nature physics

ARTICLES

https://doi.org/10.1038/s41567-020-1018-2

Machine learning the thermodynamic arrow of time

Alireza Seif[1,2]✉, Mohammad Hafezi[1,2,3] and Christopher Jarzynski[1,4]

The asymmetry in the flow of events that is expressed by the phrase 'time's arrow' traces back to the second law of thermodynamics. In the microscopic regime, fluctuations prevent us from discerning the direction of time's arrow with certainty. Here, we find that a machine learning algorithm that is trained to infer the direction of time's arrow identifies entropy production as the relevant physical quantity in its decision-making process. Effectively, the algorithm rediscovers the fluctuation theorem as the underlying thermodynamic principle. Our results indicate that machine learning techniques can be used to study systems that are out of equilibrium, and ultimately to answer open questions and uncover physical principles in thermodynamics.

The microscopic dynamics of physical systems are time-reversible, but the macroscopic world clearly does not share this symmetry. When we are shown a movie of a macroscopic process, we can typically guess easily whether the movie is played in the correct order or in time-reversed order. In 1927, Sir Arthur Eddington coined the phrase 'time's arrow' to express this asymmetry in the flow of events, arguing that it traces back to the second law of thermodynamics[1]. Almost a century later, advances in statistical mechanics have extended our understanding of this problem to the microscopic regime. There, fluctuations prevent us from discerning the direction of time's arrow with certainty[2–4]. Instead, the probability that a movie is being shown in the correct chronological order is determined by the energy that is dissipated during the process, as expressed by equation (2) below. This prediction, which has been experimentally verified[5], is equivalent to the Crooks fluctuation theorem[6,7], an important result in modern non-equilibrium statistical mechanics[3,8]. The recent success of applications of machine learning and artificial intelligence in physics begs the question of whether these techniques can speed up scientific discovery[9,10]. Machine learning methods have emerged as exciting tools to study problems in statistical and condensed matter physics, such as classifying phases of matter, detecting order parameters and generating configurations of a system from observed data[11–27].

Here, we apply machine learning to the problem of time's arrow, within the framework of non-equilibrium statistical mechanics. We show that a machine can learn to guess accurately the direction of time's arrow from microscopic data, and (more importantly) that it does so by effectively discovering the underlying thermodynamics, identifying dissipated work as the relevant quantity and correctly establishing its relation to time's arrow. Remarkably, the main machine learning tool used here, logistic regression, was developed decades before the derivation of fluctuation theorems by human experts[28,29]. This suggests that a data-driven approach could have led to an earlier discovery of these theorems. Moreover, we show that the machine can generate representative trajectories for forward and backward time directions correctly. Finally, we design a neural network that can detect the underlying process and classify the direction of time's arrow at the same time.

We first introduce the relevant physical laws that govern microscopic, non-equilibrium fluctuations, and briefly review the machine learning techniques that we will use. We then apply our methods to various model physical examples, and study the ability of machine learning techniques to learn and quantify the direction of time's arrow.

Thermodynamics and the arrow of time

When small systems undergo thermodynamic processes, fluctuations are non-negligible and the second law is expressed in terms of averages. Therefore, the Clausius inequality that relates the work W that is performed on a system to the net change in its free energy ΔF takes the form

$$\langle W \rangle \geq \Delta F, \tag{1}$$

where the angular bracket denotes an average over many repetitions of the process. These non-equilibrium fluctuations satisfy strong constraints that enable us to rewrite such inequalities in terms of stronger equalities[6,7,30–32], and to quantify the direction of time's arrow as a problem in statistical inference[2,3,6,33,34]. To frame this problem, let us first specify the class of processes that we will study, and introduce the notation.

Consider a system that is in contact with a thermal reservoir at temperature β^{-1}. The system's Hamiltonian $\mathcal{H}_\lambda(\mathbf{x})$ depends on both the system's microstate $\mathbf{x}$ and a parameter λ. An external agent performs work by manipulating this parameter. Now imagine that the system begins in equilibrium with the reservoir, and that the agent then varies the parameter according to a schedule $\lambda_F(t)$ from $\lambda_F(0)=A$ to $\lambda_F(\tau)=B$. We refer to this as the 'forward process'. The trajectory that describes the system's evolution can be pictured as a movie, and is denoted by $\{\mathbf{x}_{A\to B}(t)\}$, where the time interval $0\leq t\leq\tau$ is implied. We refer to this as the 'forward trajectory' (Fig. 1a). We also imagine the 'reverse process', in which the system starts in an equilibrium state at $\lambda=B$ and the agent varies the parameter from B to A according to $\lambda_R(t)=\lambda_F(\tau-t)$. The trajectory (movie) for this process is denoted by $\{\mathbf{x}_{B\to A}(t)\}$. Finally, consider the time reversal of this trajectory, $\overline{\mathbf{x}}_{B\to A}(t)=\mathbf{x}^*_{B\to A}(\tau-t)$, where the asterisk implies

[1]Department of Physics, University of Maryland, College Park, MD, USA. [2]Joint Quantum Institute, NIST/University of Maryland, College Park, MD, USA. [3]Department of Electrical and Computer Engineering and The Institute for Research in Electronics and Applied Physics, University of Maryland, College Park, MD, USA. [4]Department of Chemistry and Biochemistry, and Institute for Physical Science and Technology, University of Maryland, College Park, MD, USA. ✉e-mail: seif@umd.edu

