nature portfolio

Peer Review File

Learning reservoir dynamics with temporal self-modulation

Reviewers' comments:

Reviewer #1 (Remarks to the Author):

This work aims to improve the reservoir computing-based time-series processing model with self-modulation. Conventional reservoir computing models fix the parameters of the reservoir layer and therefore obtain suboptimal performance. Targeting this problem, self-modulation is incorporated in this work to adapt the reservoir layer. Specifically, this modulation consists of two gates filtering the incoming information and the retained information. The whole structure resembles an LSTM but only the gates are trainable. Experiments are conducted on two tasks, and the performance is promising.

Some concerns are listed below:

1. Although the target problem is clear and the proposed technique is reasonable, there is one major unclear point about this work. According to the introduction, the reason that reservoir computing is more favorable over deep learning-based methods is that the data-driven methods incur significant computational load. Then, if the self-modulation is introduced into the model, what is the essential difference between learning-based methods? Wouldn't the model lose the advantage of being computationally efficient?

2. It would be better if the physical implementation of the SM-RC could be introduced with more details, as well as a comparison with the learning-based method to highlight the advantage of reservoir-based methods.

3. The proposed method lies somewhere between the fully trainable models and conventional RC without a trainable RC layer. Therefore, it would be better to also include the learning-based methods for empirical comparison.

4. In the experiments, what kind of reservoir computing layer is chosen for the baseline 'Conventional RC'? Is it possible to adopt a different design for the RC layer?

5. To demonstrate the efficiency of the proposed method, which is an advantage of RC-based methods, a running time comparison between the proposed SM-RC, conventional RC, and learning-based methods should also be provided.

Minor problems:

The first sentence of the abstract is confusing. Maybe 'transferring signal to RNNs' should be 'transferring signal via RNNs'.

Reviewer #2 (Remarks to the Author):

This paper proposes a self-modulated reservoir computing (RC) model to improve the learning ability of RC without enlarging the size of the reservoir layer. The model introduces two learnable gate units, which enable itself to learn the reservoir dynamics according to the input signal adaptively. Experiments show that the proposed SM-RC model outperforms conventional RC on 3 tasks.

Detailed comments：
1. According to the paper, the input and reservoir gates are the keys to learning the reservoir dynamics, but the organization of the two gates is unclear. I can’t tell how they combine the merits of feedback structure, reservoir-layer learning, and the attention mechanism.

2. I suggest visualization analysis of the learned gates on given inputs. An explainable conclusion can better support the motivation.

Revision Notes

Title: Learning Reservoir Dynamics with Temporal Self-Modulation

Authors: Yusuke Sakemi, Sou Nobukawa, Toshitaka Matsuki, Takashi Morie, and Kazuyuki Aihara.

Dear Reviewers,

We appreciate the time and effort that the reviewers have dedicated to providing your valuable feedback on our manuscript. We are grateful to the reviewers for their insightful comments. We have been able to incorporate changes to reflect all the suggestions provided by the reviewers and the changes are presented in red font within the revised manuscript. We hope that you find the revised manuscript suitable for publication.

In the revised manuscript, we have modified and appended data in the figures. First, we would like to apologise for including incorrect SM-RC performance data in Figs. C and D, wherein the number of epochs was 5000. In the revised manuscript, we have changed this to include the correct data wherein the number of epochs is 10000. Incidentally, this modification indicates that the SM-RC performance is even better than that presented previously; we have confirmed that this modification does not affect the subsequent discussion. Second, to answer the question raised by Reviewer #1, in the revised manuscript, Fig. 5 has been modified. In Fig. 5, we compared the learning performance of SM-RC with other state-of-the-art RNN models (LSTM and GRUs). The learning curves that were shown in the previous version of Fig. 5 were moved to Supplementary Fig. 3 C. To answer the question raised by Reviewer #2, we added Supplementary Fig. 2 in the revised manuscript. In this figure, we show the dynamics of the SM-RC models when the input gate and/or reservoir gate are temporally fixed.

The following are point-by-point responses to the reviewer's comments and concerns:

Responses to Reviewer #1

(R1.1)
Although the target problem is clear and the proposed technique is reasonable, there is one major unclear point about this work. According to the introduction, the reason that

reservoir computing is more favorable over deep learning-based methods is that the data-driven methods incur significant computational load. Then, if the self-modulation is introduced into the model, what is the essential difference between learning-based methods? Wouldn't the model lose the advantage of being computationally efficient?

