# Reproducibility Report: Neural Networks Fail to Learn Periodic Functions and How to Fix It

**Anonymous Author(s)**
Affiliation
Address
email

## Reproducibility Summary

### Scope of Reproducibility

Neural Networks Fail to Learn Periodic Functions and How to Fix It [1] demonstrates experimentally that standard activations such as ReLU, tanh, and sigmoid and their variants all fail to learn to extrapolate simple periodic functions. The original paper goes on to propose a new activation, which is named the snake function.

The central claims of the paper are two-fold. (1) The properties of the activation functions are carried over to the neural networks. A tanh network will be smooth and extrapolates to a constant function, while ReLU extrapolates in a linear way. Standard neural networks with conventional activation functions are insufficient for extrapolating periodic functions. (2) The proposed activation function manages to learn periodic functions while being able to optimize as well as conventional activation functions. While both experimental proof and theoretical justifications are provided for the claims, we shall only be concerned with testing the claims via experimental means.

### Methodology

While the author was contacted to clarify certain difficulties, the reproduction of all experiments was completed using only the information provided in the original paper itself. With one exception, the link to all datasets used was also provided in the paper itself. This allowed us to implement most experiments from scratch.

### Results

We were able to successfully replicate experiments supporting the central claim of the paper, that the proposed snake nonlinearity can learn periodic functions. We also analyze the suitability of the snake activation for other tasks like generative modeling and sentiment analysis.

### What was easy

Many experiments included descriptions of the neural network architectures and graphs showcasing performance, giving us a clear benchmark to compare our results.

### What was difficult

Data for the human body temperature experiment was not available. Proper implementation details were not given for initializing the weights in neural networks with snake and using snake with RNNs.

### Communication with original authors

One author, Liu Ziyin was contacted to provide the dataset used for the human body temperature experiment, elaborate upon the implementation of variance correction during weight initialization and provide his implementation of RNN using snake. He provided the Github link to his code for the human body temperature, market index, and extrapolation experiments. He also provided an explanation on how to implement variance correction. While the code for the RNN implementation using snake activation was not made public, he provided a screenshot of the same.

Submitted to ML Reproducibility Challenge 2020. Do not distribute.

# 1 Introduction

Periodic functions underpin several key phenomena in both human society and natural science. The world's daily and yearly cycles [2], the human body's intrinsic biological clock [8; 9], and stock market fluctuations [6; 7] are just some examples of phenomena that can be modeled using periodic functions. In scientific scenarios, these models can be used to predict how a system will evolve. If a problem is known to be periodic, it can be solved easily e.g. in the Fourier space. However, in most cases, it is not known a priori if a problem is periodic or contains a periodic component. Therefore, a model must be flexible enough to learn both periodic and non-periodic functions.

While deep neural networks are excellent tools in interpolating between existing data, standard versions of networks are not suited for extrapolation beyond the training range, struggling in particular with periodic functions. Attempts have been made to address this issue such as using periodic activation functions [13; 14; 5]. Standard periodic functions such as $sin(x)$ and $cos(x)$ or their linear combinations were used as activation functions. However, it was found that they are hard to optimize due to large degeneracy in local minima [5], and the experimental results suggest that using sin as the activation function does not work well except for some very simple models, and that it can not compete against ReLU-based activation functions [4; 11; 10; 12] on standard tasks.

The original paper: (1) studies the extrapolation properties of a neural network beyond a bounded region; (2) shows that standard neural networks with standard activation functions are insufficient to learn periodic functions outside the bounded region where data points are present; (3) proposes a solution for this problem in the form of a novel activation function and its variants, and showcases its performance on toy examples and real tasks. We have tested the claims made in the original paper, replicating both the experiments displaying the failure of standard activation functions to learn periodic functions as well as the results of the novel activation function on toy and real-world tasks.

# 2 Scope of reproducibility

The authors make two key claims:

- Standard neural networks with standard activation functions are insufficient to learn periodic functions outside the bounded region where data points are present.
- Propose an activation function that learns periodic functions while maintaining the favorable optimization property of the ReLU-based activations. The novel activation is dubbed "snake":

$$snake_a(x) := x + \frac{1}{a} sin^2(ax)$$

  where $a$ is treated as a fixed parameter in initial experiments, and as a learnable parameter in a few experiments. Snake is shown to outperform standard activation functions ReLU, tanh, LeakyReLU [11], as well as more recently proposed functions such as swish [4], and sin [5].

Due to the broad and far-reaching consequences of the two claims, the original paper supports them via both theoretical justification as well as an extensive list of experiments which range from performance on toy datasets to real world applications. We have exhaustively replicated the original list of experiments, and have conducted a few additional experiments of our own, using the proposed activation function in a Deep Convolutional Generative Adversarial Network (DCGAN) to generate images of handwritten digits and in Long Short Term Memory (LSTM) networks for sentiment analysis.