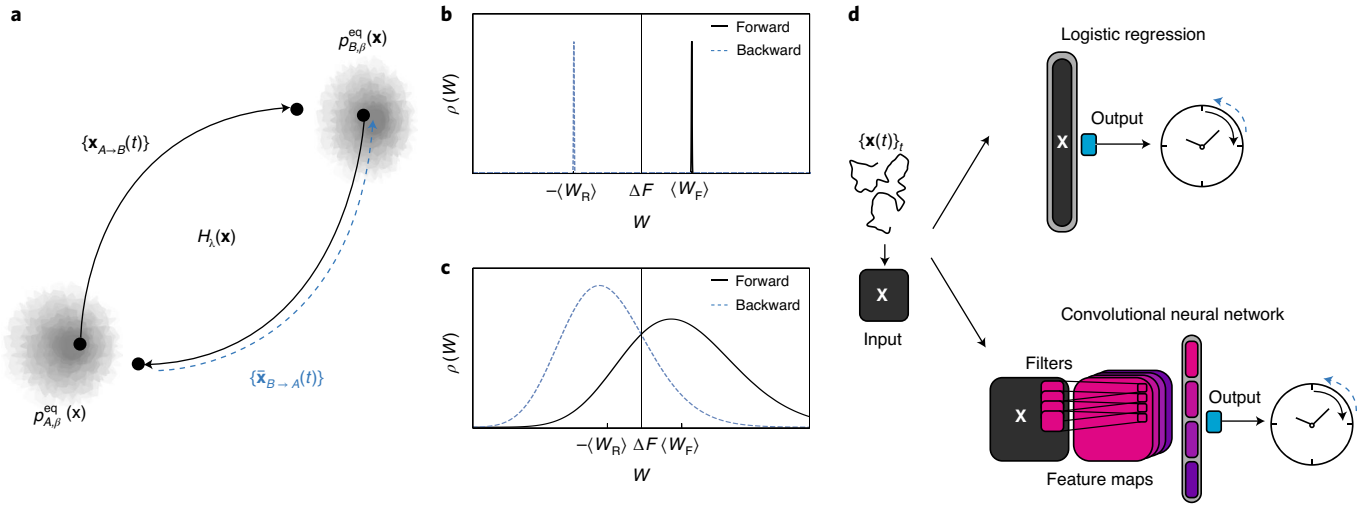

Fig. 1 | Non-equilibrium physics, time's arrow and machine learning. a, The system evolves under a Hamiltonian $\mathcal{H}_\lambda$ that depends on a parameter λ. The solid black trajectories show the system's evolution during the forward ($A \to B$) and reverse ($B \to A$) processes. In the forward (reverse) process, the system starts in equilibrium $p^{eq}_{A(B),\beta}$ and λ is varied from $A(B)$ to $B(A)$, respectively. The dashed blue trajectory $\{\tilde{\mathbf{x}}_{B\to A}(t)\}$ is the time reversal of the system's evolution during the reverse process. **b**, The work distribution $\rho(W)$ that corresponds to the forward W_F and the backward $-W_R$ trajectories. The change in the free energy during the forward process is denoted by ΔF. For macroscopic irreversible phenomena, fluctuations are negligible, $W_F > \Delta F > -W_R$, and the distinction between the forward and backward trajectories are clear. **c**, The work distribution for a microscopic system is similar to that of **b**, but fluctuations are more pronounced than in **b** and the distinction between the two distributions is less clear. **d**, A trajectory is represented by a matrix **X**. This matrix is the input to a neural network, which detects the direction of the time's arrow. The top shows a logistic regression network in which the input is flattened and reshaped into a vector, and the output is calculated by applying a non-linear function to a linear combination of the input coordinates. The bottom shows a convolutional neural network in which filters are first convolved with the input, which creates feature maps that encode abstract information about the local structure of the data. These feature maps are then reshaped and processed through a fully connected layer. The output of the network is used to decide the direction of time's arrow.

negation of momentum coordinates. This time-reversed trajectory corresponds to a movie of the reverse process that is played backward in time, and is referred to as the 'backward trajectory' (Fig. 1a).

Guessing the direction of time's arrow can be cast as a game in which a player is shown either a forward or a backward trajectory. In either case, the player therefore 'sees' only the parameter being varied from A to B. The player must then guess which process, forward or reverse, was in fact used to generate the trajectory[35]. The player's score, or accuracy, is the ratio of correct predictions to the total number of samples.

To optimize the likelihood of guessing correctly, it suffices for the player to know the sign of the quantity $W-\Delta F$, where W is the work that is performed on the system and $\Delta F = F_B - F_A$ is the free energy difference between the system's initial and final states, as depicted in the movie. Specifically, let $P(\mathrm{F}|\{\mathbf{x}(t)\})$ denote the likelihood that a given trajectory $\{\mathbf{x}(t)\}$ is obtained by performing the forward process, and let $P(\mathrm{R}|\{\mathbf{x}(t)\})$ denote the likelihood that the trajectory is the time reversal of a realization of the reverse process. Note that $P(\mathrm{F}|\{\mathbf{x}(t)\}) + P(\mathrm{R}|\{\mathbf{x}(t)\}) = 1$. In addition, assume that the game is unbiased; for example, the choice of performing the forward or reverse process in the first place was decided by flipping a fair coin. Then the likelihood that the trajectory was generated during the forward process is given by[3,33,34]

$$P(\mathrm{F}|\{\mathbf{x}(t)\}) = \frac{1}{1+e^{-\beta(W-\Delta F)}}, \tag{2}$$

which is greater than (less than) 50% when $W-\Delta F$ is positive (negative). Here, the work that is performed by the external agent is

$$W = \int_0^\tau \mathrm{d}t \dot{\lambda} \frac{\partial \mathcal{H}_\lambda(\mathbf{x})}{\partial \lambda}, \tag{3}$$

and the change in free energy is given by

$$\Delta F = -\frac{1}{\beta}\log\left(\frac{Z_{B,\beta}}{Z_{A,\beta}}\right), \tag{4}$$

where

$$Z_{\lambda,\beta} = \int \mathrm{d}\mathbf{x} \exp[-\beta \mathcal{H}_\lambda(\mathbf{x})] \tag{5}$$

is the partition function. In macroscopic systems, the values of the work performed on the system that correspond to forward trajectories, W_F, and backward trajectories, $-W_R$, are peaked sharply around their mean values (Fig. 1b), and the sign of $W-\Delta F$ is a reliable indicator of the direction of time's arrow. (Here, W_R is the work performed during a given realization of the reverse process, and therefore for the corresponding backward trajectory the work value is $-W_R$.) However, for microscopic systems these distributions can overlap substantially (Fig. 1c). Equation (2) shows that the player optimizes the chance of success simply by guessing 'forward' whenever $W > \Delta F$, and 'reverse' otherwise, without accounting for any further details of the trajectory. Note that if $|W-\Delta F| \gg \beta^{-1}$, then determining the arrow of time is easy, but when $|W-\Delta F| \lesssim \beta^{-1}$, the problem becomes more difficult and, in effect, time's arrow is blurred.

Neural networks

To train a computer program to infer the direction of time's arrow from a movie of the system's trajectory, we first simulate a number of trajectories from the forward and the reverse processes, and we 'time-reverse' the latter so that each trajectory is ordered chronologically with λ varying from A to B (see Methods for simulation details). We attach a label $y=0$ (reverse) or $y=1$ (forward) to indicate

which process was used to generate that trajectory. We then provide the machine with this collection of labelled trajectories, which serves as the training data. A priori, any one of the trajectories could have been generated from either the forward or the reverse process, and the training stage now consists of using a neural network classifier to construct a model of the function $P(\mathrm{F}|\{\mathbf{x}(t)\})$, which gives the likelihood that the trajectory was generated by the forward process. Although this function is known analytically (equation (2)), the machine is not provided with this information. We now sketch how the training is accomplished.

As each numerically generated trajectory consists of a discretized time series of microstates, we represent the trajectory as a matrix $\mathbf{X}$ whose rows correspond to different times and whose columns correspond to phase-space coordinates. The training stage amounts to designing a function that maps any such matrix $\mathbf{X}$ onto a real number p between 0 and 1, whose value is the machine's best estimate of the likelihood that the trajectory was generated by the forward process.

In this work, we consider two types of classifiers: logistic regression, which can be thought of as a single-layer neural network, and convolutional neural network (CNN). The input to logistic regression is a vectorized trajectory $\mathbf{a}=\mathrm{vec}(\mathbf{X})$, and the output is $p=g(\boldsymbol{\Omega}^{\intercal}\mathbf{a}+b)$, where $\boldsymbol{\Omega}$ is a vector of weights, b is the bias, and $g(z)=1/(1+\exp(-z))$ is the logistic sigmoid function (Fig. 1d, top). Compared to the logistic regression, the CNN can compute functions that are more complicated[36]. The input to our CNN is a trajectory matrix $\mathbf{X}$, and the output is again a value p. The CNN has convolutional layers that extract useful information by taking advantage of the temporal and spatial structure of the data (Fig. 1d, bottom). To train the classifier, we determine optimal values of parameters (such as the weights and biases in logistic regression) by minimizing the cross-entropy with a gradient-based optimization algorithm[36] (see Methods for details of training neural networks and the CNN architecture).

Case studies

We apply the neural network machinery to detect the direction of the time's arrow, and compare the output of the network with the theoretical optimal result of equation (2). We first consider a single Brownian particle in a moving potential, and then move on to the more complicated problem of a spin chain with nearest-neighbour coupling in a magnetic field and discuss how controlling the field and the coupling affects the results. The networks learn not only to guess the direction of the time's arrow but also to closely reproduce the likelihood function. The sensitivity of the results to the choice of the activation function g that is applied to the output of the networks is considered (see Supplementary Information and Supplementary Fig. 3 for discussion).

Brownian particle in a moving potential. An overdamped Brownian particle at temperature β^{-1} in a harmonic potential (Fig. 2a) evolves according to

$$\dot{x}=-\frac{k}{\gamma}(x-\lambda)+\xi(t), \tag{6}$$

where k denotes the strength of the potential, λ is the position of the centre of the potential and γ is the damping rate. The noise term $\xi(t)$ satisfies $\langle\xi(t)\xi(t')\rangle=2(\beta\gamma)^{-1}\delta(t-t')$. In the forward protocol, the value of λ is changed from A to B at a fixed rate $\dot{\lambda}=u$. Hence, the reverse protocol changes λ from B to A with $\dot{\lambda}=-u$.

