Complexity-calibrated Benchmarks for Machine Learning Reveal When Prediction Algorithms Succeed and Mislead

Sarah E. Marzen,[1,*] Paul M. Riechers,[2,†] and James P. Crutchfield[3,‡]

[1] *W. M. Keck Science Department of Pitzer, Scripps, and Claremont McKenna College, Claremont, CA 91711*
[2] *Beyond Institute for Theoretical Science (BITS), San Francisco, CA*
[3] *Complexity Sciences Center and Physics Department, University of California at Davis, One Shields Avenue, Davis, CA 95616*

(Dated: March 28, 2024)

Abstract Recurrent neural networks are used to forecast time series in finance, climate, language, and from many other domains. Reservoir computers are a particularly easily trainable form of recurrent neural network. Recently, a "next-generation" reservoir computer was introduced in which the memory trace involves only a finite number of previous symbols. We explore the inherent limitations of finite-past memory traces in this intriguing proposal. A lower bound from Fano's inequality shows that, on highly non-Markovian processes generated by large probabilistic state machines, next-generation reservoir computers with reasonably long memory traces have an error probability that is at least $\sim 60\%$ higher than the minimal attainable error probability in predicting the next observation. More generally, it appears that popular recurrent neural networks fall far short of optimally predicting such complex processes. These results highlight the need for a new generation of optimized recurrent neural network architectures. Alongside this finding, we present concentration-of-measure results for randomly-generated but complex processes. One conclusion is that large probabilistic state machines—specifically, large ϵ-machines—are key to generating challenging and structurally-unbiased stimuli for ground-truthing recurrent neural network architectures.

PACS numbers: 02.50.-r 05.45.Tp 02.50.Ey 02.50.Ga

I. INTRODUCTION

Success in many scientific fields centers on prediction. From the early history of celestial mechanics we know that predicting how planetary objects move stimulated the birth of physics. Today, predicting neuronal spiking drives advances in theoretical neuroscience. Outside the sciences, prediction is quite useful as well—predicting stock prices fuels the finance industry and predicting English text fuels social media companies. Recent advances in prediction and generation are so impressive (e.g., GPT-4) that one is left with the impression that time series prediction is a nearly solved problem. As we will show using randomness- and correlation-calibrated data sources, this hopeful state of affairs could not be further from the truth.

Recurrent neural networks [1], of which reservoir computers are a prominent and somewhat recent example [2], have risen to become one of the major tools for prediction. From mathematics' rather prosaic perspective, recurrent neural networks are simply input-dependent dynamical systems. Since input signals to a learning system affect its behavior, over time it can build up a "memory trace" of the input history. This memory trace can then be used to predict future inputs.

There are broad guidelines for how to build recurrent neural networks [1] and reservoir computers that are good predictors [2]. For instance, a linearized analysis shows that one wants to be at the edge of instability [3]. However, a theory of how these recurrent neural networks work optimally is lacking; though see Ref. [4]. Recently, a new architecture was introduced for prediction called a "next-generation reservoir computer", whose memory trace intriguingly only included the last few timesteps of the input, while demonstrating low prediction error with simultaneously small compute power [5].

The general impression from these and many additional reports is that these recurrent neural networks have conquered natural stimuli, including language [6], video [7], and even climate data [8]. They have certainly maximized performance on toy tasks [9, 10] that test long memory. This noted, it is unknown how far they are from optimal performance on the tasks of most importance, such as prediction of language, video, and climate. We need a calibration for how far away they are from nearly-perfect prediction. And this suggests developing a suite of complex processes for which we know the minimal achievable probability of error in prediction.

In the service of this goal, the following adopts the perspective that calibration is needed to understand the limitations inherent in the architecture of the next-generation reservoir computers and to understand how well state-of-the-art recurrent neural networks (including

* smarzen@cmc.edu, Corresponding author
† pmriechers@gmail.com
‡ chaos@ucdavis.edu

next-generation reservoir computers) perform on tasks for which optimal prediction strategies are known. This calibration is provided by time series data generated by a special type of hidden Markov model specialized for prediction called ϵ-*machines*. We find, surprisingly perhaps, that large random multi-state ϵ-machines are an excellent source of complex prediction tasks with which to probe the performance limits of recurrent neural networks.

More to the point, benchmarking on these data demonstrates that reasonably-sized next-generation reservoir computers are inherently performance limited: they achieve no better than a $\sim 60\%$ increase in error probability above and beyond optimal for "typical" ϵ-machine tasks even with a reasonable amount of memory. A key aspect of the calibration is that the optimalities are derived analytically from the ϵ-machine data generators, providing an objective ground truth. This increase in error probability above and beyond the optimal increases to $10^5\%$ if interesting [11, 12] stimuli are used. Altogether, we find that state-of-the-art recurrent neural networks fail to perform well predicting the high-complexity time series generated by large ϵ-machines. In this way, next-generation reservoir computers are fundamentally limited. Perhaps more surprisingly, a more powerful recurrent neural network [9] also has an increase in error probability above and beyond the minimum of roughly 50% for these new prediction benchmarks.

Section II reviews reservoir computers, recurrent neural networks, and ϵ-machines. Section III derives a lower bound on the average rate of prediction errors. Section IV describes a new set of complex prediction tasks and surveys the performance of a variety of recurrent neural networks on these tasks. Section V draws conclusions and proposes new calibration strategies for neural network architectures. Such objective diagnostics should enable significant improvements in recurrent neural networks.

II. BACKGROUND

Section II A describes ϵ-machines and Sec. II B lays out the setup of the typical recurrent neural network (RNN) and reservoir computer (RC).

A. Complex processes and ϵ-machines

Each stationary stochastic process is uniquely represented by a predictive model called an ϵ-*machine*. This one-to-one association is particularly noteworthy as it gives explicit structure to the space of all such processes. One can either explore the space of stationary processes or, equivalently, the space of all ϵ-machines. This is made all the more operational, since ϵ-machines can be efficiently enumerated [13].