# 3 Methodology

The code used by the authors had not been made public at the time we started working on re-implementing the paper. That meant we reproduced all the results in the paper from scratch relying on the descriptions of the neural network architecture and a link or description of the dataset. The descriptions were brief but sufficient such as "feedforward neural network with 2 hidden layers, both with 64 neurons" for the Body Temperature Prediction experiment and "4-layer feedforward network with $1 \rightarrow 64 \rightarrow 64 \rightarrow 1$ hidden neurons" for the Financial Data Prediction experiment. In the case of experiments that utilized large standard networks such as ResNet18, default values were assumed for hyperparameters unless otherwise specified in the original paper. Besides the model implementations, we were also required to make $a$ a learnable parameter in snake. No implementation details were provided for this variation.

### 3.1 Model descriptions

Models used in the original paper included fully-connected, feed-forward neural networks with different architectures for the various experiments. Larger standard models such as ResNet18 were also used. The authors of the original paper had not made their code available and we had to implement all the models ourselves.

### 3.2 Datasets

The datasets used were:

The data used in the extrapolation experiments are directly sampled from periodic functions such as $sin(x)$. Some experiments dealt with standard datasets such MNIST and CIFAR-10. Data for the real-life datasets had to be downloaded:

- Daily data from 1995-1-1 to 2020-1-31 of Wilshire 5000 Total Market Full Cap Index: Downloaded from link provided in the original paper: `https://www.wilshire.com/indexes`
- Average weekly temperature evolution in Minamitorishima, an island south of Tokyo (longitude: 153.98, latitude: 24.28) after April 2008: Downloaded from link provided in the original paper: `https://join.fz-juelich.de/access`
- Patient body temperature: Made available by the authors upon request
- IMDB Reviews Dataset used for our additional sentiment analysis experiment: Downloaded from `https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews`

### 3.3 Hyperparameters

Different experiments included varying levels of detail with respect to hyperparameters. Many experiments provided an overview of the neural network architecture (i.e. "4-layer fully connected neural network") but not other hyperparameters, such as batch size, loss function, or learning rate. In cases where information was missing, assumptions had to be made, with some trial-and-error required to obtain a close approximation of the original result.

### 3.4 Experimental setup

The entire codebase has been uploaded to GitHub and is publicly available: `https://github.com/mayurak47/Reproducibility_Challenge`

The experiments were run locally as well as on GPU enabled session on Google Colab.

### 3.5 Computational requirements

Many of the experiments could be run locally, not requiring more than a few minutes to train. Training a ResNet18 on CIFAR-10 with six activation functions for 50 epochs required roughly 6 hours on a Tesla T4 GPU on Google Colab. Our additional experiments on training a GAN and an LSTM required roughly 2 hours each on the same hardware.

## 4 Results

Wherever possible, the claims of the original papers were tested and in each case, we were able to reproduce the original results. The list of experiments that we reproduced is listed below.

### 4.1 Extrapolation experiments on analytic functions

Neural networks with a single hidden layer consisting of 512 neurons are trained on data sampled from four different analytic functions using the ReLU and tanh activation functions. It is observed that the extrapolation of neural networks depends on the activation function used. When ReLU is used, the extrapolation diverges to $\pm\infty$. When tanh is used, the extrapolation levels off. The authors formally prove these observations and conclude that neural networks using these activation functions cannot learn to extrapolate periodic functions.