After generating samples of the forward and backward trajectories by using equation (6) (Fig. 2b,c), we train a classifier to predict the label for a given trajectory, as described earlier. In this example it is easy to detect the direction of the time's arrow, as the work distributions have a modest overlap (Fig. 2d). We compare the accuracy and the output of a logistic regression classifier with the theoretical likelihood that is obtained from equation (2) (Fig. 2e). The remarkable agreement with the theory can be understood by noting that the work W, calculated by numerically integrating $\dot{W}=-ku(x-ut)$ for a given trajectory, is a linear function of the elements of the trajectory matrix $\mathbf{X}$. Therefore, logistic regression is well equipped to calculate this quantity and reproduce the likelihood function (see Supplementary Information for detailed analysis).

Spin chain in a magnetic field. Now let us consider a more complicated, many-particle system and a non-linear work protocol, namely a spin chain in a magnetic field and in contact with a thermal reservoir at temperature β^{-1}. This spin chain is described by the Hamiltonian

$$H=J(t)\sum_i\sigma_i\sigma_{i+1}-B(t)\sum_i\sigma_i, \tag{7}$$

where $\sigma_i\in\{-1,+1\}$ is the spin variable at site i, $J(t)$ is the nearest-neighbour coupling strength and $B(t)$ is the magnetic field. The dynamics of this system are modelled as a Markov process (see Methods). The Hamiltonian aligns the spin in preferred energy configurations, whereas thermal fluctuations cause the spins to flip randomly. We consider two scenarios. First, the coupling is assumed to be constant and the magnetic field is varied in time (Fig. 3a). Next, the magnetic field is constant and the coupling is varied (Fig. 4a). We refer to the former as the $\mathcal{B}$ protocol and the latter as the $\mathcal{J}$ protocol.

In the forward process of the $\mathcal{B}$ protocol, $J(t)$ is constant over time, and $B(t)=B_0\cos(\pi t/\tau)$ changes from $B_0>0$ at $t=0$ to $-B_0$ at $t=\tau$ (Fig. 3b). At low temperatures and in large magnetic fields, the spins are aligned with the direction of the field. As the temperature is increased, fluctuations become more prominent (Fig. 3c,d). These fluctuations increase the overlap in work distributions, which blurs the direction of time's arrow (Fig. 3e). We train a single classifier by using samples of forward and backward trajectories for three different temperatures; these samples (trajectories) are generated by the Metropolis algorithm. This training scheme, known as multitask learning, improves the performance and generalizability of the classifier[37]. We observe that the success of logistic regression in learning both the correct labels and in approximating the likelihood function persists (Fig. 3f). The reason, again, lies in the functional form of W, which is evaluated by numerically integrating $\dot{W}=-\dot{B}(t)\sum_i\sigma_i$, and therefore is proportional to a weighted sum of the elements of the input trajectory. As $\Delta F=0$ in this protocol, logistic regression is a perfect model of the likelihood function for all of the temperatures (see Supplementary Information for details).

The $\mathcal{J}$ protocol is more complicated and has a ferromagnetic–antiferromagnetic transition. In this protocol, $B(t)$ is constant in time and $J(t)=J_0\cos(\pi t/\tau)$ is varied non-linearly from $J_0>0$ at $t=0$ to $-J_0$ at $t=\tau$ in the forward process. The logistic regression classifier does not perform well in this case (Supplementary Fig. 1) as $\dot{W}=\dot{J}(t)\sum_i\sigma_i\sigma_{i+1}$ is no longer linearly related to the input. However, by using a CNN with periodic boundary condition, we are able to recover the optimal accuracy and obtain results (Fig. 4) that are similar to those for the $\mathcal{B}$ protocol. The convolution layer in a CNN has filters that can capture the two-body nearest-neighbour correlations that are required to calculate the work (see Supplementary Information). Note that in this process $\Delta F\neq 0$, which adds to the complexity of the problem.

Interpretation and extensions

We now use three approaches to investigate trained networks and to develop insight into what they have learned.

First, we use inceptionism techniques[38,39] to learn the network's ideal representative of forward and backward trajectories.

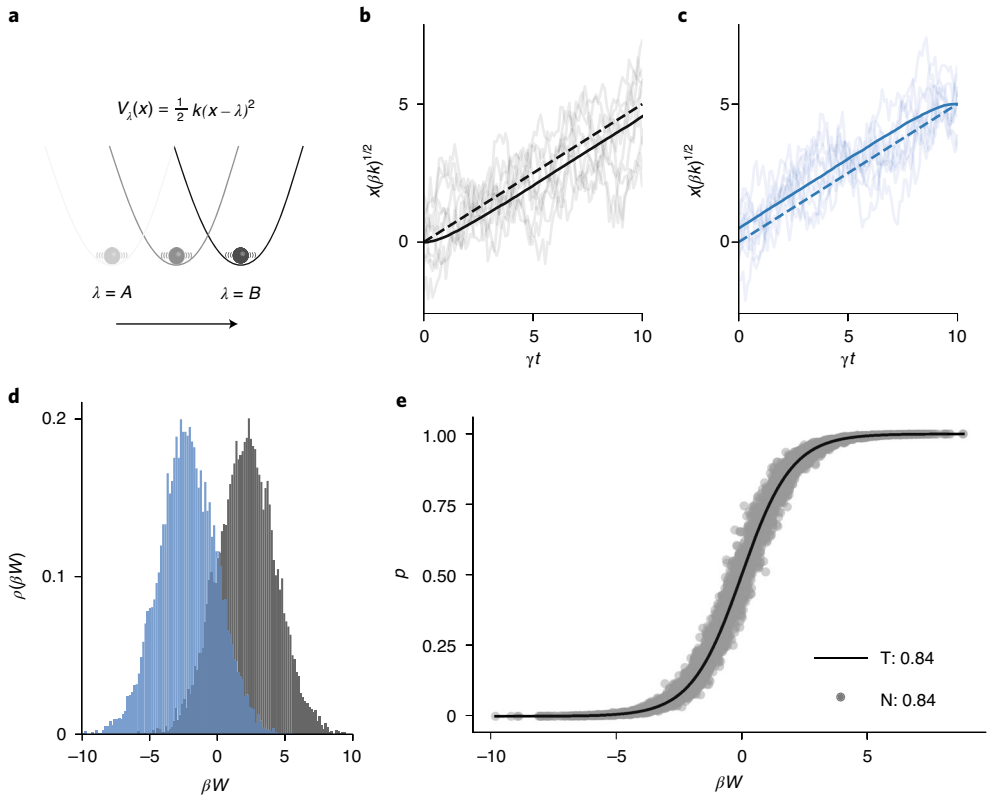

Fig. 2 | Brownian particle in a moving potential. a, An overdamped Brownian particle with damping rate γ at temperature β^{-1} is in a harmonic potential $V_\lambda(x)$, with stiffness k. The position of the potential's centre, λ, is controlled externally and is moved from A to B in the forward process. **b**, Sample trajectories (grey) and the average trajectory (solid black line) in the forward protocol. Note that the average trajectory lags behind the centre of the potential (dashed line). In the simulations, γ, β and k are chosen to equal 1, and $\lambda = 0$ at $t = 0$ is varied to $\lambda = 5$ at $t = 10$. **c**, Sample backward trajectories (light blue) and their average (solid dark blue line). The average trajectory leads the potential's centre (dashed line). **d**, Work distribution for the forward (black) and the backward (blue) trajectories. Both are distributed normally and are symmetric around 0. **e**, The likelihood of the forward process for a set of test trajectories. The output of the neural network (N), p, over the test set (grey dots) resembles the theoretical likelihood (T), $P(\mathrm{F}|\mathbf{X})$ (black line). The values in the key denote the accuracy of the methods over 4,000 test trajectories. The classifier is trained on 12,000 samples.