(Response by authors)
Thank you very much for your constructive comment. The difference between SM-RC and other learning-based models, such as LSTM and GRUs, are twofold. First, as the RNN connections in LSTM and GRUs require training, implementing these models with physical systems wherein the internal connections cannot be finely tuned is difficult. By contrast, SM-RC employs a reservoir wherein the recurrent connections do not require training, enabling SM-RC to be implemented with physical systems. Second, the gate structure of SM-RC is considerably simpler than that of LSTM and GRUs. In LSTM and GRUs, the gate is the same as that of the output units. Therefore, RNN neurons are modulated differently. However, in SM-RC, the gates are scalar (dimension of 1).
In other words, RNN neurons (reservoir in this case) are modulated as a whole. This simple structure enables implementation with physical systems whose dynamic properties can be modulated only globally for use as reservoirs.

We have also added this explanation at the end of the third paragraph in the Discussion section (page 9) of the revised manuscript.

(R1.2)
It would be better if the physical implementation of the SM-RC could be introduced with more details, as well as a comparison with the learning-based method to highlight the advantage of reservoir-based methods.

(Response by authors)
We would like to thank you for this excellent suggestion. We believe that SM-RC can potentially become an important model for physical implementations, however, unlike digital implementations, physical implementations are domain-specific, and their detailed designs require a significant amount of effort. For example, reservoirs constructed using a photonic system [1] and spintronics devices [2] are significantly different owing to different device characteristics. Therefore, the detailed physical design of the SM-RC will be addressed in future studies. Additionally, based on the first comment (R1.1), we added an explanation of the difference between SM-RC and learning-based models at the end of the third paragraph in the Discussion section (page

9) of the revised manuscript.

[1] Lugnan et al., “Photonic neuromorphic information processing and reservoir computing” APL Photonics, 5(2):020901 (2020)
[2] Finocchio et al., “The promise of spintronics for unconventional computing”, Journal of Magnetism and Magnetic Materials, 521:167506 (2021).

(R1.3)
The proposed method lies somewhere between the fully trainable models and conventional RC without a trainable RC layer. Therefore, it would be better to also include the learning-based methods for empirical comparison.

(Response by authors)
We would like to thank you for this valuable suggestion. In the newly created Fig. 5, which is included in the revised manuscript, we compare the learning performances of SM-RC and learning-based models (LSTM and GRUs). This led to an interesting finding that the learning performance was considerably higher than the standard RC for the same number of trainable parameters, and it was comparable with that of LSTM and GRUs. These results indicate that the learning ability of RC can reach that of state-of-the-art RNN models by employing a temporal self-modulation mechanism. We have added this discussion in the last paragraph of the “Result” section in the revised manuscript.

(R1.4)
In the experiments, what kind of reservoir computing layer is chosen for the baseline 'Conventional RC'? Is it possible to adopt a different design for the RC layer?

(Response by authors)
We would like to thank you for this important question. As explained at the end of the first paragraph of the Model section, the time evolution of conventional RC is obtained by replacing the input and reservoir gates with 1. In other words, we employed a fully connected RNN with random connections as a reservoir. Of course, we can use another type of reservoir; for example, by changing the sparsity of the RNN and employing feedback from the output to the reservoir. However, in this study, we used a fully

connected RNN owing to its simplicity. Because the learning performance of RC is primarily determined based on the input strength and spectral radius, we optimized these hyperparameters through Bayesian optimization.

(R1.5)
To demonstrate the efficiency of the proposed method, which is an advantage of RC-based methods, a running time comparison between the proposed SM-RC, conventional RC, and learning-based methods should also be provided.

(Response by authors)
We would like to thank you for this constructive suggestion. Because SM-RC is targeted for implementation in physical systems, running time on CPUs and GPUs is not a primary issue. Although we could conduct performance comparisons, LSTM and GRUs provided by Pytorch are highly optimized to obtain good computation speed on GPUs. Therefore, a fair comparison between SM-RC and these RNNs would require a significant amount of coding for optimizing the computational speed of SM-RC on GPUs. This optimization, while important for developing a large-scale SM-RC model, is beyond the scope of this study. Because we discovered that the learning performance of SM-RC was comparable to that of LSTM and GRUs for the same number of trainable parameters, we assume that the computational efficiency of SM-RC on GPUs will not be significantly worse than that of LSTM and GRUs. We will aim to perform this comparison in a future work.

