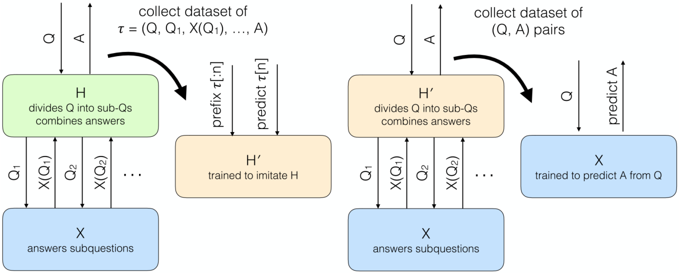


Figure 1: Schematic of our Iterated Amplification implementation.

Algorithm

Training X

1. Sample $Q \sim D$
2. Run $Amplify^{H'}(X)$ by doing the following for $i \in \{1, \ldots, n\}$
    1. H gets Q_i from Q
    2. $A_i = X(Q_i)$

   then $A = H'(A_1, \ldots, A_k)$ to get $\tau = (Q, Q_1, \ldots, Q_k, A_1, \ldots, A_k, A)$
3. Let H' define $A = H'(A_1, \ldots, A_k)$ and collect (Q, A)
4. Train X on (Q, A)

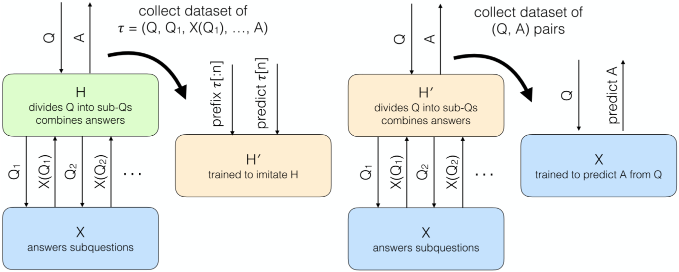
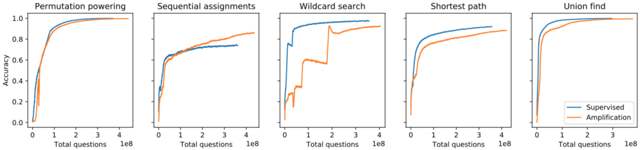


Figure 1: Schematic of our Iterated Amplification implementation.

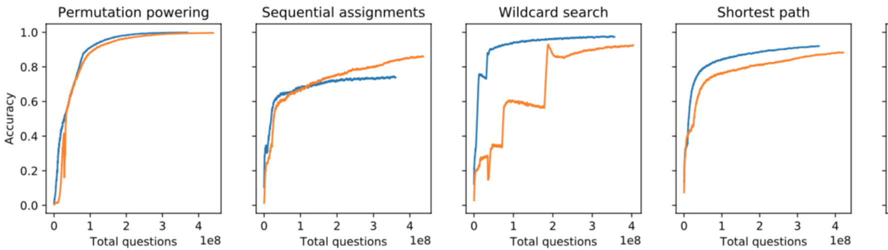
Experiment Results

Present Approaches made in the paper

1. Given a permutation $\sigma : \{1, ..., 64\} \rightarrow \{1, ..., 64\}$, compute $\sigma^k(x)$ for k up to 64.
2. Given $f : \{1, ..., 8\}^2 \rightarrow \{1, ..., 8\}$ and a sequence of 64 assignments of the form $x := 3$ or $x := f(y, z)$, evaluate a particular variable.
3. Given a function $f : \{0, 1\}^6 \rightarrow \{-1, 0, 1\}$, answer questions of the form "What is the sum of $f(x)$ over all x matching the wildcard expression $0 * *1 * *?$"
4. Given a directed graph with 64 vertices and 128 edges, find the distance from node s to t.
5. Given a rooted forest on 64 vertices, find the root of the tree containing a vertex x.