Specifically, we use a Monte Carlo approach to transform a randomly selected input trajectory into one for which the networks that are trained for $\mathcal{B}$ and $\mathcal{J}$ protocols provide an output of 1 or 0, which correspond to forward and backward trajectories, respectively. This is in contrast to the previous section, in which we optimized for the weights and biases of the network. Among the simulated trajectories in the test set, we choose one with $p \cong 0.5$ (ref. [39]); this is a trajectory for which the network has difficulty assigning the direction of time's arrow. We propose random spin flips and accept those moves that cause the output to get closer to the desired value of 1 or 0. In addition, we demand that there be at most one spin flip per time step, to ensure that the network 'dreams' of trajectories that are consistent with our simulations. We find that the networks' ideas of the forward and backward trajectories show strong agreement with the true physical picture (Fig. 5a). We also present an alternative gradient-based dreaming algorithm (see Supplementary Information and Supplementary Fig. 4).

Second, to assign a physical interpretation to the networks' decision-making process, we project the trajectories onto a two-dimensional reduced phase space that corresponds to the collective coordinates $\{\tilde{x}^{(1)}(t)\} = \{\sum_i \sigma_i(t)\}$ and $\{\tilde{x}^{(2)}(t)\} = \{\sum_i \sigma_i(t)\sigma_{i+1}(t)\}$ (taking period boundary conditions), which represent magnetization and nearest-neighbour correlations, respectively. We also replace the value of $\tilde{x}^{(\ell)}(t)$ within each time window of 10 time steps by the sum of the values within that window (see Methods for details). By such coarse-graining in both phase space and time, we reduce the noise that is due to finite size effects and variations over samples. We use these coarse-grained trajectories to train logistic regression classifiers for both $\mathcal{B}$ and $\mathcal{J}$ protocols (see Supplementary Fig. 2 for the performance of these classifiers). Finally, we investigate the weights $\mathbf{\Omega}^{(\ell)}$ that the networks assign to the magnetization ($\ell = 1$) and the nearest-neighbour correlations ($\ell = 2$) of an input trajectory. For the $\mathcal{B}$ protocol (Fig. 5b, top), the network cares mostly about the magnetization, whereas when the $\mathcal{J}$ protocol is performed (Fig. 5b, bottom), the network bases its decision on the nearest-neighbour correlations. Moreover, the learned values of $\mathbf{\Omega}^{(\ell)}$ agree with our analytical results that reproduce the correct likelihood value (see Supplementary Information for details). These observations suggest that the network learns that the time derivative of the Hamiltonian, and by extension the work (equation (3)), is an important feature in guessing the direction of time's arrow. We note that when the process is highly irreversible, the distributions of the forward and reverse work are well separated. In this case, the network determines easily the arrow of time, but does not learn about the importance of work and bases its decision on other visible differences in the trajectories (see Supplementary Information and Supplementary Fig. 5).

Last, by using our knowledge about the structure of the problem, we design a neural network that can learn to guess accurately the direction of time's arrow for trajectories that are generated by using multiple protocols, when the identity of the protocol is not specified. Specifically, we use a mixture of experts (MoE), with an output

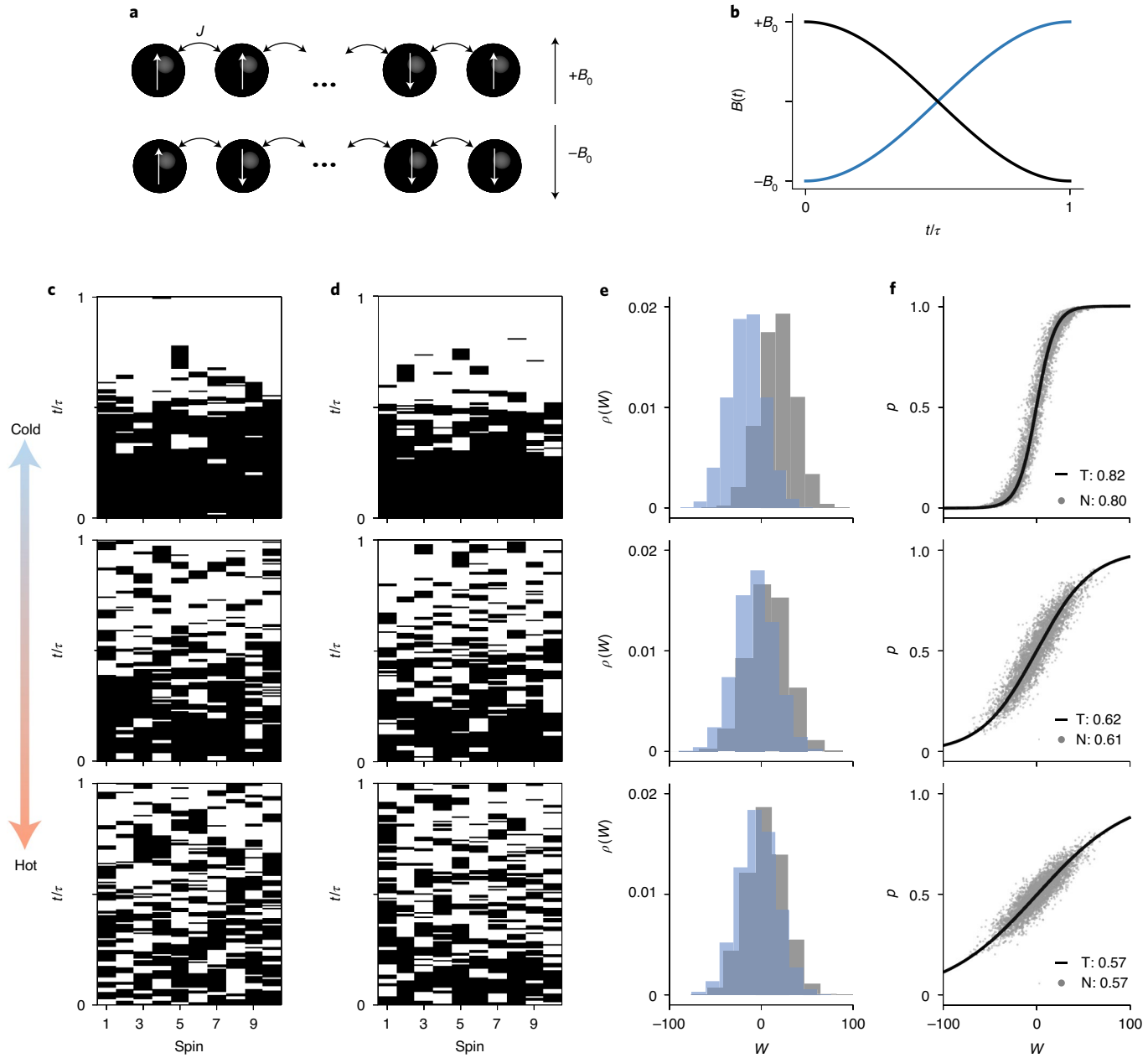

Fig. 3 | Spin chain in a time-dependent magnetic field. a, A chain of ten spins with periodic boundary condition is placed in a magnetic field. The strength of coupling between nearest neighbours $J(t)=-1$ is fixed, and all of the quantities are in the units of $|J|$. The forward process starts with spins in equilibrium at temperature β^{-1} with $B(0)=+B_0>0$ at time $t=0$ and ends at a non-equilibrium state with $B(\tau)=-B_0$ at at time $t=\tau$. **b**, The forward (black) and the reverse (blue) protocols $B(t)$. In our examples, $B_0=20$ and $\tau=500$. **c**, Sample forward trajectories that are encoded in a matrix, in which the black and white pixels (±1 entries) denote spins that point up and down, respectively. The rows and columns correspond to time steps and spin positions, respectively. **d**, Sample backward trajectories that are obtained from time reversal of the reverse process. **e**, The distribution of work $\rho(W)$ for the forward (black) and backward (blue) trajectories. **f**, The theoretical likelihood function (black line) and the output of the neural network p over the test set (grey dots) for various temperatures. The values in the keys denote the accuracy of the theory (T) and neural network (N) over 4,000 test trajectories for each temperature. In this example, a single classifier is trained simultaneously with sample trajectory data with three different temperatures (12,000 samples for each temperature). The temperatures that correspond to the rows in parts **c**–**f** are $\beta^{-1}=10$, 30 and 50 from top to bottom. As the temperature increases, the distinction between the forward and backward trajectories is blurred.

