The Need for a Big World Simulator: A Scientific Challenge for Continual Learning

Saurabh Kumar*
szk@stanford.edu
Department of Computer Science
Stanford University
*Denotes equal contribution.

Hong Jun Jeon*
hjjeon@stanford.edu
Department of Computer Science
Stanford University

Alex Lewandowski
lewandowski@ualberta.ca
Department of Computing Science
University of Alberta

Benjamin Van Roy
bvr@stanford.edu
Department of Electrical Engineering
Stanford University

Abstract

The "small agent, big world" frame offers a conceptual view that motivates the need for continual learning. The idea is that a small agent operating in a much bigger world cannot store all information that the world has to offer. To perform well, the agent must be carefully designed to ingest, retain, and eject the right information. To enable the development of performant continual learning agents, a number of synthetic environments have been proposed. However, these benchmarks suffer from limitations, including unnatural distribution shifts and a lack of fidelity to the "small agent, big world" framing. This paper aims to formalize two desiderata for the design of future simulated environments. These two criteria aim to reflect the objectives and complexity of continual learning in practical settings while enabling rapid prototyping of algorithms on a smaller scale.

1 Introduction

The real world is an unfathomably complex system, both to humanity and to the agents that it designs. The small agent, big world frame captures this perspective by positing that to an agent with bounded computational resources, or *capacity*, the world appears complex and non-stationary (Dong et al., 2022). As such, in order for said bounded capacity agent to continually fruitfully engage with the world, it must continuously ingest new knowledge while selectively retaining previously acquired knowledge. We refer to this process as *continual learning*.

Thus far, the study of continual learning has revolved around the discovery of phenomena which plague the naive application of algorithms designed in traditional machine learning settings. Such phenomena include the likes of "catastrophic forgetting" which describes an agent's tendency to forget information from the past (Kirkpatrick et al., 2017; Zenke et al., 2017), and "plasticity loss" which describes an agent's inability to absorb new information with increased time (Dohare et al., 2021; 2023; Kumar et al., 2023b; Lyle et al., 2023). These studies have brought about the design of synthetic environments which produce artificially generated and controllable data streams. These environments are designed to stress test an agent's resilience to these phenomena.

However, if we frame the goal of continual learning to design agents which continue to engage fruitfully with the world, existing benchmarks do not capture the essence of the world. For instance, on current benchmarks, evaluation metrics for catastrophic forgetting measure an agent's ability to remember everything in the past, which is unlikely to be necessary to fulfill a definition of fruitful engagement with the world. Current benchmarks also suffer from a limited variation in data since they draw from datasets designed for non-continual learning settings, like MNIST, CIFAR, and ImageNet. Further, they exhibit unnatural distribution shifts, such as the abrupt change in pixel permutation seen in Permuted MNIST and the random reassignment of labels to images in Random Label CIFAR. Such constraints do not aptly capture the gradual and often subtle evolution of data encountered in real-world settings.

Moreover, these benchmarks do not embody the small agent, big world view. In particular, for natural models of an optimal capacity-constrained agent operating in a vastly more complex world, the agent is always able to substantially improve its performance if granted greater capacity. For example, performance shortfall might shrink by a factor of at least some fixed A each time capacity is scaled by a factor B. In existing continual learning benchmarks, there are diminishing returns to increasing an agent's capacity. In particular, at sufficient scale, each successive factor B increase in capacity reduces performance shortfall by an exponentially smaller rather than fixed factor.

In this paper, we emphasize the need for a "big world simulator." By this we mean a new type of synthetic environment that more accurately reflects the objectives and complexity of continual learning in practical settings. Such an environment should simultaneously retain the property of existing benchmarks that rapid prototyping of algorithms at small scale is possible. Furthermore, we echo Kumar et al. (2023a) by suggesting a concrete objective – that of minimizing average error over an infinite horizon subject to a capacity constraint – which reflects an agent's ability to fruitfully engage with the world. We also derive a novel decomposition of this objective into two terms which closely resemble the notions of "forgetting" and "plasticity loss" from the literature. By highlighting aforementioned environment criteria and a clear objective, we aim to guide future research toward developing algorithms that can be evaluated at small scale while maintaining practical relevance in real-world applications.

We view this workshop as an opportunity to engage with the continual learning community in exploring the development of an environment simulator that can facilitate scientific advances in the area. This paper serves as an important part of this communication, laying out principles and concepts that can help frame the discussion. The paper is organized as follows. In Section 2, we briefly review a common form of synthetic continual learning benchmark that is currently used for evaluating agents. In Section 3, we formalize the notion of an environment and an agent as they pertain to continual learning. In Section 4, we propose a set of criteria for the design of future synthetic environments. In Section 5, we discuss the implications of these desiderata on evaluating forgetting and implasticity. Finally, in Section 6, we propose an illustrative example of a big world simulator that satisfies the two criteria.

