The Big World Hypothesis and its Ramifications for Artificial Intelligence

Khurram Javed
kjaved@ualberta.ca

Richard S. Sutton
rsutton@ualberta.ca

Alberta Machine Intelligence Institute (Amii)
Department of Computing Science, University of Alberta
Edmonton, Canada

Abstract

The big world hypothesis says that for many learning problems, the world is multiple orders of magnitude larger than the agent. The agent neither fully perceives the state of the world nor can it learn the correct value or optimal action for each state. It has to rely on approximate solutions to achieve its goals. In this paper, we make a case for embracing the big world hypothesis. We argue that even as computational resources grow, the big world hypothesis remains relevant. We conclude by discussing the implications of accepting the big world hypothesis on the design and evaluation of algorithms.

1 The Big World Hypothesis

The *big world hypothesis* says that in many decision-making problems the agent is orders of magnitude smaller than the environment. It can neither fully perceive the state of the world nor can it represent the value or optimal action for every state. Instead, it must learn to make sound decisions using its limited understanding of the environment. The key research challenge for achieving goals in big worlds is to come up with solution methods that efficiently utilize the limited resources of the agent.

The opposing view to the big world hypothesis is that real-world decision-making problems have a simple solution. The agent is not only capable of representing the simple solution but also has additional capacity that can be used to search for the solution more efficiently—they are *over-parameterized*. The key research challenge for achieving goals with over-parameterized agents is to find the solution that enables optimal decision-making in perpetuity.

There are many problems that satisfy the big world hypothesis and many that do not. The problem of finding roots of a second degree polynomial admits a simple solution that always work. Representing the value function of the game of Go for all states does not have a simple solution. The big world hypothesis is more a statement about the class of problems we should care about than a fact about all decision-making problems. It can be made true or false by exercising control over the design of the environment and the agent (*e.g.*, when developing benchmarks).

Developing algorithms for big worlds poses unique challenges. The best algorithms for big worlds might prefer fast approximate solutions over slow exact ones. They might learn incorrect simplistic models that are sufficient for achieving agent's goals over causally correct complex models (*e.g.*, Newtonian physics as opposed to quantum mechanics). They might forgo knowledge that is not frequently used by the agent to make room for knowledge used more often. Such trade-offs do not exist for over-parameterized agents.

The big world hypothesis is not a novel proposition. Over the past few years, several independent works have entertained the idea of small bounded agents learning in large unbounded environments.

Sutton (2020) argued that the world is large and complex and an agent cannot learn everything there is to learn exactly. He proposed embracing function approximation for learning values, policies, models, and states. Dong et.al. (2022) theoretically studied the performance of a reinforcement learning algorithm without making simplifying assumptions about the environment. Their work shifts the focus from making assumptions about the environment to making assumptions about the capabilities of the agent. Javed et.al. (2023) empirically studied the performance of small agents in large environments. They found that approximate algorithms that use less computation can outperform exact algorithms that use more computation in big worlds. Kumar et.al. (2023) showed that continual learning is a necessary element of reinforcement learning when the agent is computationally constrained.

Is the big world hypothesis a temporary artifact of limitations of our current computers? Or would it have relevance even as computational resources grow? In the next section, we argue that the big world hypothesis is here to irrespective of the rate at which computational resources grow.

2 Reconciling the Big World Hypothesis and Exponentially Growing Computation

Historically, access to computation has increased exponentially. With continuing growth, computers of the future could be sufficiently powerful to solve all problems we care about using over-parameterized agents. We see two problems with this view.

First, it is not just our agents that are constrained by compute. The sensors used by our agents are also constrained by compute. A rise in computation makes it possible to sense the world with more precision and at a higher frequency. For example, within the last decade the camera sensors in our phones have gone from sensing 640 x 420 pixels at 30 fps—around 7 million pixels per second—to sensing in 4k at 60 fps—around 500 million pixels per second. To put these numbers in perspective, a modern smartphone camera sensor in 2024 can generate more data in a week than that used to train GPT-3 (Brown et.al., 2020). Even with these massive increases in the ability to sense the world, our agents are not even close to sensing the world at its full scale. We speculate that as computational resources grow so would the appetite to sense the world at higher fidelity, making the decision-making problem more challenging.

The second problem with waiting for compute to grow is that as compute becomes more readily available, the world itself becomes more complex. From the perspective of an agent, the world consists of everything outside of itself. This includes other equally complex agents and computers. An agent that interacts with multiple other agents of similar capabilities would be unable to model the world exactly regardless of the rate at which computation grows.