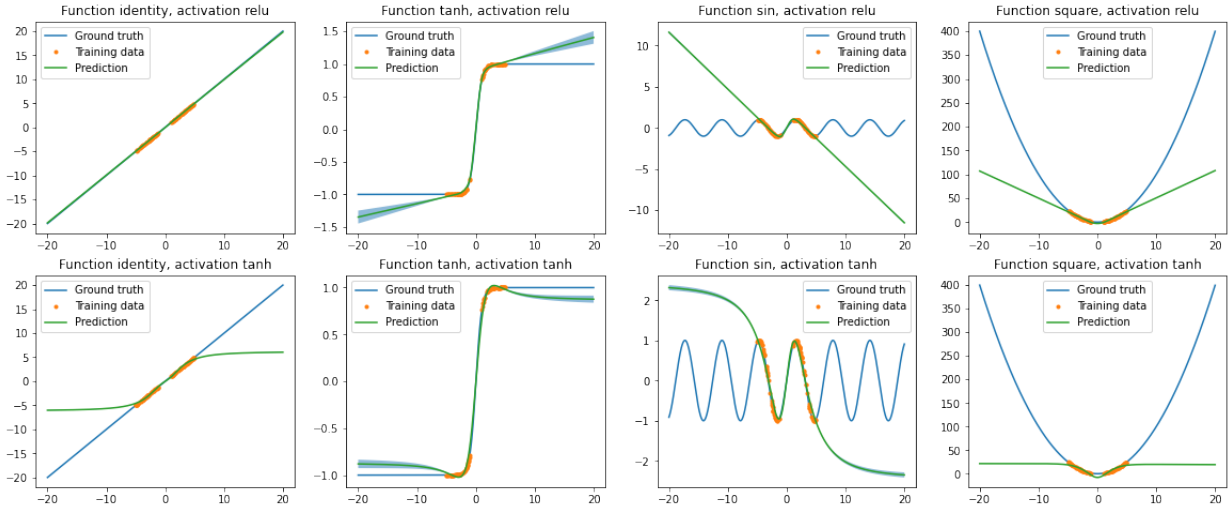


Figure 1: Regressing analytic functions with neural networks having the specified activation function

### 4.2 Applicability of proposed method

It is first demonstrated that the snake activation function is easier to optimize than other commonly used baseline periodic activation functions like $sin(x)$ and $x + sin(x)$ by training a 4-layer fully-connected neural network is trained on the MNIST dataset.

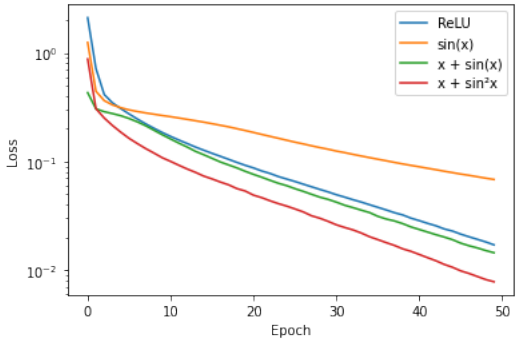


Figure 2: Optimization of different activation functions on MNIST

It is then shown that snake is able to regress the periodic function $sin(x)$. While all activation functions learn the training data (Fig. 1), only snake is able to capture the periodic behavior of $sin(x)$.

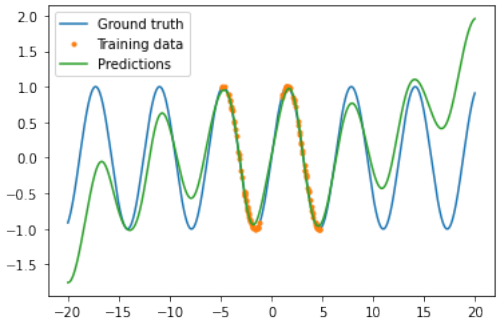


Figure 3: Regressing $sin(x)$ using the snake activation function

### 4.3 Applications

Multiple experiments are conducted to illustrate the performance of snake on a range of tasks.

ResNet18 [15], with 10M parameters, is trained on the CIFAR-10 dataset. This is a 10-way image classification task. The ReLU activation functions in the architecture are replaced with the specified activation, and the network is trained for 100 epochs. The LaProp optimizer [16] is used; the learning rate is $4e-4$ for the first 50 epochs and $4e-5$ for the next 50. The test accuracy achieved by snake is in line with that of the other standard activation functions.

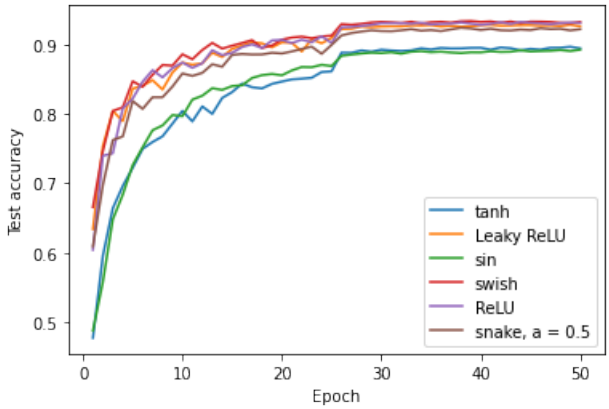


Figure 4: Test accuracy of ResNet18 with different nonlinearities

We show the utility of snake in two real-life problems. The two tasks are predicting the evolution of temperature in Minamitorishima island in Japan, and the modeling the body temperature of a patient. In the Minamitorishima experiment, the parameter $a$ was made learnable. In both cases, snake is the only activation function that makes meaningful extrapolation and predictions.