2 A Common Recipe for Synthetic Continual Learning Benchmarks

Synthetic benchmarks are useful for rapid prototyping, enabling a researcher to evaluate an agent's capabilities with respect to an array of identified challenges (Osband et al., 2020; 2022). Despite three decades of active research on continual learning (Ring, 1994; Thrun, 1998), there remains comparative difficulty in i) identifying the challenges unique to continual learning and (ii) measuring progress made on previously identified challenges. For example, the challenge of catastrophic forgetting associated with existing continual learning benchmarks has been been addressed by computationally inefficient solutions which are infeasible in most real-world scenarios (Prabhu et al., 2020). In this section, we briefly discuss a commonly used recipe for creating synthetic continual learning benchmarks and its fundamental limitations.

Most synthetic continual learning benchmarks use the following recipe: take an existing non-continual dataset and create a continual learning problem by incorporating some form of non-stationarity. This data stream is often delineated into a sequence of tasks, where the non-stationary is applied at specific points in time to create a new task by transforming the observation distribution or the target function. Within each task, the environment faced by the agent is stationary. For example, in Permuted MNIST (Srivastava et al., 2013; Goodfellow et al., 2013; Kirkpatrick et al., 2017), each task involves a fixed, distinct permutation of the pixels of all images in the MNIST dataset, and the challenge is to classify the digit of each image with these new inputs (See Appendix A).

This recipe for creating a continual learning benchmark has limitations stemming from its dependence on applying non-stationarity to a previously collected dataset. First, the dataset is limited in its complexity to observations and classes that have been largely collected for the purposes of training an agent to convergence. Second, the non-stationarities used are limited to artificial, unnatural and abrupt changes to the experience stream unlike the kind of structured non-stationarity of the world. These limitations of the benchmark can result in an agent that either (i) is able to outscale the challenge of learning continually (Prabhu et al., 2020), or, (ii) can no longer benefit from additional capacity. There are certainly advantages to using the recipe: several datasets already exist, along with baselines and performance levels for comparison. However, addressing these limitations can lead to a continual learning benchmark that provides a controllable, open-ended, and challenging experience stream from which an agent must learn continually.

3 Formalizing the Environment and Agent

Before outlining the criteria for a big world simulator, we first propose information-theoretic characterizations of an environment and an agent. These characterizations will help formalize the notion that the world appears complex and non-stationary to a bounded capacity agent, necessitating a trade-off between ingesting and ejecting information.

3.1 Notation

We begin with a review of notation and probability and information theory that we will use in this paper. We define random variables with respect to a common probability space $(\Omega, \mathbb{F}, \mathbb{P})$. For a random variable $X : \Omega \mapsto \mathcal{X}$, we use $\mathbb{P}(X \in \cdot)$ to denote the distribution of X. We use $\mathbb{H}, \mathbb{I}$ to denote (conditional) entropy and mutual information respectively. Concretely, for random variables X, Y, Z

$$\mathbb{H}(X|Y) = \mathbb{E}\left[\ln \frac{1}{\mathbb{P}(X|Y)}\right]; \quad \mathbb{I}(X;Y|Z) = \mathbb{H}(X|Z) - \mathbb{H}(X|Y,Z).$$

We use the notation $\mathbf{d}_{\mathrm{KL}}$ to denote the KL divergence, which maps two probability distributions to $\Re_+ \cup \{+\infty\}$, where $\Re_+$ denotes the non-negative reals.

3.2 The Environment

Let $(\mathcal{X}, \Sigma_{\mathcal{X}})$ be a measurable space which consists of the set of observations and a sigma-algebra on that set. Let $\mathcal{H} = \{\mathcal{X}^n : n \in \mathbb{N}\}$ denote the set of all *histories* of observations. An environment is identified by a function f which maps $\mathcal{H}$ to the set of probability measures $\Sigma_{\mathcal{X}} \mapsto [0, 1]$.

We assume that the environment generates a stream of observations which is characterized by a stochastic process $(X_0, X_1, X_2, \ldots)$ defined with respect to a probability space $(\Omega, \mathbb{F}, \mathbb{P})$. The conditional distribution $\mathbb{P}(X_{t+1} \in \cdot|H_t) = f(H_t)$, where for all $t \in \mathbb{Z}_+$, $H_t = (X_0, X_1, \ldots, X_t) \in \mathcal{X}^n$ denotes the history of observations up to time t.

3.3 The Agent

In this work, we consider an agent to be a map $\pi : \mathcal{U} \times \mathcal{X} \mapsto \mathcal{U}$ where $\mathcal{U}$ denotes a set of *agent states*. For all t, we let $U_t = \pi(U_{t-1}, X_t)$ denote the agent state at time t, which is computed from the previous agent state U_{t-1} and the latest observation X_t. We represent constraints placed on the agent via an *informational* constraint $\mathbb{I}(H_t; U_t) \le c$ for all t. We refer to this value c as the *capacity* of the agent.

3.4 Predictions and Error

An agent is evaluated for its ability to accurately predict future observations. Concretely, for any agent π, we define the error of π to be

$$\mathcal{L}_\pi \;=\; \liminf_{T\to\infty}\; \mathbb{E}_\pi\left[\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{d}_{\mathrm{KL}}\left(\underbrace{\mathbb{P}(X_{t+1}\in\cdot|H_t)}_{\text{environment probability}}\;\|\;\underbrace{\mathbb{P}(X_{t+1}\in\cdot|U_t)}_{\text{agent prediction}}\right)\right].$$

We characterize an agent's effectiveness via it's error. Since the function f (environment probabilities) may depend arbitrarily on H_t it is clear that an agent with limited capacity will incur nonzero error. For all t, U_t will likely be missing information about H_t which may be relevant for making predictions about X_{t+1}. Therefore, an effective agent ought to selectively remember and forget aspects of H_t which lead to the best predictions about the future, while obeying the capacity constraint.