In information theory they are viewed as process *generators* and described as minimal unifilar hidden Markov chains (HMC). In computation theory they are viewed as process *recognizers* and described as minimal probabilistic deterministic automata (PDA) [14, 15]. Briefly, an ϵ-machine has hidden states $\sigma \in \mathcal{S}$, referred to as *causal states*, and generates a process by emitting symbols $x \in \mathcal{A}$ over a sequence of state-to-state transitions. For purposes of neural-network comparison in the following, we explore binary-valued processes, so that $\mathcal{A} = \{0, 1\}$. ϵ-Machines are unifilar or "probabilistic deterministic" models since each transition probability $p(\sigma'|x, \sigma)$ from state σ to state σ' given emitted symbol x are singly supported. More simply, there is at most a single destination state. In computation theory this is a deterministic transition in the sense that the model reads in symbols which uniquely determine the successor state. That said, these models are probabilistic as process generators: given that one is in state σ, a number of symbols x can be emitted, each with emission probability $p(x|\sigma)$. In this way, these models represent stochastic languages—a set of output strings each occurring with some probability.

While every stationary process has an ϵ-machine *presentation*, it is usually not finite. An example is shown in Fig. 1 [16]. The finite HMC on the top is nonunifilar since starting in state A and emitting a 0 does not uniquely determine to which state one transits—either A or B. The HMC on the bottom *is* unifilar, since in every state, knowing the emitted symbol uniquely determines the next state. Note that the ϵ-machine for the process generated by the finite nonunifilar HMC has an infinite number of causal states. Also, note that the process has infinite Markov order: if one sees a past of all 0s, one has not "synchronized" to the ϵ-machine's internal hidden state [17], meaning that one does not know which hidden state of the ϵ-machine one is in. And, therefore, there is not a complete one-to-one correspondence between sequences of observed symbols and chains of hidden states. In contrast, with each step in the ϵ-machine presentation one inches closer to a one-to-one correspondence between observed symbols and hidden states—in reality, as close as possible.

In a way, a nonunifilar HMC is little more than a process generator [18] for which the equivalent ϵ-machine presentation has an infinite number of causal states. In another sense, ϵ-machines are a *very* special type of HMC generator since the ϵ-machine's causal states actually represent clusters of pasts that have the same conditional probability distribution over futures [14]. As a result,

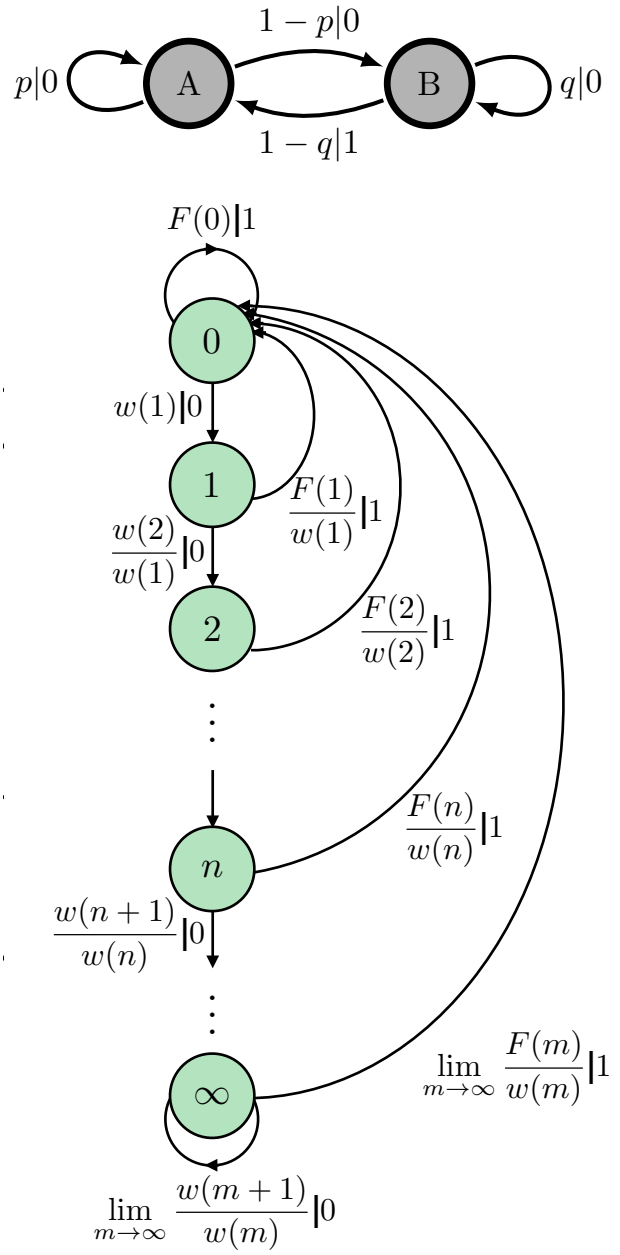

FIG. 1. At top, we see a nonunifilar hidden Markov model that is not an ϵ-Machine because, when in state A, knowing that you have emitted a 0 does not uniquely determine to which state one has transitioned. At bottom, we see the corresponding ϵ-Machine, for which in every state, knowing the emitted symbol uniquely determines the next state. For this ϵ-Machine, we have $F(n) = \begin{cases} (1-p)(1-q)(p^n - q^n)/(p-q) & p \neq q \ , \\ (1-p)^2 n p^{n-1} & p = q \ . \end{cases}$ and $w(n) = \sum_{m=n}^{\infty} F(m)$ [16]. Note that both hidden Markov models generate an identical infinite-order Markov process: if one sees a past of all 0's, one has not "synchronized" to the internal hidden state of the ϵ-Machine. Therefore, there is not a complete one-to-one correspondence between sequences of observed symbols and hidden states.

the casual states and so ϵ-machines are *predictive*.

