nature electronics

ARTICLES

https://doi.org/10.1038/s41928-019-0313-3

Temporal data classification and forecasting using a memristor-based reservoir computing system

John Moon[1,2], Wen Ma[1,2], Jong Hoon Shin[1], Fuxi Cai[1], Chao Du[1], Seung Hwan Lee[1] and Wei D. Lu[1]*

Time-series analysis including forecasting is essential in a range of fields from finance to engineering. However, long-term forecasting is difficult, particularly for cases where the underlying models and parameters are complex and unknown. Neural networks can effectively process features in temporal units and are attractive for such purposes. Reservoir computing, in particular, can offer efficient temporal processing of recurrent neural networks with a low training cost, and is thus well suited to time-series analysis and forecasting tasks. Here, we report a reservoir computing hardware system based on dynamic tungsten oxide (WO_x) memristors that can efficiently process temporal data. The internal short-term memory effects of the WO_x memristors allow the memristor-based reservoir to nonlinearly map temporal inputs into reservoir states, where the projected features can be readily processed by a linear readout function. We use the system to experimentally demonstrate two standard benchmarking tasks: isolated spoken-digit recognition with partial inputs, and chaotic system forecasting. A high classification accuracy of 99.2% is obtained for spoken-digit recognition, and autonomous chaotic time-series forecasting has been demonstrated over the long term.

Recurrent neural networks (RNNs)[1,2] offer greatly improved ability to process temporal data compared with conventional feedforward neural networks. Due to the cyclic connections among hidden neurons, which are absent from feedforward neural networks, outputs in RNNs depend on both the current inputs and the neurons' previous states, allowing RNNs to discover temporal correlations in the data. However, the cyclic connections in RNNs cause problems commonly known as vanishing gradient and exploding gradient, which make the training process expensive and difficult.

Variations of RNNs have been proposed to solve these problems, including long short-term memory (LSTM)[3] and reservoir computing (RC)[4]. In RC, in particular, a dynamic 'reservoir' that offers a short-term memory (that is, fading memory) property is used to nonlinearly map the temporal inputs into a high-dimensional feature space, represented by the states of the nodes forming the reservoir. This nonlinear mapping can cause the initial complex inputs to become linearly separable in the new space based on reservoir states, so that further processing can be performed using a simple, linear network layer (often called the readout layer)[4]. Software-based RC systems have already achieved state-of-the-art performance for tasks including speech recognition[5] (such as phoneme recognition[6]). More importantly, temporal data analysis, which is difficult to perform for conventional neural networks, is naturally suited for RC systems due to the reservoir's capability to map diverse features at different timescales. For example, chaotic system prediction is an especially difficult problem due to the high sensitivity of the system to error, and forecasting of large spatiotemporally chaotic systems has only recently been demonstrated using a software-based RC system[7,8]. RC systems have also shown superior performance to conventional neural networks in other time-series forecasting tasks, including prediction of financial systems[9] and water inflow[10].

Recent studies have aimed at hardware implementation of RC systems, using dynamic memristors[11], atomic switch networks[12], silicon photonics[13] and spintronic oscillators[14]. Of the different approaches, memristor-based RC systems can be fabricated using standard foundry processes and materials to allow direct integration with control and sensing elements with high density, making them particularly attractive for hardware implementations. Memristors[15–18] have already been extensively studied for neuromorphic computing applications[19–22], and studies on memristor-based RC systems[11] show that the intrinsic nonlinear characteristics and short-term memory effects of memristors provide central properties of a good reservoir, namely, separation and echo state properties[4]. Tasks such as hand-written digit recognition have been experimentally demonstrated using memristor-based RC systems[11].

Since the performance of the RC system depends strongly on the dimensionality of the reservoir space, increasing the number of nodes in the reservoir has become an experimental challenge. An interesting approach to this problem is to build the reservoir using a single physical nonlinear node subjected to delayed feedback, which can effectively act as a chain of virtual nodes without much performance degradation compared with a conventional reservoir[23]. Since only a single physical node is needed, the delay-system based RC approach is attractive for hardware implementation using emerging devices, and has been demonstrated recently using photonic- and spintronic-based systems[24–26].

In this article, we show that memristor-based RC systems employing the virtual-node concept can be used to efficiently process temporal data, producing excellent results for important tasks such as speech recognition and time-series forecasting. For example, time-series forecasting is an active research area that can impact broad fields. The traditional approach for time-series forecasting is to attempt to derive a set of equations describing the desired system from abundant observations. However, in most cases it is practically impossible to obtain the equations representing the system accurately. Alternatively, prediction may be performed on the basis of past and present data using statistics-based[27,28] or machine-learning techniques[29,30]. However, the prediction accuracy of statistics-based techniques is limited by the parameters of the assumed model

[1]Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA. [2]These authors contributed equally: John Moon, Wen Ma. *e-mail: wluee@umich.edu

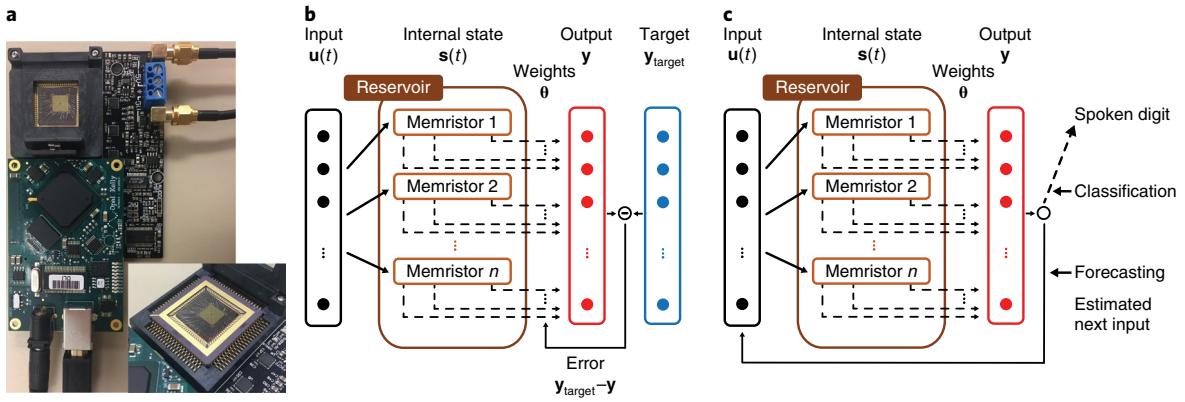

Fig. 1 | Memristor-based RC system. a, Optical image of the hardware system. The memristor array is wire-bonded and mounted on the test board system. Inset: optical image of the 32×32 memristor chip. **b**, Schematic of the training phase for the memristor-based RC system. Weights in the readout layer are updated to reduce the output error. **c**, Schematic of the testing phase for the memristor-based RC system. The system can be used to perform either classification tasks, or forecasting tasks where the output produced from the system is fed back to the reservoir as input for the next frame.

and the complexity of the system. Typical systems often could not adequately capture the nonlinear relationships in the data, even with nonlinear dependences included in the model. Neural networks offer a more general and flexible tool since they do not depend on parameters of specific tasks but are driven only by the data[31]. In particular, RNNs have gathered much attention for forecasting since the temporal information captured by the recurrent connections improves the prediction performance[32–34], although at increased training cost. To this end, RC systems offer the capability of efficient temporal processing at low training cost, and are thus well suited for time-series analysis and forecasting tasks, including autonomous prediction of chaotic systems[7,8,26].