4 Desiderata for a Big World Simulator

Equipped with formal characterizations of an environment, an agent, and error incurred by an agent interacting with the environment, we now propose two criteria for a big world simulator.

4.1 There are no diminishing returns to increasing an agent's capacity.

In the small agent, big world setting, the world is enormously – perhaps even infinitely – complex relative to any finite-capacity agent. Since there is always more to learn about the world, a desired feature of continual learning agents is that endowing an agent with significantly increased capacity should invariably lead to substantial improvement in its capabilities. We frame this feature as a property of a big world simulator.

To capture the notion that increasing capacity yields significant performance improvements, we establish a constraint on how the prediction error diminishes as the agent's capacity expands. First, let

$$\mathcal{L}(c) \;=\; \inf_{\pi:\forall t,\ \mathbb{I}(H_t;U_t)\le c} \mathcal{L}_\pi.$$

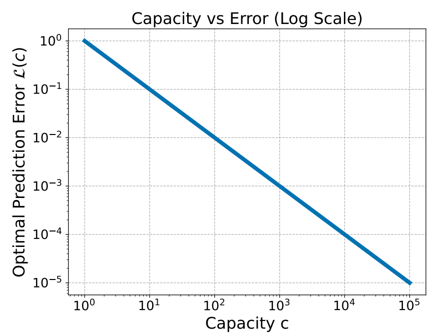

Figure 1: To illustrate the notion that there are no diminishing returns to increasing the agent's capacity, we plot what the capacity versus optimal prediction error would look like for a 1-complex environment. This is simply the curve $\mathcal{L}(c) = \frac{1}{c}$. In order reduce error by a factor of 2, we must put forward *twice* the capacity. When both the x and y-axes are in log scale, this curve appears as a line with slope of value 1.

$\mathcal{L}(c)$ denotes the optimal error incurred by an agent with capacity c. We draw inspiration from the research in large language models which observes the desired phenomenon of continued reduction in out-of-sample error for every extra unit of available compute (Kaplan et al., 2020; Hoffmann et al., 2022). In these works, they establish a *power law* relationship between the error and capacity. This is in contrast to many classical statistical settings in which the reduction in out-of-sample error decays *exponentially* fast in c. We now present the notion of k-complexity:

Definition 4.1. For $k \in \Re_{++}$, an environment is *k-complex* if

$$\mathcal{L}(c) \;=\; \Theta\left(\frac{1}{c^k}\right).$$

This definition characterizes an environment's ability to consistently offer performance improvements in response to increases in agent capacity. An illustration of what the capacity vs error curve would look like for a 1-complex environment is provided in Figure 1.

4.2 An optimal finite capacity agent interacting with the environment will never stop learning.

Since our goal is to evaluate the continual learning capabilities of agents, the environment should be designed such that an effective capacity constrained agent ought to *never stop learning*. To formalize this concept, we begin by framing nonstationarity as a description of the agent's subjective experience. Formally:

Definition 4.2. An environment f is *non-stationary* with respect to an agent π if

$$\limsup_{t \to \infty} \; \mathbb{I}(X_{t+1}; X_{t+2:\infty} | U_t) > 0.$$

This states that an agent π will always continue to encounter new information in X_{t+1} which will be relevant for making predictions about the future $X_{t+2:\infty}$ beyond what is currently known in U_t. Therefore, in our environment design we hope to design f such that it is non-stationary with respect to *all* finite-capacity agents π.

We note that for most environments studied in the literature, f is non-stationary with respect to an *unbounded* agent for which for all $t, U_t = H_t$. Concretely, in these environments,

$$\limsup_{t \to \infty} \; \mathbb{I}(X_{t+1}; X_{t+2:\infty} | H_t) > 0.$$

For instance, in an environment such as Permuted MNIST, there will continue to exist t for which X_{t+1} is data from a *new* permutation. As a result, this X_{t+1} will contain information about the future sequence which is absent from H_t and hence the environment is non-stationary even to an *unbounded* agent.

However, we make the point that even if

$$\limsup_{t \to \infty} \; \mathbb{I}(X_{t+1}; X_{t+2:\infty} | H_t) = 0, \tag{1}$$

f may be non-stationary with respect to π due to capacity constraints. Since the agent is only able to retain finite bits of information about the past, there will always be more to learn about in the future. As a result, the agent must always decide how to selectively *forget* information in U_t to incorporate relevant new information in X_{t+1} We argue that environments which satisfy equation 1 along with Definition 4.2 better capture the essence of "small agent, complex world."

Our notion of non-stationarity relates to the work of Abel et al. (2023), which proposes a definition of continual reinforcement learning. That definition offers a formal expression of what it means for an agent to never stop learning. The characterization is subjective in that it defines non-convergence with respect to a *basis*, which is a fixed set of policies. Definition 4.2 is similarly subjective, defining non-stationarity with respect to a particular agent with a particular agent state.