that is the weighted sum of expert networks[40]. When the protocol is not specified, the net forward likelihood is

$$P(\mathrm{F}|\mathbf{X}) = P(\mathrm{F}|\mathbf{X},\mathcal{B})P(\mathcal{B}|\mathbf{X}) + P(\mathrm{F}|\mathbf{X},\mathcal{J})P(\mathcal{J}|\mathbf{X}). \quad (8)$$

The quantities $P(\mathrm{F}|\mathbf{X},\mathcal{B})$ and $P(\mathrm{F}|\mathbf{X},\mathcal{J})$ are modelled by using neural networks that are similar to those considered previously for $\mathcal{B}$ and $\mathcal{J}$ protocols, respectively. These networks are referred to as experts. In addition, we use a CNN to model $P(\mathcal{B}|\mathbf{X}) = 1 - P(\mathcal{J}|\mathbf{X})$. This CNN, which is called the gating network, learns the protocol from trajectories. Therefore, we obtain a larger three-headed network by combining the output of the three neural networks, as in equation (8) (Fig. 6a). For the training, we use the pre-trained expert networks for the $\mathcal{B}$ and $\mathcal{J}$ protocols, and optimize the cross-entropy cost function over

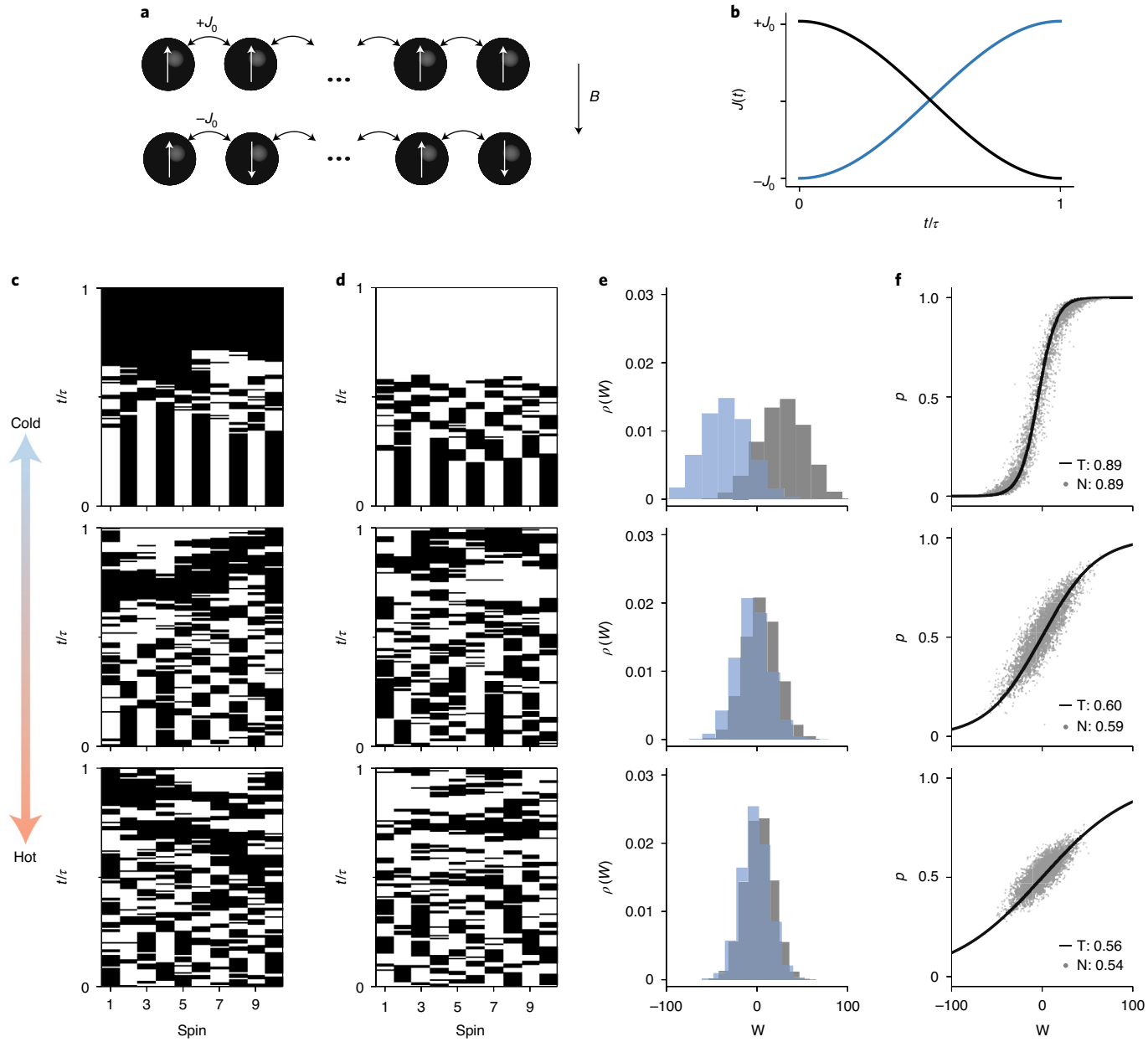

Fig. 4 | Spin chain with a time-dependent coupling. a, A chain of ten spins with periodic boundary condition is placed in a constant magnetic field $B(t) = -1$, and all of the quantities are in the units of $|B|$. The time-dependent coupling between nearest neighbours is $J(t)$. The forward process starts with spins in equilibrium at temperature β^{-1} with $J(0) = +J_0 > 0$ and ends at a non-equilibrium state with $J(\tau) = -J_0$. **b**, The forward (black) and the reverse (blue) protocols $J(t)$. In our examples, $J_0 = 20$ and $\tau = 500$. **c**, Sample forward trajectories that are encoded in a matrix, in which the black and white pixels (± 1 entries) denote spins that point up and down, respectively. The rows and columns correspond to time steps and spin positions, respectively. **d**, Sample backward trajectories that are obtained from time reversal of the reverse process. **e**, The distribution of work $\rho(W)$ for the forward (black) and backward (blue) trajectories. **f**, The theoretical likelihood function (black line) and the output of the neural network p over the test set (grey dots) for various temperatures. The values in the keys denote the accuracy of the theory (T) and neural network (N) over 4,000 test trajectories for each temperature. In this example, a single classifier is trained simultaneously with sample trajectory data with three different temperatures (12,000 samples for each temperature). The temperatures that correspond to the rows in parts **c**–**f** are $\beta^{-1} = 10$, 30 and 50 from top to bottom. As the temperature increases, the distinction between the forward and backward trajectories is blurred.

sample trajectories from both protocols. We observe that the performance of this network is similar to that of the individual networks, as the gating network learns to identify the protocol of input trajectories accurately (Fig. 6b). Note that the predictions of the gating network are more accurate at lower temperatures. This makes sense as the distribution of the initial states in the two protocols is distinguishable at low temperatures, but becomes less so as the temperature is increased.