Consider observing a process generated by a particular ϵ-machine and becoming synchronized so that you know the hidden state. (Now, this happens with probability 1 but it does not always happen [17], as we just described with the nonunifilar HMC example.) Then you can build a prediction algorithm based on the known hidden state. The result, in fact, is *the best possible prediction algorithm that one can build*. Moreover, the latter is simple: when synchronized to hidden state σ, you predict the symbol $\arg\max_x p(x|\sigma)$.

This has one key consequence in our calibrating neural networks: the minimal attainable time-averaged probability $P_{\mathrm{e}}^{\min}$ of error in predicting the next symbol can be explicitly calculated as:

$$P_{\mathrm{e}}^{\min} = \sum_{\sigma} \left[1 - \max_x p(x|\sigma)\right] p(\sigma) \ . \tag{1}$$

(The following considers binary alphabets, so that $1 - \max_x p(x|\sigma) = \min_x p(x|\sigma)$.) We are also able to calculate the entropy rate h_μ directly from the ϵ-machine [14] via:

$$\begin{aligned} h_\mu &= H[X_0|\overleftarrow{X}_0] \\ &= -\sum_{\sigma} p(\sigma) \sum_{x} p(x|\sigma) \log p(x|\sigma) \ . \end{aligned} \tag{2}$$

In contrast, until recently determining h_μ for processes generated by nonunifilar HMCs was intractable. The key advance is that for these processes we recently solved Blackwell's integral equation [19, 20].

B. Recurrent neural networks

Let $s_t \in \mathbb{R}^d$ be the state of the learning system—perhaps a sensor—and let $x_t \in \mathbb{R}^N$ be a time-varying N-dimensional input, both at time t. Discrete-time recurrent neural networks (RNN) are input-driven dynamical systems of the form:

$$s_{t+1} = f_\theta(s_t, x_t) \tag{3}$$

where $f_\theta(\cdot, \cdot)$ is a function of both sensor state s_t and input x_t with parameters θ. See Fig. 2. These parameters are weights that govern how s_t and x_t affect future sensor states $s_{t+1}, s_{t+2}, \ldots$. Alternative RNN architectures result from different choices of f. See below. Memory units (LSTMs) and Gated Recurrent Units (GRUs) are often used to optimize prediction of input. For simplicity, the following considers scalar time series: $N = 1$.

Generally, RNNs are hard to train, both in terms of required data sample size and compute resources (mem-

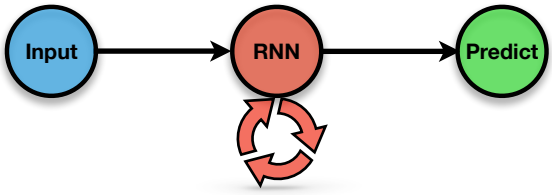

FIG. 2. A recurrent neural network (RNN) for which the future state of the recurrent node depends on its previous state and the current input. The present state of the recurrent node is then used to make a prediction.

ory and time) [21]. RCs [2, 22], also known as *echo state networks* [23] and *liquid state machines* [24, 25], were introduced to address these challenges.

RCs involve two components. The first is a *reservoir*—an input-dependent dynamical system with high dimensionality d as in Eq. (3). And, the second is a *readout layer* $\hat{u}$—a simple function of the hidden reservoir state. Here, the readout layer typically employs logistic regression:

$$P(\hat{u}|s) = \frac{e^{a_{\hat{u}}^\top s + b_{\hat{u}}}}{\sum_{\hat{u}'} e^{a_{\hat{u}'}^\top s + b_{\hat{u}'}}} ,$$

with regression parameters $a_{\hat{u}}$ and $b_{\hat{u}}$. To model binary-valued processes, our focus here, we have:

$$P(\hat{u} = 1|s) = \frac{e^{a^\top s + b}}{1 + e^{a^\top s + b}}.$$

The regression parameters are easily trained and can include regularization if desired. Note that while s was used as the input into the logistic regression probabilities, one can move to nonlinear readout by also using $ss^\top$ to inform the logistic regression probabilities.

The following compares several types of RNNs: 'typical' RCs, 'next generation' RCs, and LSTMs.

1. 'Typical' RCs

In the following, as a model of typical RCs, a subset of RC nodes are updated linearly, while others are updated according to a $\tanh(\cdot)$ activation function. Let $s = \begin{pmatrix} s^{\mathrm{nl}} \\ s^{\mathrm{l}} \end{pmatrix}$. We have:

$$s_{t+1}^{\mathrm{nl}} = \tanh\left(W^{\mathrm{nl}} s_t^{\mathrm{nl}} + v^{\mathrm{nl}} x_t + b^{\mathrm{nl}}\right) ,$$

and

$$s_{t+1}^{\mathrm{l}} = W^{\mathrm{l}} s_t^{\mathrm{l}} + v^{\mathrm{l}} x_t + b^{\mathrm{l}},$$

where $v^{\mathrm{l,nl}}$ controls how strongly the input affects the state, $b^{\mathrm{l,nl}}$ is a bias term, and $W^{\mathrm{l,nl}}$ are the weight matrices.

The weight matrices $W^{\mathrm{l,nl}}$ are chosen to have: 0 entries based on a small-world network of density 0.1 and $\beta = 0.1$; nonzero entries normally distributed according to the standard normal; and a spectral radius ~ 0.99 to guarantee the RC fading-memory condition [23]. Different recipes for choosing which nodes were connected (small-world networks with varying β and density), what distribution the weights were drawn from (normal versus uniform), and whether or not there was a bias term were tried. These variations had virtually no effect on the results. The only thing that clearly made a difference was whether or not the readout was linear (logistic regression) or nonlinear (logistic regression on a concatenation of s and $ss^\top$. These two cases are clearly demarcated in the appropriate figure. There was no bias term in the figure shown.