Given our definition of non-stationarity, we return to the implications on the behavior of an optimal agent. We assume that the agent state U_t is capacity limited in that for all t, $\mathbb{I}(U_t; H_t) \leq c$. The environment ought to be designed such that an optimal capacity limited agent must never stop learning i.e.,

$$\limsup_{t \to \infty} \; \mathbb{I}(X_{t+1}; U_{t+1} | U_t) > 0.$$

An optimal agent will never stop injecting new information into its agent state, since this information will be needed for making future predictions.

5 Forgetting and Implasticity

In this section, we draw the connection between our notion of error ($\mathcal{L}_\pi$) and phenomena described in the literature (notably "forgetting" and "plasticity loss" or "implasticity"). We begin by providing the following decomposition of error into forgetting and implasticity. This decomposition quantitatively improves upon the result of Kumar et al. (2023a) to provide definitions of forgetting and implasticity which better match our intuition. We defer the proof to Appendix B.

Theorem 1. (forgetting and implasticity) For all agents $\pi : \mathcal{U} \times \mathcal{X} \mapsto \mathcal{U}$, if for all t, $U_{t+1} = \pi(U_t, X_t)$, then

$$\mathcal{L}_\pi \;=\; \liminf_{T\to\infty} \; \underbrace{\frac{1}{T}\sum_{t=0}^{T-2} \mathbb{I}(X_{T:t+2}; U_t | U_{t+1}, X_{t+1})}_{\text{forgetting}} + \underbrace{\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{I}\left(X_{T:t+1}; X_t | U_t\right)}_{\text{implasticity}}.$$

The forgetting that an agent experiences at time $t+1$ is the information about the future sequence $X_{T:t+2}$ which was contained in the previous agent state U_t, but not in the current agent state U_{t+1} nor the most recent observation X_{t+1}. Since agent state $U_{t+1} = \pi(U_t, X_{t+1})$, this mutual information exactly captures which *relevant* information was decanted from the agent state incorporate information from X_{t+1}. Note that forgetting is *forward looking* as it measures mutual information between the previous agent state and the *future* sequence as opposed to the *past* sequence as done in the literature. Forgetting about the past is not "catastrophic" insofar as what is forgotten is uninformative about the future.

Meanwhile, the implasticity that an agent experiences at time t is the information about the future sequence $X_{T:t+1}$ which was contained in X_t, which we *failed* to include in U_t. This directly reflects what is referred to in the literature as "loss of plasticity". An agent loses plasticity ,or incurs loss from implasticity, if it fails to digest information about X_t which is relevant for making predictions about the future $X_{T:t+1}$.

Theorem 1 suggests that if our environment is k-complex, then an effective continual learning agent will effectively reduce $\mathcal{L}_\pi$ by improving upon forgetting and/or implasticity with this additional compute. Perhaps with this additional capacity, the agent will be able to hold on to more *relevant* information throughout its experience and incur lower error due to not forgetting. Or perhaps, the agent will elect to extract relevant information out of the data at each time step and suffer less error from implasticity. However, it's evident that a capacity-constrained agent in a k-complex environment will inevitably experience error due to forgetting and implasticity.

We now draw the connection to the criterion that a finite capacity agent should never stop learning. Consider an agent π which *stops* learning i.e. for some T, $U_t = U_T$ for all $t > T$. If the f is non-stationary w.r.t π, then

$$\begin{aligned} 0 &\overset{(a)}{<} \mathbb{I}(X_{t+1}; X_{t+2:\infty} | U_t) \\ &\overset{(b)}{=} \underbrace{\mathbb{I}(X_{t+1}; X_{t+2:\infty} | U_{t+1})}_{\text{implasticity}} \end{aligned}$$

where (a) follows from the definition of non-stationarity and (b) follows from the fact that π stops learning. Therefore, if f appears non-stationary to all agents π, then an agent which *stops learning* will *necessarily* incur error due to implasticity. To mitigate this, an effective capacity-constrained agent ought to never stop learning.

6 An Illustrative Example: Turing-complete Prediction Environment

In this section, we propose a Turing-complete machine as an illustrative example of a big world simulator. A Turing-complete machine can be considered the biggest possible world because it is capable of executing any computable program, world, or environment (Turing, 1936). The specific machine that we use to simulate the big world is the cellular automaton Rule 110, which is the only elementary cellular automaton proven to be Turing-complete (Cook et al., 2004). Figure 4 in Appendix C illustrates how Rule 110 updates each cell.

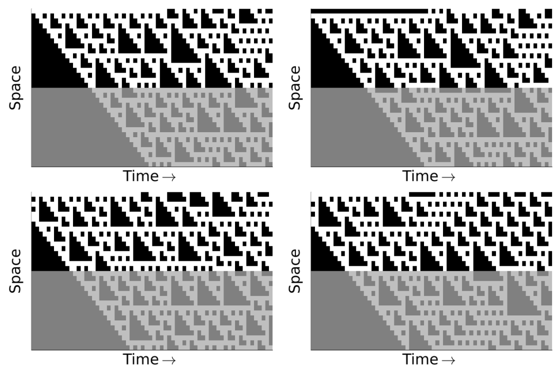

Figure 2: Four different simulations of Rule 110 with periodic boundary conditions, starting from the binary representation of integers 1, 2, 53, and 107 respectively. The gray shaded region is the unobserved region used to simulate the infinite state. At each time step, the agent oberves a vertical slice of the unshaded region.