We show that time-series forecasting, including spoken-digit recognition based on partial inputs and autonomous prediction of chaotic systems, can be successfully performed using a memristor-based RC hardware system. Features in the temporal inputs are effectively captured by the native short-term dynamics of the memristor devices, instead of loops in conventional RC systems[4,7] or delayed feedback[23]. By processing the temporal features mapped in the memristor-based reservoir, efficient analysis and even forecasting become feasible. For example, in speech recognition accuracy over 57.8% can be achieved using only one-quarter of the complete input sequence, and over 98.2% accuracy can be achieved with 62.5% of the complete input. Forecasting a chaotic system, the Mackey–Glass time series, is also successfully implemented experimentally without knowing any information about the equation describing the system. Autonomous prediction up to 50 time steps is demonstrated by applying the predicted next step as input to the system without any external feedback, and much longer prediction becomes possible through periodic updates that prevent the memristor-based reservoir from diverging from the original dynamics.

Memristor-based RC system

In a typical memristor, the applied electrical stimulus triggers the migration of oxygen vacancies or metal ions in the switching layer, leading to modulations of the local resistivity and the overall device resistance[35]. In some devices, the system can quickly relax back to the original state due to spontaneous ion diffusion, leading to a short-term memory behaviour[36,37]. These devices natively offer the 'fading memory' properties required by a reservoir and have been used to implement RC systems for tasks such as hand-written digit recognition where the features in the spatial domain are converted into features in streaming inputs[11].

We note, however, that there are major differences in the reservoir configurations between a conventional RC system and the memristor-based one. Typical reservoirs may consist of several hundreds or thousands of internal nodes through complex interconnections, where the states of the nodes are all accessible by the readout network. In the memristor-based implementation, the short-term memory effect is obtained natively from a single device, instead of loops formed by multiple nodes. As a result, a reservoir may consist of a single device (node) where the reservoir state is represented by the excited memristor state and extracted from conductance measurements. This allows much simpler experimental implementation, but also limits the size of the reservoir. Multiple devices can be used to expand the reservoir size on the basis of device-to-device variations where the reservoir state is represented by the collective states of all devices[11], but the nodes are independent of each other in this case instead of being nonlinearly coupled. To expand the dimensionality of the reservoir and improve the system's performance, we adapt the concept of virtual nodes originally developed in delay systems[23]. In a delay system, a single physical nonlinear node is subjected to delayed feedback, where the excitations of the physical node in response to the delayed signals can effectively act as a chain of virtual nodes[23]. The state of virtual nodes depends on the node's own previous state, the current state of adjacent nodes and the masked input signal, allowing them to be nonlinearly coupled. By using randomly generated masks for input signals, diverse responses can be obtained from the virtual nodes, and the delay systems have been shown to be able to achieve performance comparable to that of conventional, well designed reservoirs[23].

We hypothesize that the general virtual-node concept, represented as the nonlinear response of a physical memristor device at selected time steps in response to a streaming input, can similarly be used to increase the reservoir size and allow better mapping of the input features. Below we experimentally implement this concept and show that this approach does indeed allow the memristor-based RC system to successfully perform complex tasks such as temporal data analysis and forecasting.

The hardware implementation is based on a 32×32 WO_x memristor array (Supplementary Fig. 1 and Note 1), mounted on a custom-built test board (Fig. 1a and Methods). The operation of the RC system is shown in Fig. 1b,c, for the training phase and the testing phase, respectively. During the training phase, a teacher signal $\mathbf{u}(t)$ is applied to the reservoir to bring the reservoir to different excited states (depending on the temporal streaming inputs). The weights $\boldsymbol{\theta}$ in the readout layer are then trained to minimize the error between the output of the readout network $\mathbf{y}$ and the target output $\mathbf{y}_{target}$. The target can be either the correct digit label (for classification) or the next value in the time-series data (for forecasting). Note that only

weights in the linear readout layer need to be trained, while the connections in the reservoir remain fixed. In the testing phase, testing inputs, not included in the training set, are applied to the reservoir. Depending on the type of task, the final output implementation can be quite different. For classification, the system configuration remains the same as in the training phase, and the performance of the system is evaluated by analysing how many testing samples are correctly classified. On the other hand, in forecasting applications, the output from the readout layer is fed back to the system as the input signal for the reservoir at the next time step. In this case, external inputs are not needed, and the system is left running autonomously to generate the target time series.

Speech recognition

To validate the proposed approach, we first perform a standard speech recognition benchmark—isolated spoken-digit recognition using the memristor-based RC system. The inputs for the reservoir are sound waveforms of isolated spoken digits (0–9 in English) from the NIST TI46 database[38], preprocessed using Lyon's passive ear model[39] based on human cochlear channels. The preprocessing transforms the sound waveforms into a set of 50-dimensional vectors (corresponding to the frequency channels) with up to 40 time steps (Supplementary Fig. 2 and Note 2). One example representing digit 0 is shown in Fig. 2a. Each data point on the graph represents the firing probability of a neuron corresponding to a specific frequency channel at a time point, with 50 frequency channels in total plotted along the y axis and time plotted along the x axis. The input graph (termed a cochleagram) is then digitized to form the streaming inputs to the reservoir[40] (Fig. 2b). For this isolated spoken-digit classification task, 50 WO_x memristor devices are used to experimentally implement the RC system, with each device processing the input spike train in one of the 50 channels (see Methods).

Figure 2c shows the responses obtained experimentally from the memristors to spike trains in frequency channels 47 and 19, respectively. The input data were taken from female speaker 1, first utterance in the database. Due to the short-term dynamics of the memristors, the temporal patterns in the input spike trains led to diverse but deterministic device responses, where the device conductance is increased when stimulated by a spike and decays spontaneously, with the device conductance at a specific time depending on the recent history of the temporal inputs[11]. However, the temporal information of the early part of the input sequence (that is, far history) is not conveyed in the final responses of memristors. For example, as shown in Fig. 2c, even though the device responses are clearly different within time steps 10–35 of the streaming inputs, the responses in the end at time step 40 do not show large difference between the two inputs due to similarities in the last part of the two inputs (time steps 36–40). The loss of information will lead to poor classification results. To address this problem, we divide the whole input sequence into n equal intervals and measure and record the device state at the end of each interval. This process effectively creates n virtual nodes from a single device. The extracted virtual-node state is affected by the temporal input pattern in the near history of that virtual node, and also by the previous virtual-node states. The chain of virtual-node states in turn forms nodes in the reservoir to allow effective temporal data analysis.