(R1.m1)
The first sentence of the abstract is confusing. Maybe 'transferring signal to RNNs' should be 'transferring signal via RNNs'.

(Response by authors)
We would like to thank you for this constructive comment. Based on it, we have modified the relevant text.

Response to Reviewer #2

(R2.1)

According to the paper, the input and reservoir gates are the keys to learning the reservoir dynamics, but the organization of the two gates is unclear. I can't tell how they combine the merits of feedback structure, reservoir-layer learning, and the attention mechanism.

(Response by authors)
We would like to thank you for your valuable comment. As illustrated in Supplementary Fig. 4, both gates play a complex role in the Lorentz model task. However, in the case of simple attention tasks, the role is to some extent clear. To elucidate the role of both gates, the dynamics of the SM-RC models when the input gate and/or reservoir gate are temporally fixed is shown in the newly added Supplementary Fig. 2). Evidently, the input gate takes large values only when the input pulse arrives, which efficiently feeds information into the reservoir layer and prevents the input noise from corrupting the information stored in the reservoir layer. On the other hand, the reservoir gate takes large values after the input signal, with which the input information is successfully retained in the reservoir.

(R2.2)
I suggest visualization analysis of the learned gates on given inputs. An explainable conclusion can better support the motivation.

(Response by authors)
We would like to thank you for this excellent suggestion. In response to the first question (R2.1), we have included Supplementary Fig. 2, wherein the dynamics of the learned gates are visualized in various situations.

Reviewers' comments:

Reviewer #3 (Remarks to the Author):

The paper proposes a modification to the standard RC architecture to dynamically adapt the input scaling and the spectral radius of the recurrent matrix. The presentation is clear, and the paper is written well. The fundamental idea is sound, but some concerns must be raised regarding the results.

1a) The introduction states that SM-RC is a good candidate for hardware implementation/edge AI. This is true for regular RC because most of the computing power (i.e., the recurrent part) is carried by an untrained system, making it possible to use some non-standard computing paradigms (photonic, neuromorphic, etc.). These systems follow their specific dynamics, and tuning them via a training procedure is generally hard. So, I'm not sure that it would be feasible to use the proposed model in this context, as it would require somehow training the reservoir (thus defying the whole reservoir-computing idea).

1b) On top of that, RC systems are easier to train than fully-trained RNNs, but they tend to require more recurrent neurons (sometimes, order of magnitudes more) to achieve similar performance (as it is also found in this paper). So, it is true that the computational cost of training is reduced drastically, but at the cost of a higher inference cost. So, the hardware implementation is not advantageous if the training can be carried out beforehand.
The advantage of RC systems in hardware implementations lies, in fact, in the possibility of carrying the training on the device, which practically means avoiding BPTT. However, since SM-RC needs BPTT, it is crucial to address this point.

2) It is not clear to me why the attention task requires memory at all.
If I understand correctly, the input is white noise for all time steps excluding [250-259], for which it is 1.
The output is supposed to be always 0, except for the interval [290, 291]. But why are these related? What is exactly the part of the input that the network has to recall to perform the task? What would happen if the input was just white noise and the output was left the same?

3) It is not clear how the network is trained on the various tasks. Is there a division between a training set and a test set? Is there a validation set to tune the hyper-parameters?

4) The comparison between RC and SM-RC seems to be totally unfair. The authors write.:

> We observed that SM-RC training was somewhat unstable (see Supplementary Fig. 3). Therefore, in the performance evaluation of SM-RC, we considered the best results for training with 50 different initial weights

However, the stability of the training procedures was one of the main reasons RC was introduced (as discussed in Jaeger's seminal paper). A fair comparison would be between N different random initialization of an RC vs N different SM-RC or picking the best RC out of N vs the best SM-RC out of N. The same applies to GRU and LSTM.

Some minor comments:

5) It would be nice, for comparison, to have somewhere in the methods section the equation describing the regular RC systems.

6) It would be nice to have results also for vanilla RNNs (Elman networks), as they are basically a fully trained version of the RC system.