2. 'Next generation' RCs

Next-generation RCs employ a simple reservoir that tracks some amount of input history and a more complex readout layer [5] to improve accuracy over RC's universal approximation property. The reservoir records inputs from the last m timesteps and, then, uses a readout layer consisting of polynomials of arbitrary order. Technically, next-generation RCs are a subset of general RCs in that a reservoir can be made into a shift register that records the last m timesteps. As introduced in Ref. [5] next-generation RCs solve a regression task, but they can easily be modified to solve classification tasks. The following simply takes second-order polynomial combinations of the last m timesteps and uses those as features for the logistic regression layer. In other words, let $s_t = (x_t, x_{t-1}, ..., x_{t-m+1})$, a column vector, be the state of the reservoir; then we use s_t and $s_t^\top s_t$ as input to the logistic regression.

3. LSTMs

In contrast, *long short-term memory networks* (LSTM) [9] take a different approach by optimizing f_θ for training and for retaining memory. There, s is a combination of several hidden states and the update equations for the network are given in Ref. [9]. An LSTM's essential components consist of linearly-updated *memory cells* that make training easier and avoid exploding or vanishing gradients and a *forget gate* that may improve performance by allowing the network to access a range

of timescales [4].

III. PREDICTION ERROR BOUNDS

No matter the RNN, the conditional entropy of the next input symbol X_t given the learning system's state S_t,

$$H[X_0|S_0] = -\sum_{s_t} p(s_t) \sum_{x_t} p(x_t|s_t) \log p(x_t|s_t) \ ,$$

places a fundamental upper bound on the RNN prediction performance through Fano's inequality:

$$H[X_0|S_0] \leq H_{\rm b}\left(P_{\rm e}\right) + P_{\rm e} \log\left(|\mathcal{A}| - 1\right) \ .$$

In this, $P_{\rm e}$ is the time-averaged probability of making an error in *predicting* the next symbol x_t from RNN's state s_t, and $H_{\rm b}$ is the binary entropy function. We have also invoked stationarity of the time series, to remove the dependence on t in the steady-state operation of the RNN. In particular, for a binary process where $|\mathcal{A}| = 2$:

$$P_{\rm e} \geq H_{\rm b}^{-1}[H(X_0|S_0)] \ ,$$

where $H_{\rm b}^{-1}$, defined on the domain $[0, 1]$, is the inverse of $H_{\rm b}$ on its monotonically increasing domain $[0, 1/2]$.

In other words, the measure of RNN performance is given by a function of $H[X_0|S_0]$ that lower bounds $P_{\rm e}$, coupled with the minimal attainable probability of error calculable directly from the ϵ-machine as described in Sec. II. The lower the model's conditional entropy, the better prediction performance. For any RNN, due to the Markov chain $S_0 \to \overleftarrow{X}_0 \to X_0$, this cannot be lower than h_μ—the entropy rate:

$$\begin{aligned} H[X_0|S_0] &\geq H[X_0|\overleftarrow{X}_0] \\ &= h_\mu \ . \end{aligned}$$

Notably, the next-generation RC takes into account only the last m timesteps, so that:

$$\begin{aligned} H[X_0|S_0] &= H[X_0|\overleftarrow{X}_0^m] \\ &= h_\mu(m), \end{aligned}$$

where the *myopic entropy rate* $h_\mu(m) \geq h_\mu$ is discussed at length in Refs. [26].

IV. RESULTS

We are now ready to calibrate RNN and RC performance on the task of time-series prediction. First, we survey the performance of RCs when predicting a random sample of typical complex stochastic processes. Second, we explore RC performance on an "interesting" complex process—one from the family of memoryful renewal processes—hidden semi-Markov processes with infinite Markov order. Third and finally, we compare the prediction performance of RCs, next-generation RCs, and LSTM RNNs on a large suite of complex stochastic processes.

A. Limits of Next-Generation RCs Predicting "Typical" Processes

We construct exemplars of "typical" complex processes by sampling the space of ϵ-machines as follows:

- An arbitrary large number of candidate states is chosen for the HMC stochastic process generator. This parallels the fact that most processes have an infinite number of causal states [15, 27];
- For each (σ, x) pair, a labeled transition $\sigma \xrightarrow{x} \sigma'$ is randomly generated, with the destination state σ' chosen from the uniform distribution over candidate states;
- Symbol emission probabilities $p(x|\sigma)$ are randomly generated from a Dirichlet distribution with uniform concentration parameter α;
- We retain the largest recurrent component of this construction as our sample ϵ-machine.

Numerically, we find that approximately 20% of the candidate states become transients in the constructed directed network, which are then trimmed from the final ϵ-machine. This number of transients strongly clusters around 20% as the number of candidates grows large. (Note that this is a topological feature, independent of α.) Moreover, this candidate network typically has a single recurrent component. Accordingly, the resulting causal states typically number about 80% of the candidate states in our construction, as the number of candidate states grows large.

This results in a finite-state unifilar HMC or, equivalently, a presentation that can generate a process with a finite number of causal states. Interestingly, though, the process generated is *usually* infinite-order Markov [28]. This can be seen from the mixed-state presentation that describes optimal prediction [20, 26], whose transient states of uncertainty generically maintain nonzero probability even after arbitrarily long observation time. [This is typical even when the mixed-state presentation has a finite number of transient states. Adding a further challenge to the task of prediction, though, the mixed-

state presentation typically has infinitely many transient states.]

An expression for the myopic entropy rate $h_\mu(m)$ was developed in Ref. [26] that allows one to exactly compute $h_\mu(m)$ from the generating ϵ-machine's mixed-state presentation. However, for binary-valued processes it was more straightforward to explicitly enumerate possible length-m futures. Note, though, that this is impractical for the trajectory lengths used here if the emitted-symbol alphabet is larger than two. Figure 3(top) shows $h_\mu(m)$ as a function of m, in the case that $\alpha = 1$. Figure 3(bottom) shows percentage increases in the P_e lower bounds for next-generation RCs above and beyond the minimal $P_\mathrm{e}^\mathrm{min}$, tracking prediction error lower bounds given by Fano's inequality in Sec. IV.