Specifically, we measure the device conductance at time steps $40/n$, $40\times2/n$, $40\times3/n$, $40\times4/n$, …, $40\times(n-1)/n$ and 40 as the virtual nodes' states in the reservoir. One example of $n=8$ is shown in Fig. 2c, where the red dots represent the measured virtual-node states. The virtual-node states are then supplied to the readout layer (a 400×10 perceptron in this case, see Methods) for training and testing. Figure 2d shows the confusion matrix obtained experimentally during testing, after training the readout layer with 450 speech samples. Overall, a high recognition rate of over 99.2% was obtained for all inputs using the memristor-based RC system (see Methods).

To verify the effectiveness of the memristor-based reservoir in capturing temporal features in the input, the memristor-based system was compared with other systems based on conventional convolutional filters. The memristor-based system shows clear advantages at the same network size, due to the device's capability to natively perform nonlinear transformation of the input, and as well as the capability to capture local temporal features within an input window and the more global features between input windows, enabled by the internal short-term memory effect of the devices (Supplementary Fig. 3 and Note 3).

More interestingly, as the memristor-based reservoir maps the temporal features associated with different classes of inputs, it is possible to use the RC system to predict the spoken digit before the utterance is completed. This hypothesis was tested experimentally, as shown in Fig. 3. When the first 25% of the speech signal is used, each device produces only one virtual node, so a 100×10 readout network is used during the training phase. As shown in Fig. 3b, after training an 57.8% recognition rate can already be achieved using only the first 25% of the whole sequence. The recognition accuracy increases when the portion of the available input increases (for example, 98.2% recognition rate for using the first 62.5% of the input, and 99.2% recognition rate when the whole speech sequence is used). The evolution of the recognition rate as a function of the portion of input used is shown in Supplementary Fig. 4. The 99.2% recognition rate is comparable to results achieved in RC systems based on spintronic[14] or photonic[24,25] devices, although this experiment also verified the RC system's ability to perform spoken-digit classification using only part of the input sequence.

Time-series forecasting

The isolated spoken-digit recognition task has relatively good tolerance to prediction error because the task only requires identification of the largest output among the 10 outputs. Thus, as long as the selected label is correct, small errors in the exact output value do not degrade the system's performance. In contrast, forecasting time-series data is a much more difficult task since the quantitative difference between the ground truth and the predicted value matters and the difference can be further accumulated in subsequent predictions. One standard benchmark test for time-series forecasting is predicting a chaotic system, which is inherently very challenging due to the positive Lyapunov exponent[41] in chaotic systems, which leads to exponential growth of separation of close trajectories so that even small errors in prediction can quickly lead to divergence of the prediction from the ground truth.

To see if the memristor-based RC system can be used for long-term prediction, we tested the system using the Mackey–Glass time series[42,43] $\frac{dx}{dt}=\beta\frac{x(t-\tau)}{1+(x(t-\tau))^n}-\gamma x(t)$. These types of chaotic system have a deterministic form but are difficult to predict[7,26] and thus have been widely used as a benchmark for forecasting task tests. To obtain chaotic dynamics, we set the parameters $\beta=0.2$, $\gamma=0.1$, $\tau=18$, $n=10$ in our studies (see Methods).

To improve the accuracy of prediction, it is necessary to build a reservoir system that can capture the temporal dynamics of the given chaotic system as closely as possible, which in turn allows the readout network to achieve accurate prediction. To achieve this goal, we used several techniques to expand the reservoir dynamics. First, we used m different memristors for our memristor-based reservoir system. Due to the naturally occurring device-to-device variations (Supplementary Fig. 5), the devices produce responses that are qualitatively similar but quantitatively different even with the same input, and such device variations expand the response of the reservoir. Second, n virtual nodes are obtained from each physical memristor, on the basis of the device response at the present time step and $n-1$ previous time steps, similar to the approach used in the classification task (Supplementary Fig. 6 and Note 4).

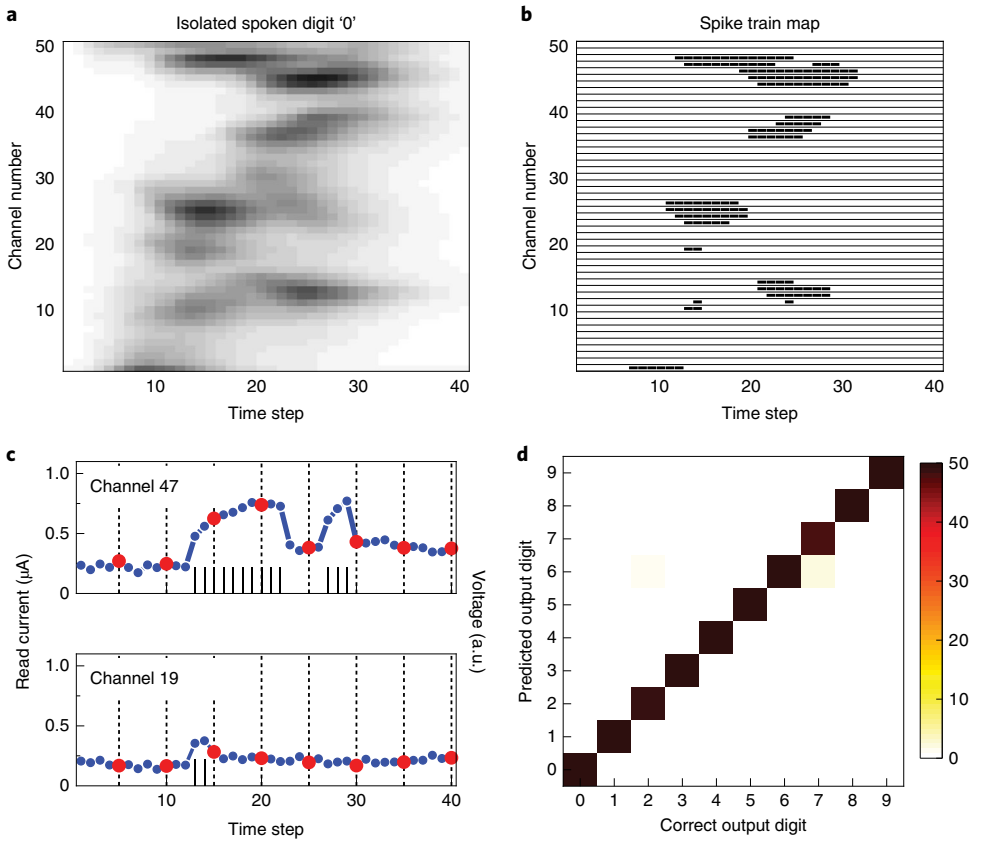

Fig. 2 | Spoken-digit recognition task implementation. a, Cochleagram of the first female speaker, first utterance speech sample after being processed by Lyon's passive ear model. Each data point represents the firing probability of a hair cell sensitive to a certain frequency (channel) at a given time point. **b**, Digitized spike trains converted from the cochleagram shown in **a**, by setting a threshold (0.5) for the firing probability. **c**, Temporal response of memristors to the spike trains in channels 47 and 19. The current (blue dots) was measured by read pulses through the test board. The spike train voltage inputs (black lines) are also plotted for reference. The complete input can be divided into eight intervals, representing eight virtual nodes whose states are measured at the end of the intervals (red dots). **d**, Confusion matrix showing the experimentally obtained classification results from the memristor-based reservoir system versus the correct outputs. An overall recognition rate of 99.2% is achieved. Colour bar: occurrence of a given predicted output.