Although in this work we rediscovered elements of thermodynamics that had been developed in recent decades, we are interested ultimately in answering open questions in thermodynamics. The physics of systems that are out-of-equilibrium is an area of interest, with unsolved questions that could be answered by studying the dynamics of the systems with machine learning algorithms. For instance, identifying the thermodynamic principle that determines

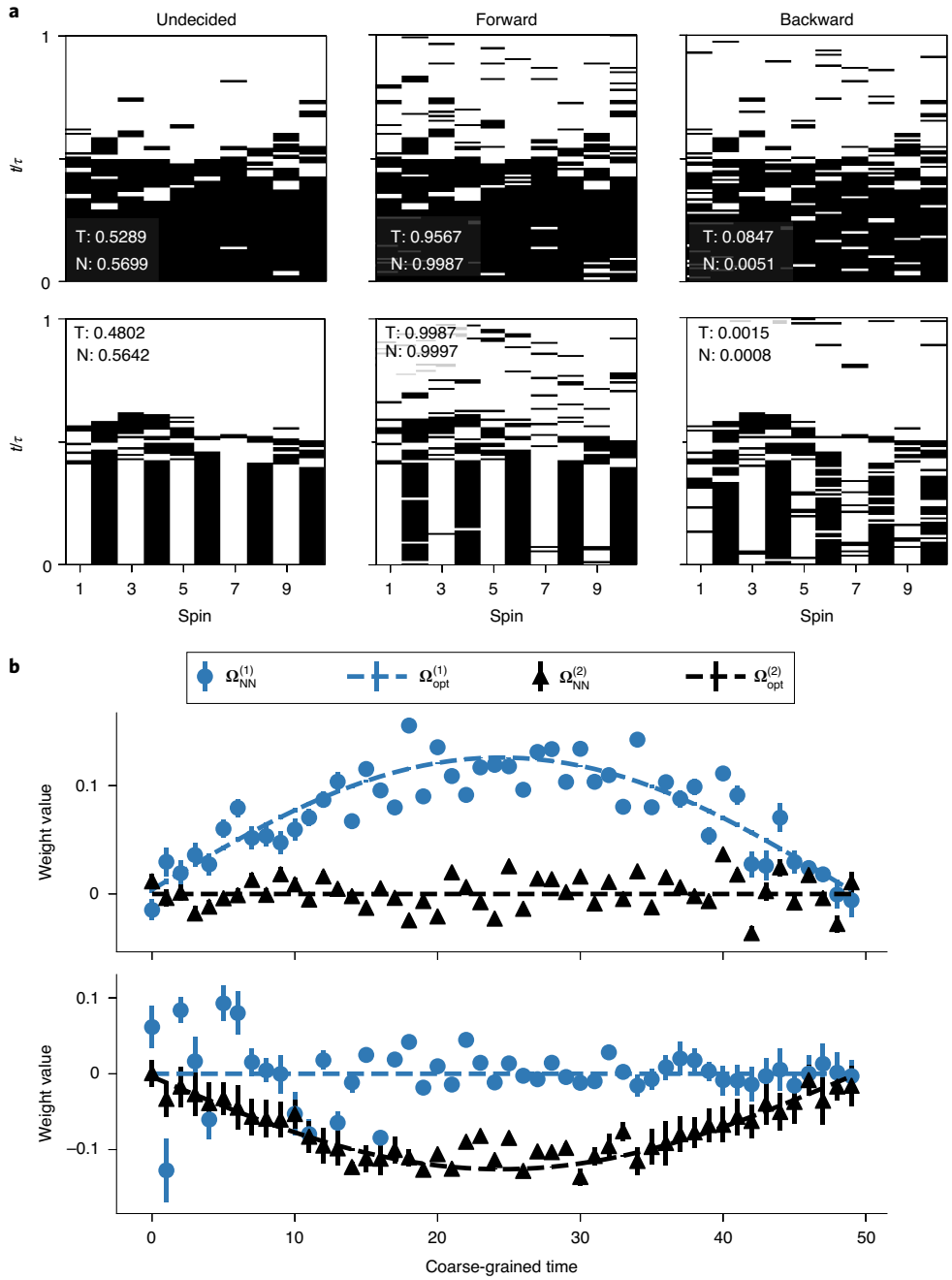

Fig. 5 | Interpreting the neural network's inner mechanism. a, Starting with a random trajectory (left), we ask the trained network to 'dream' of its idea of the forward (middle) and backward (right) trajectories. Here, t is time and τ is the duration of the process. The top row corresponds to the $\mathcal{B}$ protocol, with parameters identical to those in Fig. 3, and the bottom row corresponds to the $\mathcal{J}$ protocol with similar parameters but with the values of magnetic field $B(t)$ and coupling strength $J(t)$ interchanged. The black and white pixels denote spins that point up and down, respectively. The values in the insets indicate the forward likelihood $P(\mathrm{F}|\mathbf{X})$ that is obtained from the theory (T) by using equation (2) and from the output of the neural network (N). **b**, Weights of the networks trained by using coarse-grained trajectories that are associated with the magnetization $\mathbf{\Omega}^{(1)}$ and the nearest-neighbour correlations $\mathbf{\Omega}^{(2)}$. The network for the $\mathcal{B}$ protocol (top) is trained simultaneously at temperatures $\beta^{-1}=10$, 30 and 50, and the network for the $\mathcal{J}$ protocol (bottom) is trained at $\beta^{-1}=10$. The x axis represents coarse-grained time in intervals of 10 steps for trajectories in part **a**. The error bars are the standard deviation over 10 trained networks with random weight initialization. The network bases its decision on the magnetization in the $\mathcal{B}$ protocol and on the nearest-neighbour correlations in the $\mathcal{J}$ protocol. If the values of $\mathbf{\Omega}^{(\ell)}_{\mathrm{NN}}$s of the trained networks (markers) match the optimal weights $\mathbf{\Omega}^{(\ell)}_{\mathrm{opt}}$ (dashed line), the output of the network agrees with the exact likelihood (equation (2)).

the steady state of a system[41], an outstanding problem in statistical physics, and deciding when a non-equilibrium system has an effective equilibrium description[42] are two examples in which the analysis of time-series data by using machine learning could shed light on the inner mechanism of the problem.

Moreover, machine learning researchers have shown that machine learning techniques can be used to detect the playback direction of real-world videos [43,44]. These studies are concerned with videos of macroscopic objects that are in principle irreversible, and for which the arrow of time has a clear direction. In such scenarios,

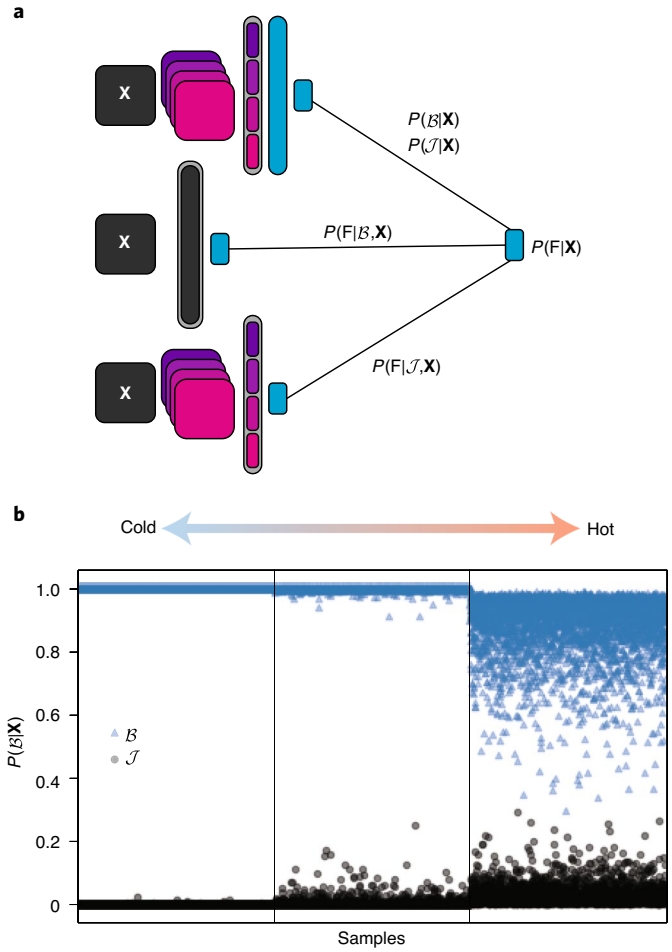

Fig. 6 | Mixture of experts (MoE). a, The MoE network models the forward likelihood $P(\mathrm{F}|\mathbf{X})$ for an input trajectory $\mathbf{X}$. It consists of a gating CNN that predicts the protocol $P(\mathcal{B}(\mathcal{J})|\mathbf{X})$, and two networks that predict the forward likelihood of a trajectory given the protocol $P(\mathrm{F}|\mathcal{B}(\mathcal{J}),\mathbf{X})$. **b**, The output of the gating network, which models $P(\mathcal{B}|\mathbf{X})$, is shown for different sample trajectories of the $\mathcal{B}$ (blue triangles) and $\mathcal{J}$ (grey dots) protocols. The trajectories for the $\mathcal{B}$ protocol have the same parameters as in Fig. 3, with three temperatures $\beta^{-1}=10$, 30 and 50 and quantities in the units of $|J|$. The parameters in the $\mathcal{J}$ protocol are similar, but the values of magnetic field $B(t)$ and coupling strength $J(t)$ are interchanged. The horizontal axis shows different samples in three temperature regions that are separated by vertical lines. It is more difficult to predict the protocol at higher temperatures.

many indicators can reveal the true playback direction, and therefore it is difficult to quantify the optimal performance. However, in the physical examples, the optimal attainable accuracy of the classifier is dictated by the laws of physics. Therefore, problems with a large number of phase-space coordinates and with complicated dynamics, such as the $\mathcal{J}$ protocol for the two-dimensional Ising model, can serve as a standardized benchmark for video classification algorithms.