Across this family of stochastic processes, typical values of the myopic entropy rate $h_\mu(m)$ and the entropy rate h_μ exhibit a concentration of measure as the number of causal states grows large, with values clustering around 1/2 nat (not shown here). Typical values of the percentage increase in the P_e above and beyond the minimal $P_\mathrm{e}^\mathrm{min}$ show a concentration of measure, and the minimum probability $P_\mathrm{e}^\mathrm{min}$ of error cluster around 1/4 (not shown here), reminiscent of the process-survey results reported by Ref. [29].

A quick plausibility argument suggests that there is a genuine concentration of measure for these two quantities, using the formulae in Sec. II. Roughly speaking, when the ϵ-machine generator has a large number of causal states, the transitions from any particular state have little effect on the stationary state distribution $p(\sigma)$. Hence, h_μ and $P_\mathrm{e}^\mathrm{min}$ are roughly the sum of N i.i.d. random variables. The Central Limit Theorem dictates for the concentration parameter $\alpha = 1$ that h_μ estimates should cluster around 1/2 nat and that $P_\mathrm{e}^\mathrm{min}$ should cluster around 1/4. In contrast, $H[X_0]$ has the larger expected value of ln(2) nats, which becomes typical as the number of causal states grows large. The gradual decay of uncertainty from ln(2) to 1/2 nat per symbol can only be achieved by predictors that (at least implicitly) synchronize to the latent state of the source via distinguishing long histories.

These typical processes are surprisingly non-Markovian, exhibiting infinite-range correlation. A process' degree of non-Markovianity is reflected in how long it takes for $h_\mu(m)$ to converge to h_μ: how large must m be to synchronize? Even after observing $m = 15$ symbols, these processes (with a finite but large number of causal states) are still ~ 0.2 nats away from synchronization. This convergence failure contributes to a minimal probability of error that cannot be circumvented *no matter the cleverness in choosing the RC nonlinear readout function.*

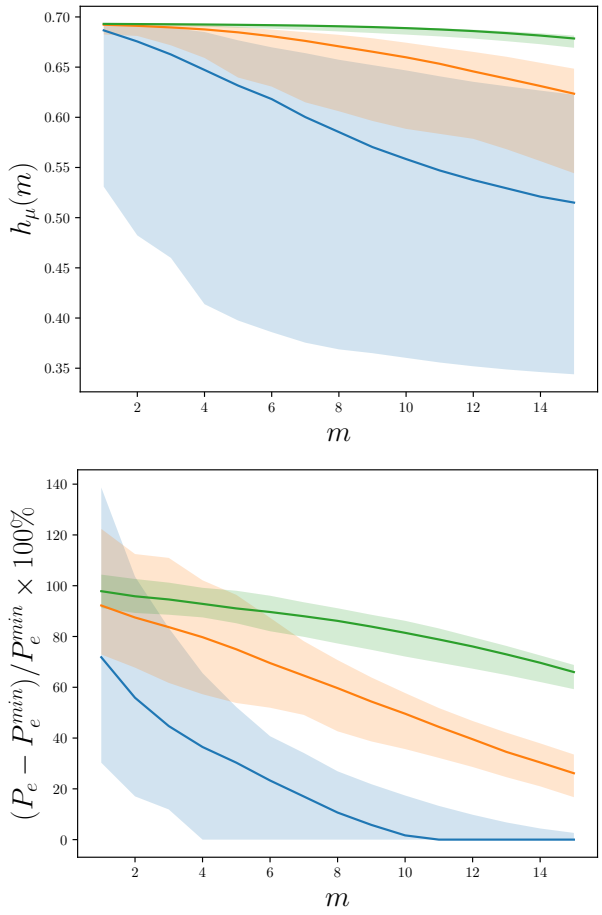

FIG. 3. (Top) Finite-length entropy rate $h_\mu(m)$ in nats for typical random unifilar HMCs constructed with 30 (blue), 300 (orange), and 3000 (green) candidate states as a function of the number of input timesteps m. (Bottom) Increase of the lower bound on the probability P_e of error from Fano's inequality, above and beyond $P_\mathrm{e}^\mathrm{min}$, with the same random unifilar HMCs as a function of the number of input timesteps m. Since occassionally the lower bound on this quantity fell below 0%, the maximum of 0% and the quantity is used. 90% confidence intervals are shown on both graphs.

B. Limits of Next-Generation RCs Predicting an "Interesting" Process

References [11, 12] define complex and thus "interesting" processes as those that have infinite mutual information between past and future—the so-called "predictive information" or "excess entropy". The timescales of predictability are revealed through the growth $I_\mathrm{pred}(m)$ as longer length-m blocks of history and future are taken into account. The *predictive information* is:

$$I_\mathrm{pred}(m) = \sum_{l=0}^{m} \left[h_\mu(l) - h_\mu\right] \ .$$

And so, its growth rate is:

$$I_{\text{pred}}(m+1) - I_{\text{pred}}(m) = \sum_{l=0}^{m+1}[h_\mu(l) - h_\mu] - \sum_{l=0}^{m}[h_\mu(l) - h_\mu]$$
$$= h_\mu(m+1) - h_\mu \ .$$

That is:

$$h_\mu(m+1) = h_\mu + I_{\text{pred}}(m+1) - I_{\text{pred}}(m) \ .$$

The gap between $h_\mu(m+1)$ and h_μ quantifies the excess uncertainty in the next observable, due to observation of only a finite-length past. This is governed by $I_{\text{pred}}(m+1) - I_{\text{pred}}(m)$ in discrete-time processes or, analogously, by $dI_{\text{pred}}(t)/dt$ in continuous-time processes.

What constitutes an acceptable increase in prediction error above and beyond h_μ? The intuition for this follows from inverting Fano's inequality to determine the additional conditional entropy implied by a substantial increase in the probability of error.