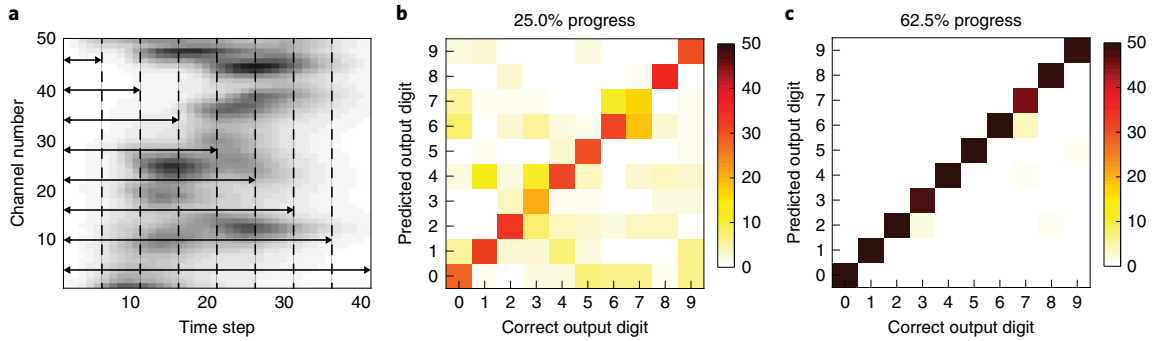

Fig. 3 | Classification using partial inputs. a, Schematic of the different portions of data used for classification. **b**,**c**, Confusion matrices showing the experimental classification results obtained using 25.0% (**b**) and 62.5% (**c**) of the input signal. Colour bars: occurrence of a given predicted output.

The ability of the memristor-based RC system in time-series prediction was tested both experimentally using a reservoir with $m = 20$ devices and $n = 50$ virtual nodes for each device, and through simulations of the reservoir using a realistic device model (see Methods). The reservoir states are then applied to the readout layer (a $1{,}000 \times 1$ network, see Methods) to generate the predicted data for the next time step. Figure 4a,c shows the results obtained during training from the experimental measurements and the simulation, respectively. Excellent agreement between the target and the predicted value can be obtained, indicating that the trained readout weights can correctly calculate the next time-step signal on the basis of the internal states of the reservoir. Further evidence of successful training can be found by examining the network performance in the frequency domain and in the phase space, as shown in Fig. 4e,f for the experimental results, where an excellent match can again be observed between the memristor RC output and the ground truth. The experimental results are further verified through simulation, as shown in Fig. 4c and Supplementary Fig. 7.

The network is then used to forecast the time series autonomously. Before the start of the autonomous prediction, an