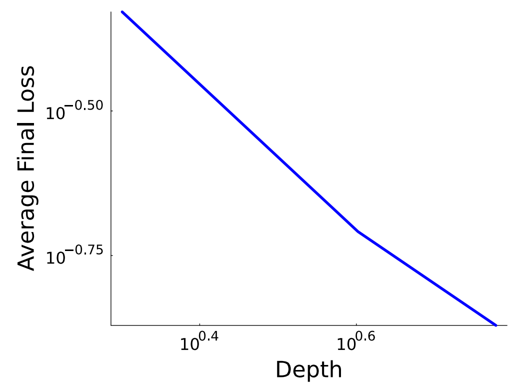

Figure 3: The error as a function of depth approximates k-complexity. Here, the depth of a feed-forward neural network is our measure of capacity.

The two desiderata of a big world are satisfied in theory by the the problem of predicting future observations from Rule 110. This is because Rule 110 defines a transition over an infinite state. An agent with finite capacity tasked with predicting an observable region of the state can only observe a finite portion of the state and can only predict up to some finite horizon. Critically, the observed region at each time step depends on both the observed and unobserved regions from previous time steps. Increasing the agent's capacity can improve its predictions, but the agent must never stop learning because of the infinite unobserved region.[1] See Figure 2 for different example simulations.

We now demonstrate that the Turing-complete prediction environment empirically satisfies the two desiderata of a big world simulator. We trained a neural network with SGD to predict the observation 16 steps into the future given the current observation. In Figure 3, we observe that increasing the agent's capacity (in the form of depth) leads to make increasingly more accurate predictions of the future observation. This relationship approximates k-complexity, and we anticipate a stronger relationship for the capacity of a history-encoding agent. Lastly, because of the unobserved region which simulates the infinite state, the agent must not stop learning. In Appendix C, we observed that some agent algorithms did stop learning due to loss of plasticity.

7 Conclusion

In this paper, we have highlighted the need for a big world simulator that accurately mirrors the challenges and complexities inherent in continual learning. We propose two properties which a big world simulator should exhibit: (1) there are no diminishing returns to increasing an agent's capacity, and (2) an optimal finite capacity agent should never stop learning. Furthermore, we reiterate the concrete objective of minimizing average error over an infinite horizon subject to a capacity constraint and provide a decomposition of this objective into two terms which closely resemble the notions of "forgetting" and "plasticity loss" from the literature. Finally, we present a Turing-complete prediction environment as an illustration of a simulator with desired big world properties. We hope that this work stimulates discussion around next steps for designing a simulator that facilitates the discovery of effective continual learning agent design and evaluation metrics which better reflect the objective.

[1] To simulate the infinite state, we add an unobserved region of state and use a periodic boundary condition to join the edge of the unobserved region to the observed region. While an agent with unbounded capacity could observe the infinite state, in our finite simulation, an agent with unbounded capacity could observe its entire history.

References

Zaheer Abbas, Rosie Zhao, Joseph Modayil, Adam White, and Marlos C. Machado. Loss of plasticity in continual deep reinforcement learning. In *Conference on Lifelong Learning Agents*, 2023.

David Abel, Andre Barreto, Benjamin Van Roy, Doina Precup, Hado van Hasselt, and Satinder Singh. A definition of continual reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Matthew Cook et al. Universality in elementary cellular automata. *Complex systems*, 15:1–40, 2004.

Shibhansh Dohare, Richard S. Sutton, and A. Rupam Mahmood. Continual backprop: Stochastic gradient descent with persistent randomness. *CoRR*, abs/2108.06325v3, 2021.

Shibhansh Dohare, J. Fernando Hernandez-Garcia, Parash Rahman, Richard S. Sutton, and A. Rupam Mahmood. Maintaining plasticity in deep continual learning. *CoRR*, abs/2306.13812, 2023.

Shi Dong, Benjamin Van Roy, and Zhengyuan Zhou. Simple agent, complex environment: Efficient reinforcement learning with agent states. *Journal of Machine Learning Research*, 23(255):1–54, 2022.

Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks. *CoRR*, abs/1312.6211v3, 2013.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Joel Jang, Seonghyeon Ye, Changho Lee, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, and Minjoon Seo. Temporalwiki: A lifelong benchmark for training and evaluating ever-evolving language models. In *Conference on Empirical Methods in Natural Language Processing*, 2022.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, 2021.

Saurabh Kumar, Henrik Marklund, Ashish Rao, Yifan Zhu, Hong Jun Jeon, Yueyang Liu, and Benjamin Van Roy. Continual learning as computationally constrained reinforcement learning, 2023a.