Reviewer #4 (Remarks to the Author):

The authors have addressed the previous reviewers' comments and suggestions. I think the manuscript in its current form warrants publication in Commphysic.

My only suggestion is to further elaborate the training complexity for the input and reservoir gates and how that is compared with typical RNN as the number of neurons increases.

Revision Notes

Title: Learning Reservoir Dynamics with Temporal Self-Modulation

Authors: Yusuke Sakemi, Sou Nobukawa, Toshitaka Matsuki, Takashi Morie, and Kazuyuki Aihara.

Dear Reviewers,
We appreciate the reviewers for providing their valuable feedback on our manuscript. We have incorporated changes to reflect all the suggestions provided by the reviewers, and these changes are presented in red font within the revised manuscript. We hope that you find the revised manuscript suitable for publication.

We have modified the data plots for the conventional RC's performance in Fig. 2, Fig. 4, and Fig. 5. The data now represents the best performance among 50 random weight initializations. This modification ensures a fair performance comparison between the conventional RC and SM-RC models.

We have newly included Supplementary Fig. 3 and Fig. 6. In Supplementary Fig. 3, we show the time evolution of the conventional RC and SM-RC models with and without input pulses in the test data. Since both models failed to produce output pulses without input pulses, we confirmed that the task setting successfully prevents both models from using the initial states ($x_i(t=0)=0$) to generate output pulses. In Supplementary Fig. 6, a comparison between the conventional RC and SM-RC models is presented. This figure is the same as the one used in the main text of the previous version of the manuscript.

In all figures that show the time evolution of input and reservoir gates, the colors and line styles were made the same. Furthermore, we replaced the words "Lorentz model" in many places in the manuscript as "Lorenz model".

Responses to Reviewer #3

(R3.1a)
The introduction states that SM-RC is a good candidate for hardware implementation/edge AI. This is true for regular RC because most of the computing power (i.e., the recurrent part) is carried by an untrained system, making it possible to use some non-standard computing paradigms (photonic, neuromorphic, etc.). These

systems follow their specific dynamics, and tuning them via a training procedure is generally hard. So, I'm not sure that it would be feasible to use the proposed model in this context, as it would require somehow training the reservoir (thus defying the whole reservoir-computing idea).

(Response by authors)
As you pointed out, it is crucial to demonstrate the ease of hardware implementation of SM-RC. The prospects for the hardware implementation of SM-RC are discussed in the third paragraph of the Discussion section. We admit that the ease of hardware implementation is considered to be lower than that of a regular RC. However, since there is no need to individually optimize the internal connections of the reservoir layer in SM-RC, the ease of hardware implementation is considered to be higher compared to other RNN models such as LSTMs and GRUs. Additionally, [Nakajima] has successfully demonstrated the ability to learn connections between photonic reservoirs in a deep structure. Based on these points, we consider that SM-RC holds value by playing an intermediate role between conventional RC and fully-trained RNNs.

(R3.1b)
On top of that, RC systems are easier to train than fully-trained RNNs, but they tend to require more recurrent neurons (sometimes, order of magnitudes more) to achieve similar performance (as it is also found in this paper). So, it is true that the computational cost of training is reduced drastically, but at the cost of a higher inference cost. So, the hardware implementation is not advantageous if the training can be carried out beforehand.
The advantage of RC systems in hardware implementations lies, in fact, in the possibility of carrying the training on the device, which practically means avoiding BPTT. However, since SM-RC needs BPTT, it is crucial to address this point.

(Response by authors)
We would like to present counterarguments on the following two points.

Firstly, as you pointed out, when using computing devices such as CPUs and GPUs, RC does lead to a reduction in training costs compared to fully-trained Recurrent Neural Networks (RNNs). However, it results in an increase in the cost of inference. Nevertheless, as explained in the introduction section, RC is physically implementable. Therefore, even if the RNN size becomes large, RC can demonstrate superiority in terms

of inference speed and power efficiency.

Secondly, we believe that performing backpropagation through time (BPTT) on hardware will become feasible in the future. This is discussed in the final paragraph of the Discussion section, and recent advancements in directly training physical systems have been significant. Additionally, in [Nakajima], there has been successful learning of connections between deep structures of photonic reservoirs.