A concrete example of the world getting more complex as computation grows is that of an agent playing the game of Go against an opponent. If the opponent picks moves randomly, it is fairly simple for the agent to model the environment exactly. The dynamics of the environment can be simulated with a short program. However, if the opponent is more complex, such as an AlphaZero (Silver et.al., 2015) agent, the only way to model the dynamics of the environment correctly is to be able to represent the policy of the large AlphaZero agent accurately.

As computational resources increase so does the complexity of the world. The big world hypothesis is not a temporary artifact of limitations of our current computers. For many problems, the world will always be much larger than any single agent.

3 Existing Evidence Consistent with the Big World Hypothesis

There is some indirect evidence that shows that the behavior of our learning algorithms on large problems is consistent with the big world hypothesis. We discuss two cases.

Silver et al. (2015) trained a large neural network to learn the value function for the game of Go. They found that even after extensive training the performance of the system could be improved if the decisions were taken by combining the value function with a planner.

If the neural network had the capacity to represent the optimal value function of Go, and it had been trained for sufficiently long time, decision time planning should not have improved performance. Perhaps the neural network did not have sufficient capacity to represent the value function correctly for all states and the planner was able to fill in the gaps.

The second and more direct evidence comes from the work of Brown et.al. (2020). They showed a clear trend between the model size and performance of neural networks when fitting large language datasets. They found that the train and validation error on the dataset could be reduced by increasing the number of parameters in the network. Their finding is consistent with the big world hypothesis and makes little sense if the neural networks were over-parameterized.

Neither of the two papers directly set out to test the big world hypothesis and their results have other explanations. However, they don't contradict the big world hypothesis and provide circumstantial evidence for its relevance.

4 Ramifications of the Big World Hypothesis on Algorithm Design

The big world hypothesis is only worth discussing if accepting it would directly impact how we do research in AI. In the next subsections, we discuss three ways accepting the hypothesis can influence research today.

4.1 Online continual learning is an important solution method for achieving goals in big worlds

The need for online continual learning in big worlds is intuitive—if the agent does not have the resources to learn and retain everything important about the world simultaneously, it can learn aspects that are important for decision-making at the current time and discard them when they are no longer useful by learning continually. In the over-parameterized setting, on the other hand, there is no need for online continual learning. Once the agent has found the underlying optimal solution, it can use it forever without changing.

Learning things when they are needed and discarding them when they are not is sometimes called *tracking*. Tracking has been empirically demonstrated to be superior to fixed solution in partially observable environments by Sutton, Koop, & Silver (2007) and Silver, Sutton, & Müller (2008).

A key requirement or tracking to be effective is *temporal coherence.* Temporal coherence means that parts of the world the agent experiences from one step to the next are correlated. An agent learning online can exploit the temporal coherence to direct its resources to learn about the states of the world that are temporally close at the expense of those that are far away. Tracking can be a powerful solution method in temporally coherent big worlds.

Humans extensively rely on tracking in everyday life
Humans are continually learning agents. We extensively rely on tracking to achieve our goals. An intuitive example is that of exams. Given the choice between taking exams of different subjects on different days or taking them all on the same day, most of us would pick the former. Intuitively, it feels easier to have to only have to learn and remember the material for one exam at a time. This is exactly the behavior we should expect from a tracking agent in a big world.

An analogy of a tracking system is the cache used by a CPU. The cache is much smaller than the memory and can only store a small fraction of instructions and data used by the program. However, by retaining the right pieces of information and discarding the least useful ones, a small cache can have a high hit ratio. Moreover, a high hit ratio is only possible when the program accesses memory predictably, akin to having temporal coherence in big worlds.

If we were to accept the hypothesis, we would have to develop algorithms that can learn online and continually. This is a significant departure from the current practice of training agents offline and then deploying them.

4.2 Computationally efficient learning algorithms can be advantageous in big worlds

In big worlds, increasing the size of the agent can improve performance. This raises an important trade-off between the complexity of the learning algorithm and the size of the agent. A trivial example is the mini-batch size of a deep RL algorithm, such as DQN (Mnih et.al. 2015). For a fixed amount of resources, an agent can double the number of parameters by halving the mini-batch size.