Saurabh Kumar, Henrik Marklund, and Benjamin Van Roy. Maintaining plasticity via regenerative regularization. *CoRR*, abs/2308.11958v1, 2023b.

Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on Robot Learning*, 2017.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 2017.

Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks. In *International Conference on Machine Learning*, 2023.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

Ian Osband, Yotam Doron, Matteo Hessel, John Aslanides, Eren Sezener, Andre Saraiva, Katrina McKinney, Tor Lattimore, Csaba Szepesvari, Satinder Singh, Benjamin Van Roy, Richard Sutton, David Silver, and Hado Van Hasselt. Behaviour suite for reinforcement learning. In *International Conference on Learning Representations*, 2020.

Ian Osband, Zheng Wen, Seyed Mohammad Asghari, Vikranth Dwaracherla, Xiuyuan Lu, Morteza Ibrahimi, Dieterich Lawson, Botao Hao, Brendan O'Donoghue, and Benjamin Van Roy. The neural testbed: Evaluating joint predictions. *Advances in Neural Information Processing Systems*, 2022.

Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *European Conference on Computer Vision*, 2020.

Mark Bishop Ring. *Continual learning in reinforcement environments*. The University of Texas at Austin, 1994.

Ryne Roady, Tyler L. Hayes, Hitesh Vaidya, and Christopher Kanan. Stream-51: Streaming classification and novelty detection from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2020.

Rupesh K Srivastava, Jonathan Masci, Sohrob Kazerounian, Faustino Gomez, and Jürgen Schmidhuber. Compete to compute. *Advances in neural information processing systems*, 2013.

Sebastian Thrun. Lifelong learning algorithms. In *Learning to Learn*, pp. 181–209. Springer, 1998.

A. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society, 2*, 42:230–265, 1936.

Huaxiu Yao, Caroline Choi, Bochuan Cao, Yoonho Lee, Pang Wei Koh, and Chelsea Finn. Wild-time: A benchmark of in-the-wild distribution shift over time. In *Neural Information Processing Systems (Datasets and Benchmarks Track)*, 2022.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, 2017.

A Additional Details on Existing Continual Learning Benchmarks

In addition to Permuted MNIST, there are several other benchmarks that follow a similar recipe. The Split MNIST benchmark (Zenke et al., 2017; Nguyen et al., 2018) divides the MNIST dataset into tasks based on digit groupings, requiring the model to learn to distinguish between different sets of numbers sequentially. In the case of Incremental CIFAR-100 (Lopez-Paz & Ranzato, 2017), tasks are created by progressively introducing new classes, testing the model's ability to incorporate new knowledge without forgetting the previously acquired knowledge. This approach to creating a continual learning problem has also been applied to reinforcement learning, in which an agent plays a sequence of games in the Arcade Learning Environment (Abbas et al., 2023).

A.1 Non-synthetic Continual Learning Benchmarks

There are also continual learning problems in which the dataset was curated specifically for the purposes of continual learning. In CoRE50, temporally correlated images for object recognition are introduced corresponding either to new classes or to new instances of an already encountered task (Lomonaco & Maltoni, 2017). Stream51 takes a similar approach, using temporally sequenced frames from natural videos (Roady et al., 2020). Lastly, there have also been efforts to curate datasets with natural non-stationarities that may better reflect the real-world challenges associated with continual learning, such as WILDS (Koh et al., 2021), Wild-time (Yao et al., 2022) and TemporalWiki (Jang et al., 2022).

B Proofs

We provide the proof of Theorem 1 below.

Theorem 1. For all agents $\pi : \mathcal{U} \times \mathcal{X} \mapsto \mathcal{U}$, if for all t, $U_{t+1} = \pi(U_t, X_t)$, then

$$\mathcal{L}_\pi \;=\; \liminf_{T\to\infty} \; \underbrace{\frac{1}{T}\sum_{t=0}^{T-2} \mathbb{I}(X_{T:t+2}; U_t | U_{t+1}, X_{t+1})}_{\text{forgetting}} + \underbrace{\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{I}\left(X_{T:t+1}; X_t | U_t\right)}_{\text{implasticity}}.$$

Proof.