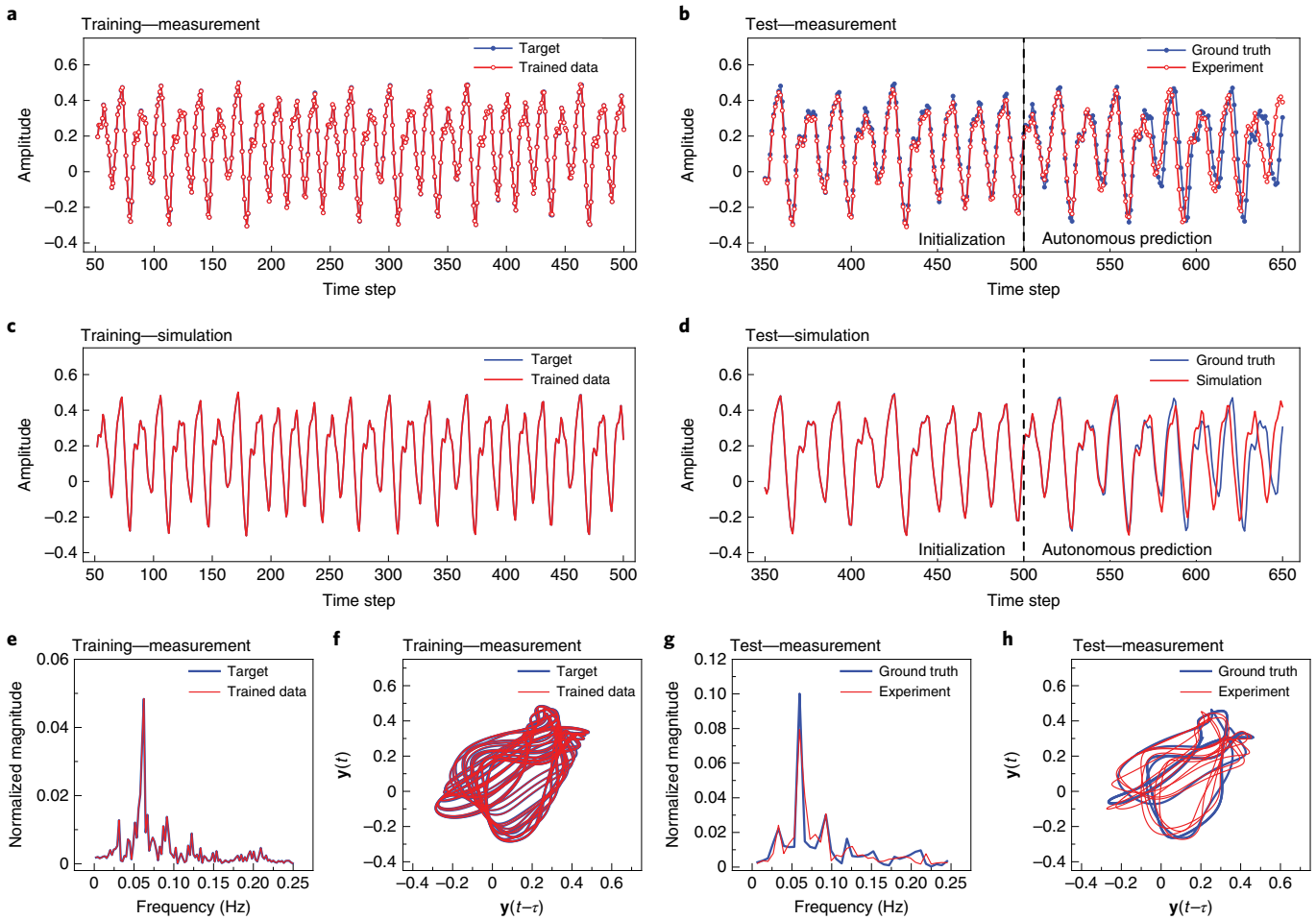

Fig. 4 | Autonomous forecasting of Mackey–Glass time series. a,**b**, Training (**a**) and forecasting (**b**) results obtained experimentally from the memristor-based RC system. The ground truth (blue) and the predicted output from the RC system (red) are plotted. An initialization step is used before the autonomous forecasting process, which starts from time step 501 in **b**. **c**,**d**, Training (**c**) and forecasting (**d**) results obtained from simulation using a realistic device model and the same RC system set-up as in the experiments. **e**,**f**, Experimental results of the memristor RC system output plotted in the frequency domain (**e**) and in the phase space (**f**) for the training stage. **g**,**h**, Experimental results of the memristor RC system output during autonomous forecasting, plotted in the frequency domain (**g**) and in the phase space (**h**).

initialization stage is needed to prepare the internal states of the reservoir, since chaotic systems depend strongly on the initial conditions. During the initialization stage, the feedback connection between the network output and the reservoir input is removed, and the true input data are applied to the reservoir instead. Note that the goal of the initialization is to simply excite the reservoir to the state just before autonomous prediction starts, not to train the system. After the reservoir is initialized, the output from the readout function, that is, the predicted data for the next time step, is then connected to the reservoir as the new input, and the system autonomously produces the forecasted time series continuously.

Figure 4b,d shows the results of the experimental measurements and simulation, respectively, for autonomous time-series prediction using the memristor-based RC system. As the reservoir states are stabilized during the initialization stage, the output values obtained from the readout layer follow accurately the correct values. Afterwards, the autonomously generated output (from the 500th time step onwards) still matches very well the ground truth (which is not used in the signal generation but simply used as a reference in the plot), showing the ability of the memristor-based RC system to autonomously forecast the chaotic system. After 60–70 time steps of autonomous prediction, the predicted signal starts to diverge from the correct value, in both the simulation and the experiment. Careful analysis of the divergence shows that small errors in the prediction are accumulated during the autonomous prediction, and can lead to a phase shift of the time-series data as represented by missing one or more data points in the prediction, resulting in increased prediction error as shown in Fig. 4b at time steps 570–650 (Supplementary Note 5). Nevertheless, even though in the long term phase shifts occur due to the small network size in the experiment and the accumulation of errors, the memristor-based RC system can still correctly capture the characteristic temporal dynamics of the Mackey–Glass series (see Supplementary Fig. 7 for data up to 1,000 time steps and the trace plots in Supplementary Video 1). Interestingly, with the phase shifts we often observe the autonomous forecasting to precede the ground truth (versus following the ground truth), while maintaining the same characteristic dynamics (Supplementary Video 1). Examination of data in the frequency domain also shows similar frequency peaks in the autonomously generated data and the ground truth (Fig. 4g), and similar trace plots can be observed in the phase space (Fig. 4h). We note that if the memristor-based RC system could not emulate the chaotic system's dynamics the prediction would instead converge to a stable point or periodic orbits. Indeed, if the reservoir size is reduced, for example to only one physical device, the autonomous predictions obtained both in the experiment and in the simulation decay gradually and

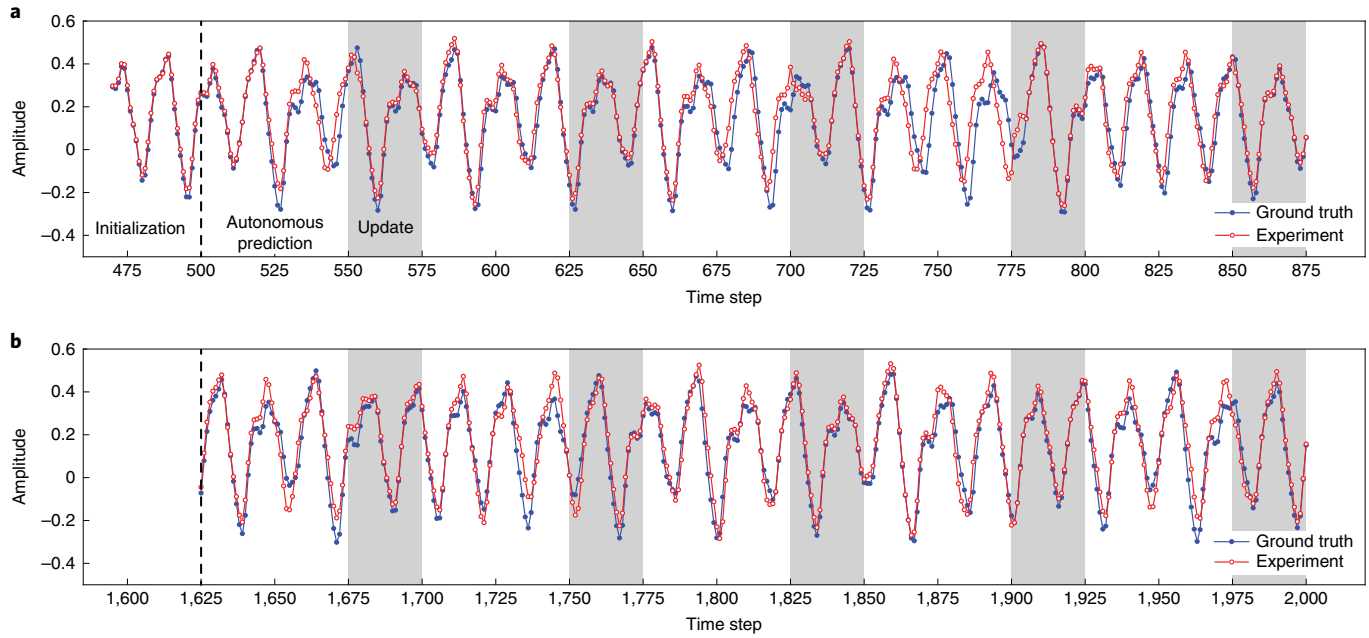

Fig. 5 | Long-term forecasting of Mackey–Glass time series with periodic updates. a, Experimental output from the memristor-based RC system at the beginning of the test, showing results from the autonomous forecasting segments (50 time steps each), followed by the update segments (shaded regions, 25 time steps each). **b**, Experimental output from the memristor-based RC system at the end of the test. Reliable forecasting can still be obtained, aided by the periodic updates.

finally converge to a stable point (Supplementary Fig. 8 and Note 6) instead of showing the targeted chaotic behaviours. In this regard, the device-to-device variations are beneficial in helping expand the reservoir space and improving the RC system's performance.