Data availablity

All data that support the plots in this paper and other findings of this study are available from the corresponding author upon reasonable request.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41567-020-1018-2.

Received: 25 September 2019; Accepted: 22 July 2020;
Published online: 21 September 2020

References

1. Eddington, A. S. *The Nature of the Physical World* (Macmillan, 1928).
2. Feng, E. H. & Crooks, G. E. Length of time's arrow. *Phys. Rev. Lett.* **101**, 090602 (2008).
3. Jarzynski, C. Equalities and inequalities: irreversibility and the second law of thermodynamics at the nanoscale. *Annu. Rev. Condens. Matter Phys.* **2**, 329–351 (2011).
4. Roldán, É., Neri, I., Dörpinghaus, M., Meyr, H. & Jülicher, F. Decision making in the arrow of time. *Phys. Rev. Lett.* **115**, 250602 (2015).
5. Hofmann, A. et al. Heat dissipation and fluctuations in a driven quantum dot. *Phys. Status Solidi B* **254**, 1600546 (2017).
6. Crooks, G. E. Nonequilibrium measurements of free energy differences for microscopically reversible Markovian systems. *J. Stat. Phys.* **90**, 1481–1487 (1998).
7. Crooks, G. E. Entropy production fluctuation theorem and the nonequilibrium work relation for free energy differences. *Phys. Rev. E* **60**, 2721–2726 (1999).
8. Seifert, U. Stochastic thermodynamics, fluctuation theorems and molecular machines. *Rep. Prog. Phys.* **75**, 126001 (2012).
9. Carleo, G. et al. Machine learning and the physical sciences. *Rev. Mod. Phys.* **91**, 045002 (2019).
10. Senior, A. W. et al. Improved protein structure prediction using potentials from deep learning. *Nature* **577**, 706–710 (2020).
11. Torlai, G. & Melko, R. G. Learning thermodynamics with Boltzmann machines. *Phys. Rev. B* **94**, 165134 (2016).
12. Carrasquilla, J. & Melko, R. G. Machine learning phases of matter. *Nat. Phys.* **13**, 431–434 (2017).
13. van Nieuwenburg, E. P. L., Liu, Y.-H. & Huber, S. D. Learning phase transitions by confusion. *Nat. Phys.* **13**, 435–439 (2017).
14. Deng, D.-L., Li, X. & Das Sarma, S. Machine learning topological states. *Phys. Rev. B* **96**, 195145 (2017).
15. Wetzel, S. J. & Scherzer, M. Machine learning of explicit order parameters: from the Ising model to SU(2) lattice gauge theory. *Phys. Rev. B* **96**, 184410 (2017).
16. Wetzel, S. J. Unsupervised learning of phase transitions: from principal component analysis to variational autoencoders. *Phys. Rev. E* **96**, 022140 (2017).
17. Ch'ng, K., Carrasquilla, J., Melko, R. G. & Khatami, E. Machine learning phases of strongly correlated fermions. *Phys. Rev. X* **7**, 031038 (2017).
18. Ch'ng, K., Vazquez, N. & Khatami, E. Unsupervised machine learning account of magnetic transitions in the Hubbard model. *Phys. Rev. E* **97**, 013306 (2018).
19. Liu, Y.-H. & van Nieuwenburg, E. P. L. Discriminative cooperative networks for detecting phase transitions. *Phys. Rev. Lett.* **120**, 176401 (2018).
20. Schindler, F., Regnault, N. & Neupert, T. Probing many-body localization with neural networks. *Phys. Rev. B* **95**, 245134 (2017).
21. Arsenault, L.-F., Lopez-Bezanilla, A., von Lilienfeld, O. A. & Millis, A. J. Machine learning for many-body physics: the case of the Anderson impurity model. *Phys. Rev. B* **90**, 155136 (2014).
22. Beach, M. J. S., Golubeva, A. & Melko, R. G. Machine learning vortices at the Kosterlitz–Thouless transition. *Phys. Rev. B* **97**, 045207 (2018).
23. van Nieuwenburg, E., Bairey, E. & Refael, G. Learning phase transitions from dynamics. *Phys. Rev. B* **98**, 060301 (2018).
24. Ponte, P. & Melko, R. G. Kernel methods for interpretable machine learning of order parameters. *Phys. Rev. B* **96**, 205146 (2017).
25. Schmidt, M. & Lipson, H. Distilling free-form natural laws from experimental data. *Science* **324**, 81–85 (2009).
26. Rudy, S. H., Brunton, S. L., Proctor, J. L. & Kutz, J. N. Data-driven discovery of partial differential equations. *Sci. Adv.* **3**, e1602614 (2017).
27. Iten, R., Metger, T., Wilming, H., del Rio, L. & Renner, R. Discovering physical concepts with neural networks. *Phys. Rev. Lett.* **124**, 010508 (2020).
28. Berkson, J. Application of the logistic function to bio-assay. *J. Am. Stat. Assoc.* **39**, 357–365 (1944).
29. Day, N. E. & Kerridge, D. F. A general maximum likelihood discriminant. *Biometrics* **23**, 313–323 (1967).
30. Jarzynski, C. Nonequilibrium equality for free energy differences. *Phys. Rev. Lett.* **78**, 2690–2693 (1997).
31. Jarzynski, C. Equilibrium free-energy differences from nonequilibrium measurements: a master-equation approach. *Phys. Rev. E* **56**, 5018–5035 (1997).

32. Hummer, G. & Szabo, A. Free energy reconstruction from nonequilibrium single-molecule pulling experiments. *Proc. Natl Acad. Sci. USA* **98**, 3658–3661 (2001).
33. Shirts, M. R., Bair, E., Hooker, G. & Pande, V. S. Equilibrium free energies from nonequilibrium measurements using maximum-likelihood methods. *Phys. Rev. Lett.* **91**, 140601 (2003).
34. Maragakis, P., Ritort, F., Bustamante, C., Karplus, M. & Crooks, G. E. Bayesian estimates of free energies from nonequilibrium work data in the presence of instrument noise. *J. Chem. Phys.* **129**, 024102 (2008).
35. Jarzynski, C. Rare events and the convergence of exponentially averaged work values. *Phys. Rev. E* **73**, 046105 (2006).
36. Goodfellow, I., Bengio, Y. & Courville, A. *Deep Learning* (MIT Press, 2016).
37. Caruana, R. Multitask learning. *Mach. Learn.* **28**, 41–75 (1997).
38. Mordvintsev, A., Olah, C. & Tyka, M. Inceptionism: going deeper into neural networks. *Google AI Blog* https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html (2015).
39. Schindler, F., Regnault, N. & Neupert, T. Probing many-body localization with neural networks. *Phys. Rev. B* **95**, 245134 (2017).
40. Nowlan, S. J. & Hinton, G. E. in *Advances in Neural Information Processing Systems* 1st edn, Vol. 3 (eds Lippmann, R. P. et al.) 774–780 (Morgan Kaufmann, 1991).
41. Wächtler, C. W., Strasberg, P., Klapp, S. H. L., Schaller, G. & Jarzynski, C. Stochastic thermodynamics of self-oscillations: the electron shuttle. *New J. Phys.* **21**, 073009 (2019).
42. Young, J. T., Gorshkov, A. V., Foss-Feig, M. & Maghrebi, M. F. Nonequilibrium fixed points of coupled Ising models. *Phys. Rev. X* **10**, 011039 (2020).
43. Pickup, L. C. et al. Seeing the arrow of time. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2035–2042 (IEEE Computer Society, 2014).
44. Wei, D., Lim, J. J., Zisserman, A. & Freeman, W. T. Learning and using the arrow of time. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 8052–8060 (IEEE Computer Society, 2018).