[Nakajima] Mitsumasa Nakajima, Katsuma Inoue, Kenji Tanaka, Yasuo Kuniyoshi, Toshikazu Hashimoto, and Kohei Nakajima. Physical deep learning with biologically inspired training method: gradient-free approach for physical hardware. Nature Communications, 13:7847, 2022

(R3.2)

It is not clear to me why the attention task requires memory at all.

If I understand correctly, the input is white noise for all time steps excluding [250-259], for which it is 1.

The output is supposed to be always 0, except for the interval [290, 291]. But why are these related? What is exactly the part of the input that the network has to recall to perform the task? What would happen if the input was just white noise and the output was left the same?

(Response by authors)

As you mentioned, in this attention task, the SM-RC model associates input pulses [250-259] with output pulses [290, 291]. There is a delay of 40 steps from the time of the input pulse to the time of the output pulse. Therefore, it is necessary to retain the information of the input pulses within the reservoir layer during this 40-step period. In that sense, this task requires memory functionality.

Moreover, the question you raised, 'What would happen if the input was just white noise and the output was left the same?' is very crucial. In this task, to prevent the conventional RC and SM-RC models from utilizing the initial state ($x_i(t=0)=0$) to generate the output pulse, common jitter noise is added to the input pulse and the time of the input pulse. This is explicitly mentioned in the Datasets section. In fact, we have added the results of removing the input pulse from the test data as Supplementary Fig.3. For both the RC model and the SM-RC model, it can be observed that the output pulse generation is not generated when there is no input pulse.

(R3.3)
It is not clear how the network is trained on the various tasks. Is there a division between a training set and a test set? Is there a validation set to tune the hyper-parameters?

(Response by authors)
In all tasks, including the attention task, NARMA task, and Lorenz task, we have generated training data and test data separately. The methods for generating these datasets are described in the Datasets section. While it is possible to use a validation set or cross-validation methods, in this paper, we prioritize the simplicity of the evaluation method. Similar to other papers such as [Vlachas, Inubushi], we have utilized only two datasets: one for training data and the other for test data.

[Vlachas] P.R. Vlachas, J. Pathak, B.R. Hunt, T.P. Sapsis, M. Girvan, E. Ott, and P. Koumoutsakos. Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. Neural Networks, 126:191–217, 2020
[Inubushi] Masanobu Inubushi and Susumu Goto. Transfer learning for nonlinear dynamics and its application to fluid turbulence. Phys. Rev. E, 102:043301, Oct 2020.

(R3.4)
The comparison between RC and SM-RC seems to be totally unfair. The authors write.:

> We observed that SM-RC training was somewhat unstable (see Supplementary Fig. 3). Therefore, in the performance evaluation of SM-RC, we considered the best results for training with 50 different initial weights

However, the stability of the training procedures was one of the main reasons RC was introduced (as discussed in Jaeger's seminal paper). A fair comparison would be between N different random initialization of an RC vs N different SM-RC or picking the best RC out of N vs the best SM-RC out of N. The same applies to GRU and LSTM.

(Response by authors)
In the revised manuscript, the best case among 50 initial values was used for the conventional RC models (Fig. 2, Fig. 4, and Fig. 5). For GRU and LSTM, a comparison was made using the best case among the initial 50 weight initializations in the former

manuscript. To better describe this, the caption of Fig. 5 has been revised.

(R3.5)
It would be nice, for comparison, to have somewhere in the methods section the equation describing the regular RC systems.

(Response by authors)
The equation has been added to the main text as Eq. (8, 9).

(R3.6)
It would be nice to have results also for vanilla RNNs (Elman networks), as they are basically a fully trained version of the RC system.

(Response by authors)
We have added the results of the vanilla RNN in Fig. 5. It can be observed that the performance tends to deteriorate when the network size is large ($N > 300$). This is attributed to the learning instability of the Vanilla RNN.

Responses to Reviewer #4

(R4.1)
My only suggestion is to further elaborate the training complexity for the input and reservoir gates and how that is compared with typical RNN as the number of neurons increases.

(Response by authors)
Thank you very much for your valuable comments. In the previous manuscript, it was difficult to distinguish between the learning methods of SM-RC and other RNN models. We have revised the second paragraph of the Learning Procedure section.

REVIEWERS' COMMENTS:

Reviewer #3 (Remarks to the Author):

I'm generally happy about the replies to my concerns and how the authors have modified the papers according to them.
Their responses are convincing.