Increasing the size of the reservoir further by using more memristors and using more previous states (virtual nodes) may further reduce the prediction error so that the length of accurate prediction can be further increased. However, it will increase hardware implementation cost as well as the training cost of the readout layer. Furthermore, even with a reduced error using a larger reservoir, accumulation of the prediction error during the autonomous prediction is inevitable, and more importantly the length of accurate prediction will not increase proportionally since the overall error grows exponentially in time. Thus, the autonomous prediction will exponentially diverge from the original chaotic system, no matter how small the initial prediction error is.

Instead of increasing the network size to extend the range of correct prediction further into the future, we explored an approach where the reservoir forecasts moderately into the future, and the reservoir state is then periodically pushed back to the original dynamics before the prediction significantly diverges from the ground truth. This 'update' stage is similar to the initialization stage, and is applied after a period of autonomous prediction by sending the true input instead of the predicted value to the reservoir for a short time period, as shown in Fig. 5. No retraining is needed in the update stage. With these periodic updates, long-term prediction of chaotic systems becomes feasible. For example, by iteratively implementing 50 time steps of autonomous prediction, followed by 25 time steps of update, the experimental prediction shows excellent agreement with the true values over 2,000 time steps (limited only by the buffer size of the test board). During the update stage, the reservoir reverses deviation from the original chaotic system caused by the prediction error, and after the update sequence the memristor-based reservoir can again correctly perform prediction for the next 50 time steps.

Compared with the autonomous prediction without updates, prediction with periodic updates produces more stable and accurate results, both in the time and frequency domains and in the phase space (Supplementary Fig. 9). Note that in many cases the ground truth can indeed be periodically measured, so that periodically updating the reservoir to maintain stable operation of the system is feasible, and reliable predictions can be obtained during the periods when the ground truth cannot be measured.

Future directions for architecture and device developments

We note that the ability to capture diverse temporal features in RC systems, including the memristor-based RC system discussed here, is critical for the system to perform complex temporal data-processing tasks such as chaotic system forecasting. Our control studies on autonomous prediction using conventional feedforward neural networks show that these systems easily get stuck in a periodic pattern rather than producing the desired chaotic pattern, even at model sizes much larger than the RC system (Supplementary Figs. 10 and 11 and Note 7). Although the neuron activation functions are nonlinear in feedforward neural networks, the weights and summations are linear so that inputs from different time steps are processed similarly. In contrast, RC systems transform inputs from different time steps in a very nonlinear fashion, allowing these systems to capture diverse temporal features and giving them better capability to process temporal data.

Compared with digital implementations of RC systems, the memristor-based hardware implementation offers much lower energy cost (Supplementary Fig. 12, Table 1 and Note 8). The energy can be further reduced by reducing the programming current of the devices and by using shorter input pulses.

In our current implementation, device-to-device variations help improve the system performance by expanding the effective reservoir size. However, the random device-to-device variations make it difficult to build copies of the trained RC systems for inference tasks. In the future, we expect that intentionally controlled device characteristics offering distinct timescales will be obtained, by tightly controlling fabrication conditions such as the oxidation time and temperature for different devices in the reservoir[44]. Better

RC performance can also be expected, since features can be captured in broader timescales not achievable solely from random device variations.

In contrast to device-to-device variations, cycle-to-cycle variations are detrimental to the system performance since they lead to errors during temporal signal transformation through the reservoir. The WO_x-based memristor devices showed good cycling characteristics; however, minor cycle-to-cycle variations can still be observed (Supplementary Fig. 13 and Note 9). The cycle-to-cycle variation acts as an error source during experimental implementation, and can explain the minor differences in the experimental and simulation results obtained for the chaotic system prediction task. Further device optimizations that reduces stochasticity during the switching process will be desirable to help further minimize cycle-to-cycle variations and improve the system performance.

The computing capacity of an RC system is mostly determined by how well the reservoir can capture the diverse temporal features and map these features to the reservoir state. Similar to conventional networks, expanding the depth or width of the reservoir can further boost the computing capacity. Deep RC systems that utilize layered subreservoirs can help the system extract features at multiple timescales[45,46]. Theoretical studies[45,47] have also suggested that such deep RC systems can operate near the edge of chaos, which is a known optimal operating point for RC. The width of RC systems can similarly be expanded by utilizing subreservoirs with dynamics at different timescales (Supplementary Fig. 14 and Note 10).

More practical forecasting tasks such as long-term forecasting of monthly sunspots (Supplementary Fig. 15) and industry and consumer trends (Supplementary Fig. 16) also seem possible. For data that show clean, periodic behaviours such as the electricity production case, statistics-based prediction tools can show comparable or even better performance; however, machine-learning algorithms including LSTM and the memristor-based RC system show improvements when the data are noisy (such as the beer production example, Supplementary Fig. 17 and Note 11). In addition, the memristor-based RC system shows clearly better performance when the data are very complex (such as the Mackey–Glass example, Supplementary Fig. 18), while predictions from both statistics-based tools and LSTM get stuck in periodic behaviours. In general, processing more complex tasks requires the ability to capture diverse temporal features, and the memristor-based RC system can offer an advantage in these cases.

Conclusions

In this study, a memristor-based RC system that utilizes the internal short-term ionic dynamics of memristor devices and the concept of virtual nodes is successfully demonstrated for time-series analysis and prediction tasks. A high classification accuracy of 99.2% was obtained for spoken-digit recognition. Since the reservoir maps the temporal features of the input, good classification was still obtained even with partial inputs. With the use of periodic updates to bring the reservoir back to the original dynamics, prediction can be maintained long term without retraining the system even for chaotic tasks.

The memristor crossbar array used in this work provides the high-density devices where the memristor devices in the reservoir work independently and in parallel to process the spatiotemporal data, for example inputs from different frequency channels. Further improvements in the reservoir system may involve using interconnected devices to form more complex reservoir structures, with properly designed connecting weights and loops in the system. Additional theoretical and algorithm developments will be necessary to help illustrate how broadly RC systems based on intrinsic device dynamics can be used in general machine-learning tasks. These theoretical developments, along with continued material and device optimizations and advances of integrated systems consisting of memristor arrays directly fabricated on complementary metal–oxide–semiconductor control and logic circuits[48], will further broaden the appeal of the memristor-based RC systems for practical applications, such as real-time temporal data processing in power-constrained environments[49].

