Jürgen Schmidhuber - One Big Net For Everything (2018)

Tom Rochette <tom.rochette@coreteks.org>

November 2, 2024 — 36c8eb68

0.1 Context

0.2 Learned in this study

0.3 Things to explore

- Given a single neural network architecture, it might be possible to store different weight values that would represent different views of the environment (this would be akin to a person's consciousness?)
- Is there any research/experiments on the impact of freezing parts of a network if the weights have little variance when training on a new task?
  - This article seems to imply (through 3.12) that weights with low variance between tasks can indicate that they are well tuned for the tasks they accomplish (thus solving some sort of sub-problem within the set of tasks)

1 Overview

2 Notes

2.1 1 Introduction

- To become a general problem solver that is able to run arbitrary problem-solving programs, the controller of a robot or artificial agent must be a general-purpose computer
- Without a teacher, reward-maximizing programs of an RNN must be learned through repeated trial and error, e.g., through artificial evolution or reinforcement learning through policy gradients
- The search space can often be reduced dramatically by evolving compact encodings of RNNs

2.2 2 One Big RNN For Everything: Basic Ideas and Related Work

- ONE or copies thereof or parts thereof are trained in various ways, in particular
  - by black box optimization/reinforcement learning/artificial evolution without a teacher
  - gradient descent-based supervised or unsupervised learning

2.3 3 More Formally: ONE and its Self-Acquired Data

- Cumulative reward $CR(t) = \sum_{\tau=1}^{t} R(\tau)$

2.3.1 3.1 Training a Copy of ONE on New Control Tasks Without a Teacher

- One of ONE's goals is to maximize $CR(t_\tau)$
- Copies of successive instances of ONE are trained in a series of trials through a black box optimization method

  - Incremental neuroevolution
  - Hierarchical neuroevolution
  - Hierarchical policy gradient algorithms
  - Asymptotically optimal ways of algorithm transfer learning
- Given a new task and a ONE trained on several previous tasks, such hierarchical/incremental methods may create a copy of the current ONE, freeze its current weights, then enlarge the copy of ONE by adding a few new units and connections which are trained until the new task is satisfactorily solved
  - This process can reduce the size of the search space for new task, while giving the new weights the opportunity to learn to somehow use certain frozen parts of ONE's copy as subroutines

2.3.2 3.2 Unsupervised ONE Learning to Predict/Compress Observations

- ONE may further profit from unsupervised learning that compressed the observed data into a compact representation that may make subsequent learning of externally posed tasks easier
- Another goal of ONE can be to compress ONE's entire growing interaction history of all failed and successful trials

2.3.3 3.6 Store Behavioral Traces

- To be able to retrain ONE on all observations ever made, we should store ONE's entire, growing, lifelong sensory-motor interaction history including all inputs and goals and actions and reward signals observed during all successful and failed trials, including what initially looks like noise but later may turn out to be regular
- Even human brains may have enough storage capacity to store 100 years of sensory input at a reasonable resolution

2.3.4 3.8 Learning Goal Input-Dependence Through Compression

- The non-trivial pattern recognition required to recognize commands such as "go to the north-east corner of the maze" will require a substantial subnetwork of ONE and many weights. We cannot expect neuroevolution to learn such speech recognition within reasonable time. However, a copy of ONE may rather easily learn by neuroevolution to always go to the north-east corner of the maze, ignoring speech inputs. In the consolidation phase, ONE then may rather easily learn the speech command-dependence of this behavior through gradient-based learning, without having to interact with the environment again

2.3.5 3.12 Heuristics: Gaining Efficiency by Tracking Weight Variance

- As a heuristic, we may track the variance of each weight's value at the ends of all trials. Frequently used weights with low variance can be suspected to be important for many tasks, and may get small or zero learning rates, thus making them even more stable, such that the system does not easily forget them during the learning of new tasks
- Weights with high variance, however, may get high learning rates, and thus participate easily in the learning of new skills

2.3.6 3.13 Gaining Efficiency by Tracking Which Weight Are Used for Which Tasks

- To avoid forgetting previous skills, instead of replaying all previous traces of still relevant trials, one can also implement ONE as a self-modularizing, computation cost-minimizing, winner-take-all RNN. Then we can keep track of which weights of ONE are used for which tasks
- To test whether ONE has forgotten something in the wake of recent modifications of some of its weights, only input-output traces in the union of affected tasks have to be re-tested

2.3.7 3.14 Ordering Tasks Automatically

- The PowerPlay framework offers a general solution to the automatic task ordering problem

    - Given is a set of tasks, which may actually be the set of all tasks with computable task descriptions, or a more limited set of tasks, some of them possibly given by a user
        * In unsupervised mode, one PowerPlay variant systematically searches the space of possible pairs of new tasks and modifications of the current problem solver, until it finds a more powerful problem solver that solves all previously learned tasks plus the new one, while the unmodified predecessor does not. The greedy search of typical PowerPlay variants uses time-optimal program search to order candidate pairs of tasks and solver modifications by their conditional computational (time and space) complexity, given the stored experience so far

3 See also

4 References

- Schmidhuber, Juergen. "One Big Net For Everything." arXiv preprint arXiv:1802.08864 (2018).