$$\begin{aligned}
\mathcal{L}_\pi &= \liminf_{T\to\infty} \; \mathbb{E}_\pi\left[\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{d}_{\mathrm{KL}}\left(\mathbb{P}(X_{t+1} \in \cdot | H_t) \parallel \mathbb{P}(X_{t+1} \in \cdot | U_t)\right)\right] \\
&= \liminf_{T\to\infty} \; \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{I}(X_{t+1}; H_t | U_t) \\
&= \liminf_{T\to\infty} \; \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{I}(X_{t+1}; U_{0:t}, H_t | U_t) \\
&= \liminf_{T\to\infty} \; \frac{1}{T}\sum_{t=0}^{T-1} \left(\sum_{k=0}^{t-1} \mathbb{I}(X_{t+1}; X_k, U_k | U_{t:k+1}, X_{t:k+1})\right) + \mathbb{I}(X_{t+1}; X_t | U_t) \\
&= \liminf_{T\to\infty} \; \frac{1}{T}\sum_{t=0}^{T-1} \left(\sum_{k=0}^{t-1} \mathbb{I}(X_{t+1;X_k}, U_k | U_{k+1}, X_{t:k+1})\right) + \mathbb{I}(X_{t+1}; X_t | U_t) \\
&\overset{(a)}{=} \liminf_{T\to\infty} \; \frac{1}{T}\sum_{t=0}^{T-1}\sum_{k=0}^{t-1} \; \mathbb{I}(X_{t+1}; U_k | U_{k+1}, X_{t:k+1}) + \mathbb{I}(X_{t+1}; X_k | U_k, U_{k+1}, X_{t:k+1}) \\
&\quad + \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{I}(X_{t+1}, U_{t+1}; X_t | U_t) \\
&= \liminf_{T\to\infty} \; \frac{1}{T}\sum_{k=0}^{T-2} \mathbb{I}(X_{T:k+2}; U_k | U_{k+1}, X_{k+1}) + \mathbb{I}(X_{T:k+2}; X_k | U_k, U_{k+1}, X_{k+1}) \\
&\quad + \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{I}(X_{t+1}, U_{t+1}; X_t | U_t) \\
&\overset{(b)}{=} \liminf_{T\to\infty} \; \frac{1}{T}\sum_{k=0}^{T-2} \mathbb{I}(X_{T:k+2}; U_k | U_{k+1}, X_{k+1}) + \frac{1}{T}\sum_{k=0}^{T-1} \mathbb{I}\left(X_{T:k+1}; X_k | U_k\right),
\end{aligned}$$

where (a) follows from the fact that $\mathbb{I}(U_{t+1}; X_t | U_t, X_{t+1}) = 0$ and (b) follows from the fact that $\mathbb{I}(U_{k+1}; X_k | U_k, X_{T:k+1}) = 0$. □

C Additional Experiments on Turing-compete Prediction Environment

The Turing-complete Prediction Environment is defined by the 3-tuple, $\{S, O, K\}$. The state-size, S, is the size of the entire world. While the Turing-completeness of Rule 110 depends on an infinite initial state, we simulate this with a periodic boundary condition. The dynamics of the local state transition are shown in Figure 4. The observation-size, O, controls the size of the observable world. The unobserved region of the state space is depicted by the shaded region in Figure 2. The prediction-horizon, K, controls the degree to which the non-stationarity induced by partial observability effects the prediction. This is because, for an elementary cellular automaton like Rule 110, each cell in the preceding state can only influence the cells to its left and right in the next state. Thus, the unobserved parts of the world can only influence the next state at the border of the observable region. The influence of the unobserved parts of the world on the prediction increases as the prediction-horizon increases. In our experiments, we set the state space to be 32 dimensional, the observation space to be the first 16 dimensions and prediction horizons $K \in \{1, 2, 4, 8, 16\}$.

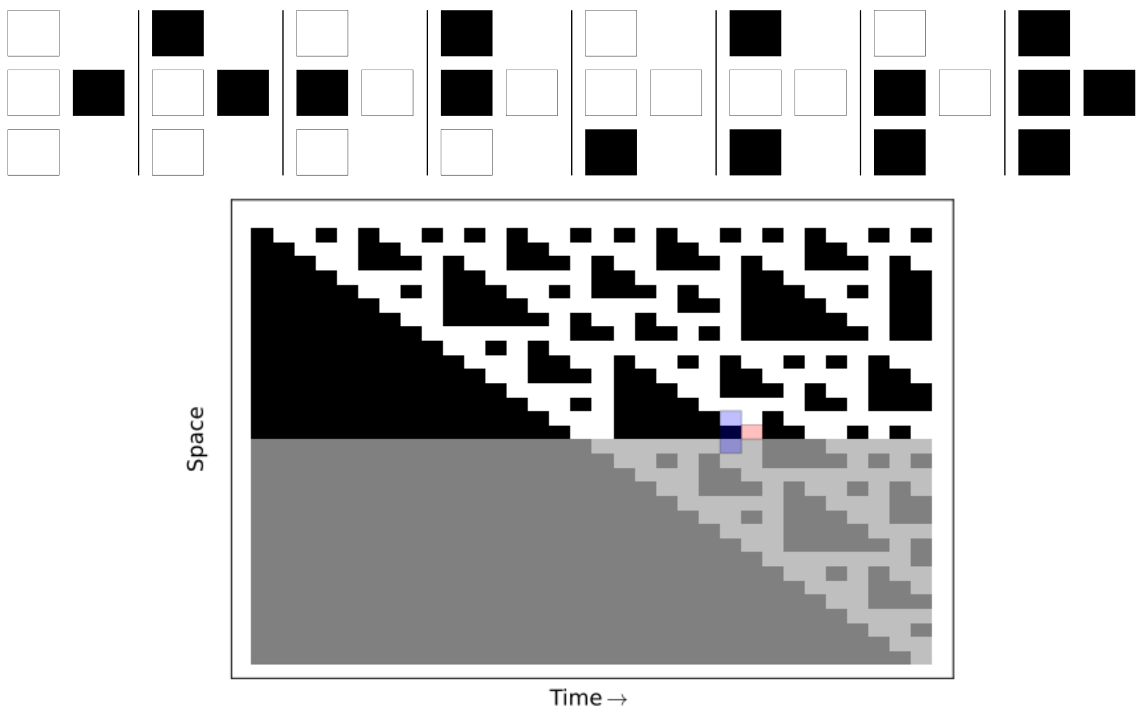

Figure 4: Top: Rule 110 updates each cell of the state using the value of the cell as well as its neighbours' values at the previous time step. (Left-Right): There are 8 possible configurations of the cell and its neighbours. Each configuration determines the cell value of the middle cell in the next state. Bottom: Zooming in on one simulation and the three cells in the blue box at a particular time step, the third rule above is applied to output the middle cell's value at the next time step in the red box. This cell's value on the border of the observable region depends on a cell in the unobservable region.