To illustrate this, we turn to an interesting process that has a very slow gain in predictive information—the discrete-time renewal process shown in Fig. 1(Bottom), with survival function:

$$w(n) = \begin{cases} 1 & n = 0 \\ n^{-\beta} & n \geq 1 \end{cases} \ .$$

Discrete- and continuous-time renewal processes are encountered broadly—in the physical, chemical, biological, and social sciences and in engineering—as sequences of discrete events consisting of an event type and an event duration or magnitude. An example critical to infrastructure design occurs in the geophysics of crustal plate tectonics, where the event types are major earthquakes tagged with duration time, time between their occurrence, and an approximate or continuous *Richter magnitude* [30]. Another example is seen in the history of reversals of the earth's geomagnetic field [31]. In physical chemistry they appear in single-molecule spectroscopy which reveals molecular dynamics as hops between conformational states that persist for randomly distributed durations [32, 33]. A familiar example from neuroscience is found in the spike trains generated by neurons that consist of spike-no-spike event types separated by *interspike intervals* [34]. Finally, a growing set of renewal processes appear in the quantitative social sciences, in which human communication events and their durations are monitored as signals of emergent coordination or competition [35].

At $\beta = 1$, this discrete-time renewal process has $I_{\text{pred}}(m) \sim \log\log m$ [36]. The minimal achievable lower bound is $P_{\text{e}}^{\text{min}} = 0.0001$. Due to an additional ~ 0.1 nats

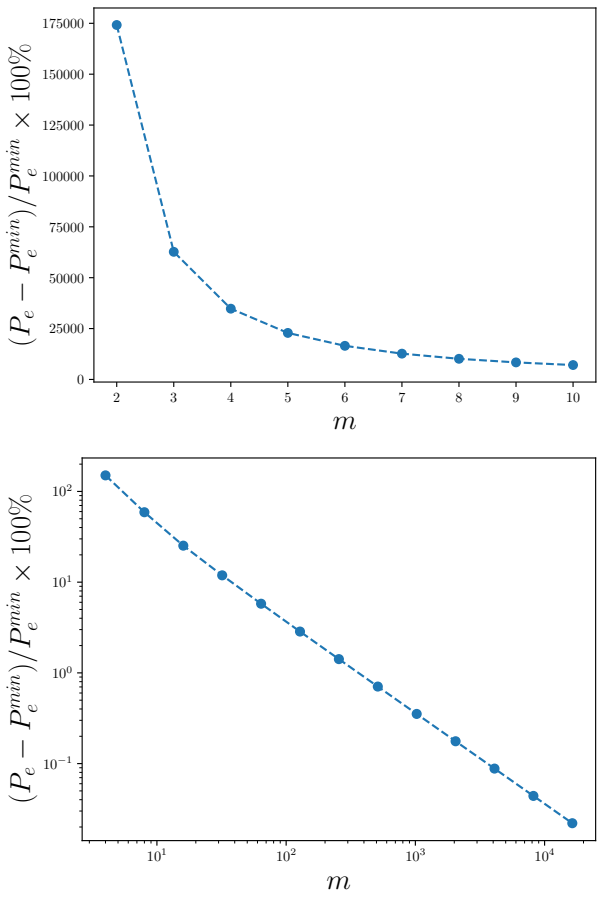

FIG. 4. Predicting a discrete-time fractal renewal process with infinite excess entropy: (Top) Percentage increase in the lower bound for the probability of error P_{e} above and beyond the minimum using Fano's inequality as a function of time steps m. (Bottom) Percentage increase in the lower bound for the probability of error P_{e} above and beyond the minimum using Fano's inequality as a function of time steps m for a process such as that in Ref. [36].

from not using an infinite-order memory trace and instead only using the last $m = 5$ symbols, the probability-of-error lower bound jumps to 0.02. This is a percentage increase in probability of error of $10^4\%$ at $m = 11$ timesteps—about two and half orders of magnitude worse than that of a typical complex process. We emphasize these are fundamental bounds that no amount of cleverness can circumvent. While any nonlinear readout function might be chosen for a next-generation RC, the process' inherent complexity demands that an infinite-order memory trace be used for relatively good prediction.

References [37, 38] constructed an HMC that ergodically [39] generated $I_{\text{pred}}(m) \sim \log m$. For this process:

$$h_\mu(m+1) \approx h_\mu + \frac{1}{m} \ .$$

Consider a process that has a "typical" entropy rate of 0.5 nats, we can invert Fano's inequality—that is not necessarily tight—to find a lower bound on the probability of error with an infinite memory trace. Assuming this lower bound, the bound on the percentage increase of the probability of error above and beyond $P_{\mathrm{e}}^{\mathrm{min}}$ decays to 10% only when the RC uses more than $m = 1000$ symbols. See Fig. 4(bottom).

C. RCs, Next-Generation RCs, and State-of-the-Art RNNs Predicting Highly non-Markovian Processes

Knowing that there are fundamental limits to the next-generation RC's ability to predict processes forces the question: how well do next-generation RCs actually do at predicting these processes when using second-order polynomial readout? Moreover, do more traditional RCs and state-of-the-art RNNs do any better?

In all experiments, we are careful to hold the number of input nodes to the readout constant for a fair comparison.

We now compare typical RCs with linear readout, typical RCs with nonlinear readout (second-order polynomial), and LSTMs to next-generation RCs on prediction tasks generated by the large ϵ-machines of Sec. IV A. Although RCs with nonlinear readout and many more nodes outperform next-generation RCs, Fig. 5 shows that when the number of readout nodes is held constant, next-generation RCs are indeed the best RC possible. This is expected from Ref. [5]. LSTMs beat all reservoir computers, however, as one can see from the red violin plot of Fig. 5 settling primarily on the lowest possible values of $(P_e - P_e^{min})/P_e^{min} \times 100\%$. This is somewhat expected since LSTMs optimize both the reservoir and readout, although the fact that they do is a testament to the fact that the successful training of the reservoir using back-propagation through time [40].