Methods

Device fabrication. The memristor-based reservoir system was fabricated in a crossbar structure, in which WO_x memristors are formed at each cross point. Starting from a substrate with 100 nm thermally grown silicon oxide on a silicon wafer, 60 nm W was deposited by magnetron sputtering and patterned by e-beam lithography and reactive ion etching using Ni as a hard mask to form W bottom electrodes (BEs) with 500 nm width. The Ni hard mask was then removed by HCl wet etching. A spacer structure formed along the sidewalls of the W BEs was employed to improve the fabrication yield. The spacer was patterned by the deposition of 250-nm-thick SiO_2 through plasma-enhanced chemical vapour deposition, followed by etch back by reactive ion etching. The WO_x switching layer was then formed on the exposed W BEs by rapid thermal annealing in oxygen gas ambient at 375 °C for 45 s. Afterwards, the top electrodes (TEs) with 500 nm width were patterned by e-beam lithography, e-beam evaporation of 90 nm Pd and 50 nm Au, and a lift-off process. A reactive ion etching process was then used to remove the WO_x between the TEs to isolate the devices and to expose the BEs for electrical contacts. Finally, photolithography, e-beam evaporation and lift-off processes were performed to form wire-bonding pads of 150-nm-thick Au. After fabrication, the memristor chip was wire-bonded to a chip carrier and mounted on a custom-built test board for electrical testing. Supplementary Fig. 1 shows a schematic of the memristor device structure and a magnified scanning electron microscopy image of the 32 × 32 memristor array.

Experimental set-up. For the isolated spoken-digit recognition experiment, the inputs are converted into digitized spike trains and applied to the memristor-based reservoir. The amplitude/width of the spikes are 3.0 V/10 µs, and the device conductance is read out with read pulses of 0.6 V/200 µs. The length of a unit time step in the experiment is 250 µs.

For the time-series forecasting experiment, the ground truth is obtained by solving the Mackey–Glass equation using the Runge–Kutta 4 method and normalized into the $[-0.5, 0.5]$ range. The input signal $u(t)$ is linearly converted to a programming pulse with 300 µs pulse width and amplitude $V(t) = u(t) + 1.8$. The amplitude/width of the read pulse to obtain the virtual-node state is 0.6 V/ 200 µs. The length of a unit time step is 150 ms.

Speech recognition. Sound waveforms of isolated spoken digits (0–9 in English) from the NIST TI46 database are transformed into a set of 50 frequency channels with 40 time steps using Lyon's passive ear model based on human cochlear channels. The output of Lyon's passive ear model, which is the firing probability, is digitized on the basis of whether the firing probability is higher than 0.5 or not. As there are 50 frequency channels, and n virtual nodes in each channel, there are $50n$ virtual nodes in total that form the reservoir. The readout layer is thus a $50n \times 10$ network with 10 outputs representing the 10 different digits. To train the readout network, we used the Python toolkit Keras, which provides a high-level application programming interface to access TensorFlow. A supervised learning algorithm, softmax regression, was used to train the readout network. A softmax function is used as the activation function of the readout network to calculate the probability corresponding to the different possible outputs. The cost is calculated following a categorical cross-entropy. A standard gradient-based optimization method, RMSprop, is used to minimize the cost function and train the output network. After training the readout function using 450 speech samples from the database, test data (from the 50 speech samples not in the training data) are fed into the reservoir, and classification is performed from the readout network on the basis of the reservoir state for each test case. To prevent the system from being overfitted to specific selections of the training and testing data, 10-fold cross validation was used. Specifically, training and testing were repeated 10 times using 500 speech samples (450 samples for training and 50 samples for testing), with each time having a different assignment of training and testing samples. The recognition rate was defined as the mean in a 10-fold cross-validation set-up during training. The same activation function, cost function and learning algorithm were used in the control studies.

Time-series forecasting. The Mackey–Glass time series is based on a nonlinear time-delayed differential equation that can display a wide range of periodic and chaotic behaviours, depending on the values of the parameters. For example, $\tau < 4.43$ produces a fixed-point attractor, $4.43 < \tau < 13.3$ produces a stable limit cycle attractor, $13.3 < \tau < 16.8$ produces a double limit cycle attractor and $\tau > 16.8$ produces chaotic behaviours. We set the parameters $\beta = 0.2$, $\gamma = 0.1$, $\tau = 18$, $n = 10$. The time-series data are normalized into the range $[-0.5, 0.5]$. As we build 20 devices and 50 virtual nodes for each device, the internal state of the reservoir is represented by a vector containing 1,000 elements, which is then applied to the readout network. A supervised learning algorithm, linear regression, was used to train the readout function. Stochastic gradient descent is used to minimize the cost, calculated by mean squared error, and train the output network. Results from the virtual-node states are applied to the $1{,}000 \times 1$ readout network to generate one output, which corresponds to the predicted signal of the next time step in the time

series. In the training phase, 500 time steps of the training set are applied to the reservoir. The predicted output is compared with the ground truth (at time steps 51–500), and the error is calculated and used to update the weights in the readout network following the linear regression learning rule.

Data availability

The data that support the plots within this paper and other findings of this study are available from the corresponding author on reasonable request.

Received: 5 August 2018; Accepted: 9 September 2019;
Published online: 14 October 2019