The environment is initialized to the state corresponding to the binary representation of $\tau = 1$, where τ is a starting state. Observations are provided sequentially for an episode length (or program length) of $T = 100$. After the 100 steps of the program, the environment state is set to the binary representation of the next integer, $\tau \leftarrow \tau + 1$. We reset the state after a finite amount of steps to avoid the environment entering a loop of repeating states. The programs for nearby integers is thus qualitatively similar, but varies for larger integers (see Figure 2). Note that the starting state has a small initial state and an agent with unbounded capacity could encode the dynamics of the full state by compress the entire history

The results are summarized in Figure 5. For each prediction horizon given in the legend, we trained a neural network of different widths and depths to predict the state at the given horizon. On the top figure, we see that longer prediction horizons lead to more non-stationarity. The optimal agent must never stop learning but regularization towards the initialization is needed to sustain plasticity at longer prediction horizons (Kumar et al., 2023b). On the bottom figure, we see that doubling the capacity leads to approximately half the error at larger prediction horizons suggesting that this indeed simulates the big world properties that we have outlined.

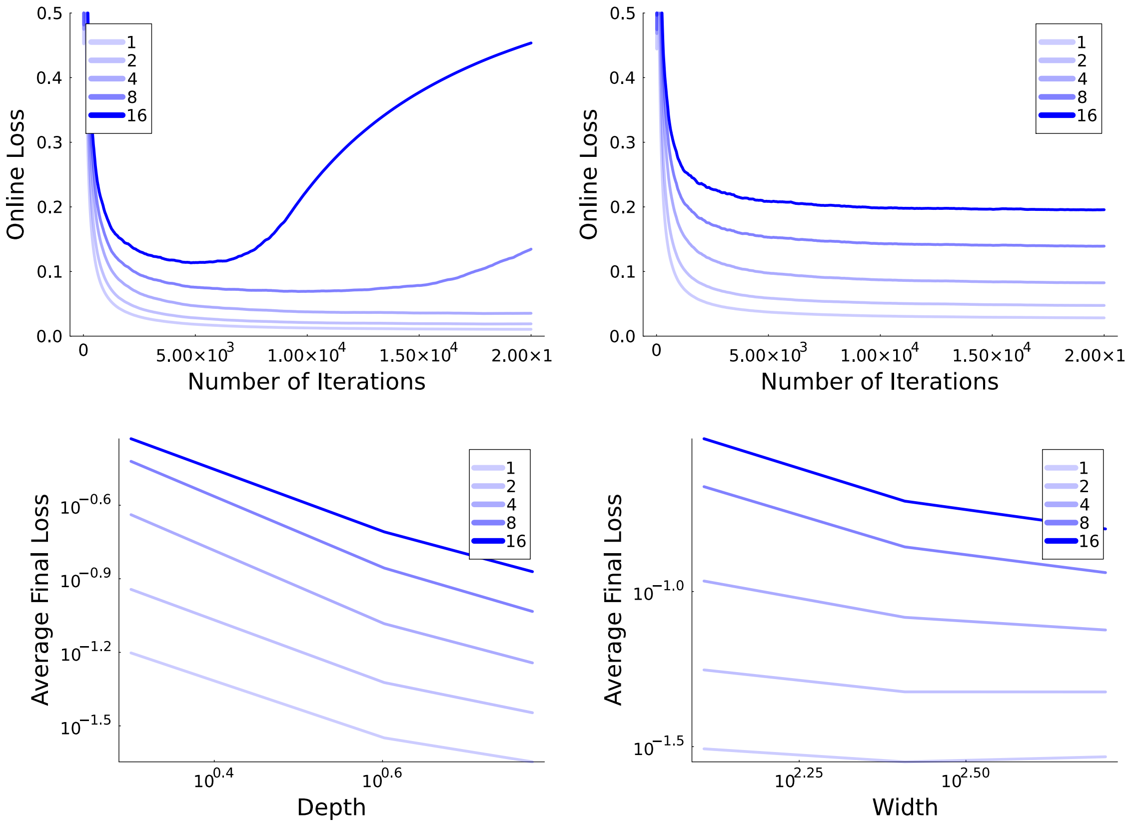

Figure 5: Top: Online accuracy for the medium sized neural network without regularization (left) and with regenerative regularization (right). Bottom: Larger prediction horizons are more difficult to make, but increasing capacity via depth (left) and width (right) leads to performance improvement. The opacity of each line indicates the prediction horizon.