Javed, Shah, Sutton, & White (2023) empirically demonstrated that approximate but efficient learning algorithms can outperform computationally expensive exact algorithms in big worlds. In their experiments, they evaluated tiny recurrent networks—hidden state is a vector of less than 10 dimensions—on the Arcade Learning Environment (Bellemare et.al., 2013). They constrained all algorithms to use the same amount of per-step computation. They found that a simple algorithm that used less computation was able to outperform a more complex algorithm by repurposing the saved computation to increase the size of the network.

Accepting the big world hypothesis mean we should actively look for more efficient learning algorithms.

4.3 Making progress on big world problems requires a different approach for evaluating algorithms

A common way to evaluate algorithms is to run them on a standardized benchmark. A good benchmark is an accurate proxy for the real-world problem we care about and allows us to do careful experiments. Designing a benchmark for big worlds requires a different approach than designing a benchmark for over-parameterized agents.

One way to evaluate algorithms for big worlds is to test them on complex environments so that even our largest agents on the latest hardware are not over-parameterized. While this approach has merit, it makes it difficult to do careful and reproducible experiments.

The alternative is to restrict the computational capabilities of the agents instead of making the environments larger. The primary limitation of restricting agents is that we might miss out on emergent properties of large agents. However, a small agent learning in a non-trivial environment is still a better proxy for learning in big worlds than a large over-parameterized agent learning in the same environment.

Example: A typical DQN agent for Atari users orders of magnitude more computation than the enviornment.
Arcade learning environment (Bellemare et.al., 2013) is a popular benchmark for reinforcement learning. A typical game in the benchmark can run at around 7000 frames per second on a modern CPU core. A DQN agent (Mnih et.al., 2014), on the other hand, runs at 300 frames per second on a modern GPU. While it is hard to directly compare different implementations of the agent and the environment running on different hardwares, it is clear that the agent uses orders of magnitude more computation than the environment in this case..

Restricting the computational capabilities of the agents is not trivial. There is no consensus on what aspects of the agents should be restricted. We could restrict the number of operations, the amount of memory, the amount of memory bandwidth, or the amount of energy the agent can use. The choice of constraints can have a significant impact on the performance of the agent.

One option is to match the constraints on the agent with the constraints imposed by current hardware. For example, if memory is cheaper than CPU cycles, we might want to restrict the CPU cycles. Alternatively, if accessing the memory is a bottleneck, we might want to restrict the memory bandwidth.

A second option is to limit energy usage. Energy is a universal constraint that can take into account the evolution of hardware overtime and can even drive research for designing better hardware for our agents. The downside of using energy as a constraint is that it is difficult to measure. Normally, the computer running the agent is also running the environment, an operating system, and other unrelated processes. Isolating the energy used by the agent from background tasks is challenging.

5 Conclusions

The big world hypothesis has direct implications on what we choose to study and how we evaluate our algorithms. It is not a temporary artifact of current limitations of our computers. It is imperative that we develop algorithms that can allow agents to achieve goals in big worlds. This requires developing computationally efficient algorithms for learning continually. It also requires rethinking the way we benchmark our algorithms.

References

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877-1901.

Dong, S., Van Roy, B., & Zhou, Z. (2022). Simple agent, complex environment: Efficient reinforcement learning with agent states. Journal of Machine Learning Research, 23(255), 1-54.

Javed, K., Shah, H., Sutton, R. S., & White, M. (2023). Scalable real-time recurrent learning using columnar-constructive networks. Journal of Machine Learning Research, 24, 1-34.

Kumar, S., Marklund, H., Rao, A., Zhu, Y., Jeon, H. J., Liu, Y., & Van Roy, B. (2023). Continual learning as computationally constrained reinforcement learning. arXiv preprint arXiv:2307.04345.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. nature, 518(7540), 529-533.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., ... & Hassabis, D. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv preprint arXiv:1712.01815.

Silver, D., Sutton, R. S., & Müller, M. (2008, July). Sample-based learning and search with permanent and transient memories. In Proceedings of the 25th international conference on Machine learning (pp. 968-975).

Sutton, R. S. (2020). Rich Sutton, Are You Ready to Fully Embrace Approximation? (June 8, 2020) [Video]. YouTube. https://www.youtube.com/watch?v=JjB58InuTqM

Sutton, R. S., Koop, A., & Silver, D. (2007, June). On the role of tracking in stationary environments. In Proceedings of the 24th international conference on Machine learning (pp. 871-878).