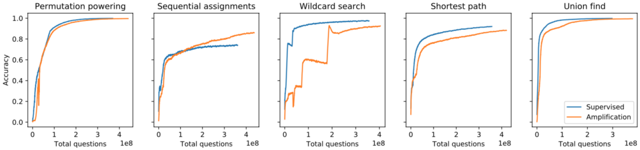
Experiment Results



Experiment Results

- Iterated Amplification is able to solve these tasks effectively with at worst a modest slowdown, achieving our main goal
- Some Differences in Requirement

Amplification	Supervised Learner
- **Tens of thousands** of of training examples - "Modestly" more training steps - Twice as much computation per question	- **Tens of millions** of of training examples



Experiment Architecture

The entire idea behind the architecture is to

- Create and embeding of the various facts and questions asked
- Use a encoder-decoder architecture with self-attention to solve the simplified questions
- Use human-predictor H as also a decoder + the ability to copy solutions from previous levels of the network.

What they got right

- Huge step forward in a relatively new field. Very good introduction to the problem.
- Establishes a framework for solving "beyond human scale" complex tasks.
- Introduces the algorithm starting with design choices that then guide implementation.
- Framework for involving a human in the training process of an algorithm.

Limitations

Theory and Experiments:

- Introduces a very general framework for solving complex problems but only implements a simplified version of it.
- Code not available anywhere with description not detailed enough to easily reproduce it.
- Only considers X as starting from a blank slate.
- Assumes tasks will have a meaningful decomposition within the Question Distribution.

Related Work

Expert Iteration

- Borrows from Daniel Kahneman's idea of System 1 (Intuition) and System 2 (Deliberate evaluation)
- Use an apprentice network to quickly determine plausible actions and use the expert system to further refine guesses
- A refinement of the idea of imitation learning
- *Amplify*^{H} is a very similar idea - expert guides plausible expansions and the learner tries to aid the expert in answering them. The major difference is lack of outside reward function.

Related Work

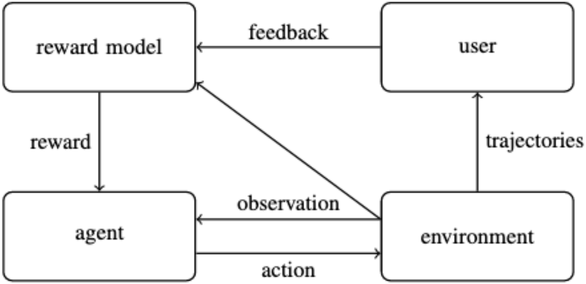
Scalable agent alignment via reward modeling:a research direction

- Attempts to solve the agent alignment problem: How do we make sure that the model we are training is behaves in accordance with our intentions ?
- Discusses key challenges we expect with scaling models to complex domains
- The approach is more or less Iterated Amplification with Reward Modelling instead of supervised learning for the model X

Scalable agent alignment via reward modeling:a research direction

Reward Modelling:

- Separates learning the reward function from user feedback (1) and actually maximizing it (2)
- (1) is called the "What", (2) is called the "How"



Scalable agent alignment via reward modeling:a research direction

The conditions we require our approach to fulfill:

- **Scalability** - Alignment becomes much more important as agents reach superhuman performance and any solution that fails to scale together with our agents can only serve as a stopgap.
- **Economics** - To defuse incentives for the creation of unaligned agents, training aligned agents should not face drawbacks in cost and performance compared to other approaches totraining agents.
- **Pragmatic** - Not supposed to be a solution to all safety problems. Instead, aimed at a minimal viable product that suffices to achieve agent alignment in practice.

Scalable agent alignment via reward modeling:a research direction

Given the two main assumptions:

- We can learn user intentions to a sufficiently high accuracy. In other words, with enough model capacity and training data and algorithms we can extract the intentions.
- For many tasks we want to solve, evaluation of outcomes is easier than producing the correct behavior. E.g. It is a lot easier to yell at a TV screen than to run a basketball team.

Other related ideas and differences

- **Inverse reinforcement learning** - We don't intend to just imitate human choices. This makes it possible to solve more challenging problems.
- **Algorithmic Learning** - We don't have access to ground truth labels.
- **Recursive model architectures** - The learned model doesn't have a recursive structure. The only recursion is generated during training.
- **Debating** - Each sub-question is answered by an independent copy of X trained by $Amplify^H(X)$