References

1. Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl Acad. Sci. USA* **79**, 2554–2558 (1982).
2. Werbos, P. J. Backpropagation through time: what it does and how to do it. *Proc. IEEE* **78**, 1550–1560 (1990).
3. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput.* **9**, 1735–1780 (1997).
4. Lukoševičius, M. & Jaeger, H. Reservoir computing approaches to recurrent neural network training. *Comput. Sci. Rev.* **3**, 127–149 (2009).
5. Verstraeten, D., Schrauwen, B. & Stroobandt, D. Reservoir-based techniques for speech recognition. In *2006 International Joint Conference on Neural Networks (IJCNN)* 1050–1053 (IEEE, 2006).
6. Triefenbach, F., Jalalvand, A., Schrauwen, B. & Martens, J.-P. Phoneme recognition with large hierarchical reservoirs. *Adv. Neural Inf. Process. Syst.* **23**, 2307–2315 (2010).
7. Jaeger, H. & Haas, H. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science* **304**, 78–80 (2004).
8. Pathak, J., Hunt, B., Girvan, M., Lu, Z. & Ott, E. Model-free prediction of large spatiotemporally chaotic systems from data: a reservoir computing approach. *Phys. Rev. Lett.* **120**, 024102 (2018).
9. Ilies, I. et al. *Stepping Forward Through Echoes of the Past: Forecasting with Echo State Networks* Technical Report http://www.neural-forecasting-competition.com/downloads/NN3/methods/27-NN3_Herbert_Jaeger_report.pdf (Jacobs University Bremen, 2007).
10. Sacchi, R., Ozturk, M. C., Principe, J. C., Carneiro, A. A. & Da Silva, I. N. Water inflow forecasting using the echo state network: a Brazilian case study. In *2007 International Joint Conference on Neural Networks (IJCNN)* 2403–2408 (IEEE, 2007).
11. Du, C. et al. Reservoir computing using dynamic memristors for temporal information processing. *Nat. Commun.* **8**, 2204 (2017).
12. Sillin, H. O. et al. A theoretical and experimental study of neuromorphic atomic switch networks for reservoir computing. *Nanotechnology* **24**, 384004 (2013).
13. Vandoorne, K. et al. Experimental demonstration of reservoir computing on a silicon photonics chip. *Nat. Commun.* **5**, 3541 (2014).
14. Torrejon, J. et al. Neuromorphic computing with nanoscale spintronic oscillators. *Nature* **547**, 428–431 (2017).
15. Chua, L. O. Memristor—the missing circuit element. *IEEE Trans. Circuit Theory* **18**, 507–519 (1971).
16. Waser, R. & Aono, M. Nanoionics-based resistive switching memories. *Nat. Mater.* **6**, 833–840 (2007).
17. Strukov, D. B., Snider, G. S., Stewart, D. R. & Williams, R. S. The missing memristor found. *Nature* **453**, 80–83 (2008).
18. Pershin, Y. V. & Di Ventra, M. Neuromorphic, digital, and quantum computation with memory circuit elements. *Proc. IEEE* **100**, 2071–2080 (2011).
19. Jo, S. H. et al. Nanoscale memristor device as synapse in neuromorphic systems. *Nano Lett.* **10**, 1297–1301 (2010).
20. Yang, J. J., Strukov, D. B. & Stewart, D. R. Memristive devices for computing. *Nat. Nanotechnol.* **8**, 13–24 (2013).
21. Prezioso, M. et al. Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **521**, 61–64 (2015).
22. Sheridan, P. M. et al. Sparse coding with memristor networks. *Nat. Nanotechnol.* **12**, 784–789 (2017).
23. Appeltant, L. et al. Information processing using a single dynamical node as complex system. *Nat. Commun.* **2**, 468 (2011).
24. Larger, L. et al. Photonic information processing beyond Turing: an optoelectronic implementation of reservoir computing. *Opt. Express* **20**, 3241–3249 (2012).
25. Brunner, D., Soriano, M. C., Mirasso, C. R. & Fischer, I. Parallel photonic information processing at gigabyte per second data rates using transient states. *Nat. Commun.* **4**, 1364 (2013).
26. Antonik, P., Haelterman, M. & Massar, S. Brain-inspired photonic signal processor for generating periodic patterns and emulating chaotic systems. *Phys. Rev. Appl.* **7**, 054014 (2017).
27. Box, G. E. & Pierce, D. A. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *J. Am. Stat. Assoc.* **65**, 1509–1526 (1970).
28. Said, S. E. & Dickey, D. A. Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika* **71**, 599–607 (1984).
29. Kim, K. J. Financial time series forecasting using support vector machines. *Neurocomputing* **55**, 307–319 (2003).
30. Kuremoto, T., Kimura, S., Kobayashi, K. & Obayashi, M. Time series forecasting using a deep belief network with restricted Boltzmann machines. *Neurocomputing* **137**, 47–56 (2014).
31. Zhang, G. P. An investigation of neural networks for linear time-series forecasting. *Comput. Oper. Res.* **28**, 1183–1202 (2001).
32. Connor, J. T., Martin, R. D. & Atlas, L. E. Recurrent neural networks and robust time series prediction. *IEEE Trans. Neural Netw.* **5**, 240–254 (1994).
33. Assaad, M., Boné, R. & Cardot, H. A new boosting algorithm for improved time-series forecasting with recurrent neural networks. *Inf. Fusion* **9**, 41–55 (2008).
34. Mirikitani, D. T. & Nikolaev, N. Recursive bayesian recurrent neural networks for time-series modeling. *IEEE Trans. Neural Netw.* **21**, 262–274 (2009).
35. Yang, Y. et al. Observation of conducting filament growth in nanoscale resistive memories. *Nat. Commun.* **3**, 732 (2012).
36. Chang, T., Jo, S. H. & Lu, W. Short-term memory to long-term memory transition in a nanoscale memristor. *ACS Nano* **5**, 7669–7676 (2011).
37. Wang, Z. et al. Memristors with diffusive dynamics as synaptic emulators for neuromorphic computing. *Nat. Mater.* **16**, 101–108 (2017).
38. *Texas Instruments-Developed 46-Word Speaker-Dependent Isolated Word Corpus (TI46)* NIST Speech Disc 7-1.1 (Texas Instruments, 1991).
39. Lyon, R. F. A computational model of filtering, detection, and compression in the cochlea. *Proc. IEEE-ICASSP'82* **7**, 1282–1285 (1982).
40. Bennett, C. H., Querlioz, D. & Klein, J. O. Spatio-temporal learning with arrays of analog nanosynapses. In *2017 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)* 125–130 (IEEE, 2017).
41. Frazier, C. & Kockelman, K. M. Chaos theory and transportation systems: instructive example. *Transp. Res. Rec.* **1897**, 9–17 (2004).
42. Mackey, M. C. & Glass, L. Oscillation and chaos in physiological control systems. *Science* **197**, 287–289 (1977).
43. Farmer, J. D. Chaotic attractors of an infinite-dimensional dynamical system. *Physica D* **4**, 366–393 (1982).
44. Du, C. *Metal Oxide Memristors with Internal Dynamics for Neuromorphic Applications* (University of Michigan, 2017).
45. Gallicchio, C., Micheli, A. & Pedrelli, L. Deep reservoir computing: a critical experimental analysis. *Neurocomputing* **268**, 87–99 (2017).
46. Triefenbach, F., Jalalvand, A., Demuynck, K. & Martens, J. P. Acoustic modeling with hierarchical reservoirs. *IEEE Trans. Audio Speech Lang. Process.* **21**, 2439–2450 (2013).
47. Gallicchio, C., Micheli, A. & Silvestri, L. Local Lyapunov exponents of deep echo state networks. *Neurocomputing* **298**, 34–45 (2018).
48. Cai, F. et al. A fully integrated reprogrammable memristor-CMOS system for efficient multiply-accumulate operations. *Nat. Electron.* **2**, 290–299 (2019).
49. Tanaka, G. et al. Recent advances in physical reservoir computing: a review. *Neural Netw.* **115**, 100–123 (2019).

Acknowledgements

We acknowledge inspiring discussions with M. Zidan. This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) through award HR0011-13-2-0015, the National Science Foundation (NSF) through grant CCF-1617315, and the Applications Driving Architectures (ADA) Research Centre, a JUMP Centre cosponsored by SRC and DARPA.

Author contributions

J.M., W.M. and W.D.L. conceived the project and constructed the research frame. J.M., W.M., F.C., C.D. and S.H.L. prepared the memristor arrays and built the hardware and software package. J.M. and W.M. performed the hardware measurements. J.M., W.M., J.H.S. and W.D.L. analysed the experimental data and simulation results. W.D.L. directed the project. All authors discussed the results and implications and commented on the manuscript at all stages.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41928-019-0313-3.

Correspondence and requests for materials should be addressed to W.D.L.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.