Figure 5's surprise is that all RNNs perform quite poorly, leaving at least $\sim 50\%$ increase in the probability of error above and beyond optimal, as one can see from the surprisingly large values on the y-axis, achieved at $m = 10$ for the next-generation RC. This nearly saturates the lower bound on this percentage increase in the probability of error placed by Fano's inequality.

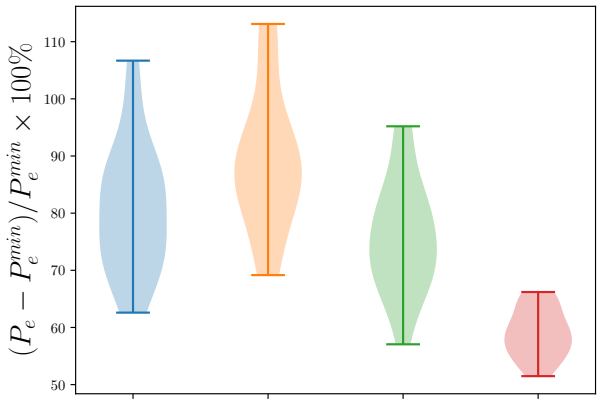

FIG. 5. Percentage increase in the probability of error of trained next generation RCs (green), trained RCs with linear readout (orange), trained RCs with nonlinear readout (blue), and trained LSTMs (red) above and beyond $P_{\mathrm{e}}^{\mathrm{min}}$ for 100 ϵ-machines with 300 candidate states. The next-generation RC has 10 timesteps as input; the typical RC with nonlinear readout has 10 nodes with 5 linear nodes; the typical RC with linear readout has 110 nodes with 10 linear nodes; and the LSTM has 110 nodes. The number of nodes has been chosen so that the number of readout nodes is equivalent across machines. Note that these nearly saturate the lower bound provided by Fano's inequality.

V. CONCLUSION

The striking advances made by RNNs in predicting a very wide range of systems—from language to climate—have not been accompanied by markedly improved explorations of how much structure they fail to predict. Here, we introduced and illustrated such a calibration.

We addressed the task of leveraging past inputs to forecast future inputs, for any stochastic process. We showed that $P_{\mathrm{e}}^{\mathrm{min}}$—the minimal time-averaged probability of incorrectly guessing the next input, minimized over *all* possible strategies that can operate on historical input—can be directly calculated from a data source's generating ϵ-machine. This provides a benchmark for all possible prediction algorithms. We compared this optimal predictive performance with a lower bound on various RNNs' P_{e}—the actual time-averaged probability of incorrectly guessing the next input, given the state of the model. We found that so-called next-generation RCs are fundamentally limited in their performance. And we showed that this cannot be improved on via clever readout nonlinearities.

In our comparison of various prediction models, we tested next-generation RCs with highly-correlated inputs that are challenging to predict. This input data was generated from large ϵ-machines. The ϵ-machines are the optimal prediction algorithm, and the minimal probability of error for these data are known in closed-form. Our extensive surveys showed, surprisingly, that models from RCs with linear readout to next-generation RCs of reasonable size to LSTMs all have a probability of prediction error that is $\sim 50\%$ greater than the theoretical minimal probability of error.

The fact that simple large random ϵ-machines generate

such challenging stimuli might be a surprise. Recently, though, it was reported that tractable ϵ-machines can lead to "interesting" processes [11, 12]. We showed that these processes provide even more of a challenge for next-generation RCs.

At first, it may seem that this new calibration is somewhat useless, both theoretically and from a practical point of view. For instance, it is perhaps not surprising that RCs, NGRCs, and maybe even LSTMs perform poorly on highly non-Markovian processes such as the ones used in this paper. However, with N nodes, one can find N predictive features that potentially reach far back into the past even though one might naively think that the N features correspond to the last N time points. Secondly, the processes used here are not of general interest, as large random ϵ-machines do not correspond to real-world signals in structure. However, one can manufacture ϵ-machines that do have the structure of real-world signals, as any real-world signal can be represented by an ϵ-machine. Then, the calibration here can improve the RC or RNN's ability to predict real-world signals. This potential research program extends even to non-stationary real-world data. Both natural language and natural data from the physical world can be understood as stochastic processes which, in principle, have some epsilon machine representation. While we focused on stationary processes in this manuscript, non-stationary processes can be accommodated via simple adaptations of our methods, where the unifilar HMM of the process would have a unique start state and possible absorbing states.

Finally, next-generation RCs—that do indeed outperform typical RCs with the same number of readout nodes—are fundamentally limited in prediction performance by the nature of their limited memory traces. We suggest that effort should be expended to optimize standard RCs that do not suffer from the same fundamental limitations—so that memory becomes properly incorporated and typical performance improves.

ACKNOWLEDGMENTS

The authors thank the Telluride Science Research Center for hospitality during visits and the participants of the Information Engines Workshops there. JPC acknowledges the kind hospitality of the Santa Fe Institute, Institute for Advanced Study at the University of Amsterdam, and California Institute of Technology for their hospitality during visits. This material is also based upon work supported by, or in part by, the Air Force Office of Scientific Research award FA9550-19-1-0411, Templeton World Charity Foundation grant TWCF0570, Foundational Questions Institute and Fetzer Franklin Fund grant FQXI-RFP-CPW-2007, U.S. Army Research Laboratory and the U.S. Army Research Office grants W911NF-21-1-0048 and W911NF-18-1-0028, and U.S. Department of Energy grant DE-SC0017324.

DATA AVAILABILITY STATEMENT

The datasets used and/or analysed during the current study are available from the corresponding author on reasonable request.

REFERENCES

[1] Z. C. Lipton, J. Berkowitz, and C. Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[2] B. Schrauwen, D. Verstraeten, and J. Van Campenhout. An overview of reservoir computing: theory, applications and implementations. In *Proceedings of the 15th european symposium on artificial neural networks. p. 471-482 2007*, pages 471–482, 2007.