Methods

Training neural networks. To train the classifier, we determine optimal values of parameters (such as the weights and biases in logistic regression) by minimizing the cross-entropy[36]

$$C = -\frac{1}{N_{\text{samp}}}\sum_m \left[y_m \log(p_m) + (1-y_m)\log(1-p_m)\right]. \tag{9}$$

The sum is carried over the N_{samp} training samples, $y_m \in \{0,1\}$ is the label that is attached to the mth trajectory to indicate which process was used to generate the trajectory, and p_m is the output of the network for that trajectory.

Throughout this work, we always split a given data set into three parts. We use 60%, 20% and 20% of the data for training, validating and testing the model, respectively. We use a data set with a total of 20,000 samples for the Brownian particle. For the spin chain examples ($\mathcal{B}$ and $\mathcal{J}$ protocols), we use 20,000 samples for each temperature. The samples are then split into three sets and are used to train, validate and test the models. The validation set is used to tune the architecture and hyperparameters of the model, whereas the test data are used for an unbiased evaluation of the accuracy of the final model. For the training, we use an Adam optimizer with parameters that were suggested in the original paper[45]. We assess the performance of the network by testing it over a balanced set of trajectories, that is, half forward and half backward. If $p_m \geq 0.5$, then the algorithm guesses that the trajectory was generated from the forward process, otherwise it guesses the reverse process. We consider the accuracy as a figure of merit, that is, the ratio of correct guesses to the total number of samples.

To reduce overfitting, it is helpful to include a regularization term to reduce the difference between the training error and the test error. We consider L_2 regularization $\alpha\sum_\ell \Omega_\ell^2$, that is, adding the sum of squares of all of the weights in the network to the cost function. The parameter α is a hyperparameter of the model and is tuned by using the cross-validation data.

In addition, for training the CNNs in this work, we use the dropout technique to reduce overfitting. Dropout involves deactivating and ignoring certain neurons during the training phase. Specifically, a random fraction p_{drop} of neurons are deactivated at every training step[46].

We find that our results do not vary substantially with the choice of hyperparameters. We choose $p_{\text{drop}} = 0.25$ for the dropout rate of neurons of the convolutional layer in the $\mathcal{J}$ network, and $p_{\text{drop}} = 0.5$ for the gating network in the MoE. The L_2 regularization rates α are shown in Supplementary Table 1.

Convolutions. A convolution layer convolves the input with a number of filters, and then applies a non-linear function to the output of the filters. Each convolution operation with a kernel $\mathbf{\Omega}$ and bias b maps an input matrix $\mathbf{X}$ to another matrix $\mathbf{Z} = \mathbf{\Omega} * \mathbf{X}$ that is given by[36]

$$Z_{j,k} = \sum_{m,n} X_{j\times s+m, k\times s+n}\Omega_{m,n} + b \tag{10}$$

where s specifies the number of steps that the filter moves in each direction; s is called the stride of the convolution and is a hyperparameter that is tuned by using the cross-validation data. The output of the convolution layer is obtained by applying a non-linear function g element-wise to $\mathbf{Z}$. The convolution layers can be repeated many times and combined with pooling layers in which the dimension of the output is reduced through a procedure such as averaging. At the end, the output of the convolution layer is flattened to form a vector, and that vector is fed into a series of fully connected layers to produce the network's output[36].

The CNN that we consider has four 2×2 filters with a stride of 1 and with periodic boundary condition. We choose the rectifier $g(z) = \max(0, z)$ for the activation of these filters. The outputs of all of the filters are then combined to form a single vector. For the CNN that classifies the $\mathcal{J}$ protocol (Fig. 4), this vector is fed into a single neuron with sigmoid activation, whose values determine the direction of time's arrow. For the MoE gating network, this vector is fed into a fully connected layer with 50 hidden neurons and the rectifier activation, followed by the output neuron with the sigmoid activation.

Generating the data. To generate trajectories, we follow ref. [6] closely. We consider a discrete set of time steps $t \in \{0, 1, \ldots, \tau\}$. The value of the control parameter and the state of the system at each time step are denoted by λ_t and $\mathbf{x}_t$, respectively. In the forward process, the initial state of the system is drawn from equilibrium with $\lambda = \lambda_0$. The time evolution can be broken into two substeps:

(i) With the state of the system fixed, the control parameter is changed, as denoted by $\lambda_t \to \lambda_{t+1}$.
(ii) At fixed λ_{t+1}, the state of the system evolves, as denoted by $\mathbf{x}_t \to \mathbf{x}_{t+1}$.

Here, the second substep is generated by either a stochastic differential equation (for the Brownian particle) or a Metropolis algorithm (for spin examples). The total work that is performed in this process is

$$W = \sum_{t=0}^{\tau-1}\left[\mathcal{H}_{\lambda_{t+1}}(\mathbf{x}_t) - \mathcal{H}_{\lambda_t}(\mathbf{x}_t)\right] \tag{11}$$

To produce backward trajectories, the system is initialized in an equilibrium state with $\lambda = \lambda_\tau$. The dynamics begin with a change in the system state, followed by a change in λ. At the end, the history of the system state is reversed, and the calculated work is negated to obtain backward trajectories and their corresponding work values.

Coarse-grained features. To obtain the coarse-grained feature results (Fig. 5), we reduce the parameters of the neural network and simplify the task of learning by pre-calculating a set of features for the network. Specifically, for the two protocols that concern the spin chain in a magnetic field, the coarse-grained features are

$$\tilde{x}_s^{(1)} = \sum_{t=ms}^{m(s+1)-1}\sum_{i=1}^{n} X_{t,i}, \tag{12}$$

$$\tilde{x}_s^{(2)} = \sum_{t=ms}^{m(s+1)-1}\sum_{i=1}^{n} X_{t,i}X_{t,i+1}, \tag{13}$$

where m is an integer and s is the scaled time. We use this feature map as the input to a logistic regression classifier. Note that the input to the network is a $2\tau/m$ dimensional vector $\begin{bmatrix}\tilde{\mathbf{x}}^{(1)}\\ \tilde{\mathbf{x}}^{(2)}\end{bmatrix}$, and the weights that correspond to $\tilde{\mathbf{x}}^{(\ell)}$ are denoted by τ/m dimensional vectors $\mathbf{\Omega}^{(\ell)}$ for $\ell = 1, 2$.

Code availability

All relevant codes or algorithms are available from the corresponding author upon reasonable request.

References

45. Kingma, D. P. & Ba, J. Adam: a method for stochastic optimization. In *Proc. 3rd International Conference for Learning Representations* (2015). Preprint of v.9 at https://arXiv.org/abs/1412.6980v9 (2017).
46. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).

Acknowledgements

A.S. thanks E. van Nieuwenburg, G. Rostkoff and A. Izadi Rad for helpful discussions. We gratefully acknowledge support from the Multidisciplinary University Research Initiative of the Army Research Office (grant no. ARO W911NF-15-1-0397) and the Physics Frontiers Centers of the National Science Foundation at the Joint Quantum Institute (A.S. and M.H.), and from the National Science Foundation under grant no. DMR-1506969 (C.J.).

Author contributions

All authors contributed to the design of the research, the analysis of the data, and the writing of the manuscript. A.S. performed the numerical simulations and implemented the machine learning algorithms.

Competing interests

The authors declare no competing interests.

Additional information

is available for this paper at https://doi.org/10.1038/s41567-020-1018-2.

Correspondence and requests for materials should be addressed to A.S.