[3] A. Hsu and S. E. Marzen. Strange properties of linear reservoirs in the infinitely large limit for prediction of continuous-time signals. *Journal of Statistical Physics*, 190(2):1–16, 2023.

[4] K. Krishnamurthy, T. Can, and D. J. Schwab. Theory of gating in recurrent neural networks. *Physical Review X*, 12(1):011011, 2022.

[5] D. J. Gauthier, E. Bollt, A. Griffith, and W. A. S. Barbosa. Next generation reservoir computing. *Nature communications*, 12(1):1–8, 2021.

[6] M. Zhang and J. Li. A commentary of gpt-3 in mit technology review 2021. *Fundamental Research*, 1(6):831–833, 2021.

[7] Y. Zhou, H. Dong, and A. El Saddik. Deep learning in next-frame prediction: A benchmark review. *IEEE Access*, 8:69273–69283, 2020.

[8] Y. Chen, Q. Cheng, Y. Cheng, H. Yang, and H. Yu. Applications of recurrent neural networks in environmental factor forecasting: a review. *Neural computation*, 30(11):2855–2881, 2018.

[9] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[10] H. Jaeger. Long short-term memory in echo state networks: Details of a simulation study. Technical report, Jacobs University Bremen, 2012.

[11] W. Bialek, I. Nemenman, and N. Tishby. *Neural Comp.*, 13:2409–2463, 2001.

[12] W. Bialek, I. Nemenman, and N. Tishby. Complexity through nonextensivity. *Physica A*, 302:89–99, 2001.

[13] B. D. Johnson, J. P. Crutchfield, C. J. Ellison, and C. S. McTague. arxiv.org:1011.0036.

[14] C. R. Shalizi and J. P. Crutchfield. *J. Stat. Phys.*, 104:817–879, 2001.

[15] D. Pfau, N. Bartlett, and F. Wood. Probabilistic deterministic infinite automata. In *Adv. Neural Info. Proc. Sys.*, pages 1930–1938, 2010.

[16] S. Marzen and J. P. Crutchfield. *Entropy*, 17(7):4891–4917, 2015.

[17] J. P. Crutchfield, C. J. Ellison, J. R. Mahoney, and R. G. James. Synchronization and control in intrinsic and designed computation: An information-theoretic analysis of competing models of stochastic computation. *CHAOS*, 20(3):037105, 2010.

[18] W. Löhr and N. Ay. On the generative nature of prediction. *Advances in Complex Systems*, 12(02):169–194, 2009.

[19] D. Blackwell. The entropy of functions of finite-state markov chains. In *Transactions of the first Prague conference on information theory, Statistical decision functions, random processes held at Liblice near Prague from November*, volume 28, pages 13–20, 1957.

[20] A. Jurgens and J. P. Crutchfield. Shannon entropy rate of hidden Markov processes. *J. Statistical Physics*, 183(32):1–18, 2020.

[21] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.

[22] M. Lukoševičius and H. Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

[23] H. Jaeger. *Short term memory in echo state networks*, volume 5. GMD-Forschungszentrum Informationstechnik, 2001.

[24] W. Maass. Liquid state machines: motivation, theory, and applications. *Computability in context: computation and logic in the real world*, pages 275–296, 2011.

[25] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[26] P. M. Riechers and J. P. Crutchfield. Spectral simplicity of apparent complexity. i. the nondiagonalizable metadynamics of prediction. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(3):033115, 2018.

[27] S. E. Marzen and J. P. Crutchfield. Nearly maximally predictive features and their dimensions. *Phys. Rev. E*, 95(5):051301(R), 2017. SFI Working Paper 17-02-007; arxiv.org:1702.08565 [cond-mat.stat- mech].

[28] R. G. James, J. R. Mahoney, C. J. Ellison, and J. P. Crutchfield. Many roads to synchrony: Natural time scales and their algorithms. *Phys. Rev. E*, 89:042135, 2014.

[29] D. P. Feldman, C. S. McTague, and J. P. Crutchfield. The organization of intrinsic computation: Complexity-entropy diagrams and the diversity of natural information processing. *CHAOS*, 18(4):043106, 2008.

[30] T. Akimoto, T. Hasumi, and Y. Aizawa. Characterization of intermittency in renewal processes: Application to earthquakes. *Phys. Rev. E*, 81:031133, 2010.

[31] R. W. Clarke, M. P. Freeman, and N. W. Watkins. Application of computational mechanics to the analysis of natural data: An example in geomagnetism. *Phys. Rev. E*, 67, 2003.

[32] C.-B. Li and T. Komatsuzaki. Aggregated Markov model using time series of a single molecule dwell times with a minimum of excessive information. *Phys. Rev. Lett.*, 111:058301, 2013.

[33] C.-B. Li, H. Yang, and T. Komatsuzaki. Multiscale complex network of protein conformational fluctuations in single-molecule time series. *Proc. Natl. Acad. Sci. USA*, 105:536–541, 2008.

[34] S. Marzen, M. R. DeWeese, and J. P. Crutchfield. Time resolution dependence of information measures for spiking neurons: Scaling and universality. *Front. Comput. Neurosci.*, 9:109, 2015.

[35] D. Darmon, J. Sylvester, M. Girvan, and W. Rand. Predictability of user behavior in social media: Bottom-up versus top-down modeling. *arXiv.org:1306.6111*.

[36] S. Marzen and J. P. Crutchfield. *Phys. Lett. A*, 380(17):1517–1525, 2016.

[37] N. Travers and J. P. Crutchfield. Infinite excess entropy processes with countable-state generators. *Entropy*, 16:1396–1413, 2014.

[38] L. Debowski. On hidden Markov processes with infinite excess entropy. *J. Theo. Prob.*, pages 1–13, 2012.

[39] J. P. Crutchfield and S. Marzen. *Phys. Rev. E*, 91(5):050106, 2015.

[40] M. C. Mozer. A focused backpropagation algorithm for temporal. *Backpropagation: Theory, architectures, and applications*, 137, 1995.