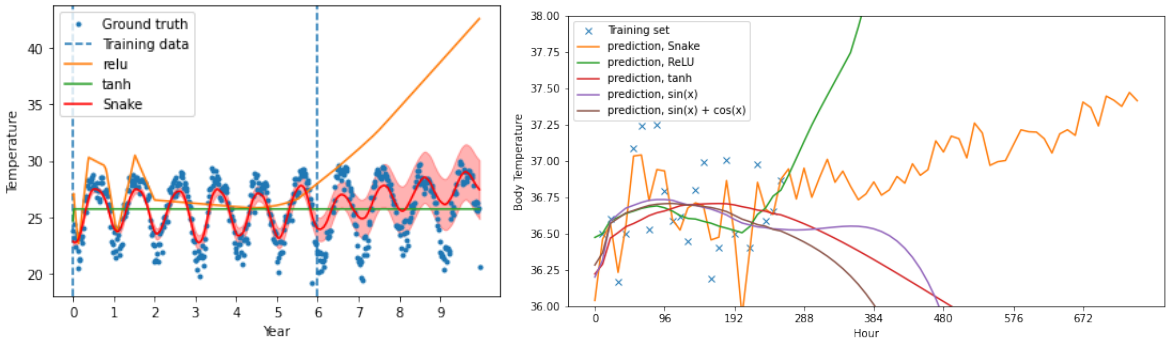


Figure 5: Prediction of atmospheric temperature (left) and body temperature (right) by neural networks using different activation functions

The snake network correctly learns the periodicity of the atmospheric temperature dataset, even though the amplitude is slightly off, and correctly infers that body temperature is roughly 37°C.

Another regression problem the authors used to demonstrate the working of snake is that of financial data prediction. The data used is from the Wilshire 5000 Total Market Full Cap Index, considered representative of the economic trend. The model, which was trained using data from 1995 to 2020-1-31, before COVID-19 impacted the world economy, predicted an economic slowdown in 2020. This might be due to the cyclic nature of world markets, which the model was able to capture.

We also use this dataset to gain insights into how the snake activation function learns (Fig. 7). Observing the predictions made at various points in the training process, we notice that first, the features learned are mostly linear, low frequency features are then learned, and high-frequency features are learned at the later stages of training.

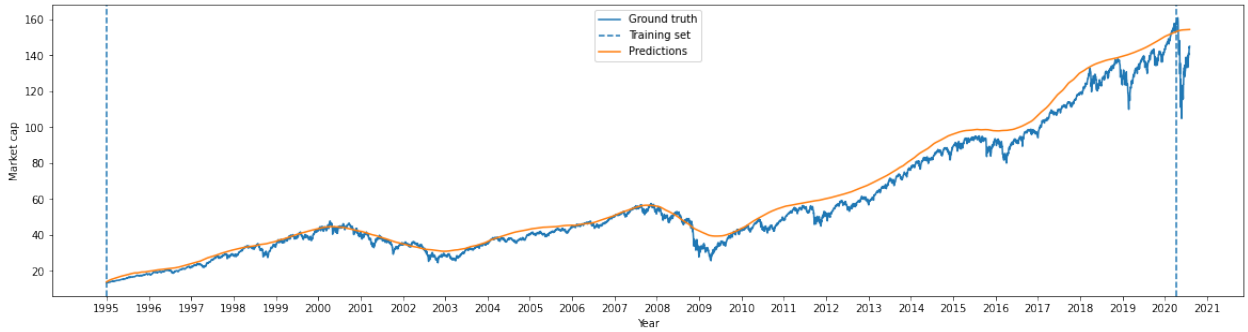


Figure 6: Predicting the Wilshere 5000 index

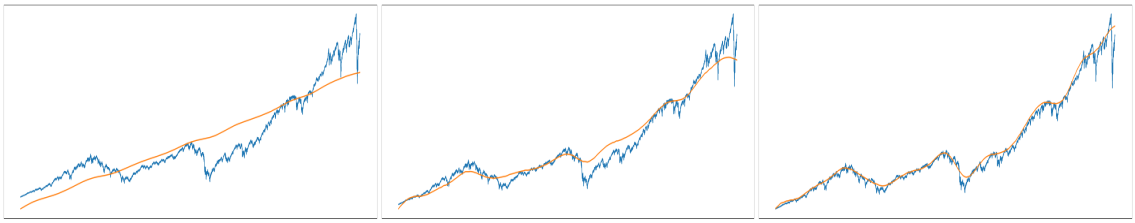

Figure 7: Predictions made by the model after 10, 20, and 50 epochs of training on the Wilshere 5000 index

The performance of a snake feedforward network and a recurrent neural network (RNN), typically used for time-series prediction, are compared. The task is to learn the function $sin(0.1x)$, with Gaussian noise $\sigma$ added, for $T$ = 300 timesteps. The first 200 are used for training, while the last 100 are used for testing.

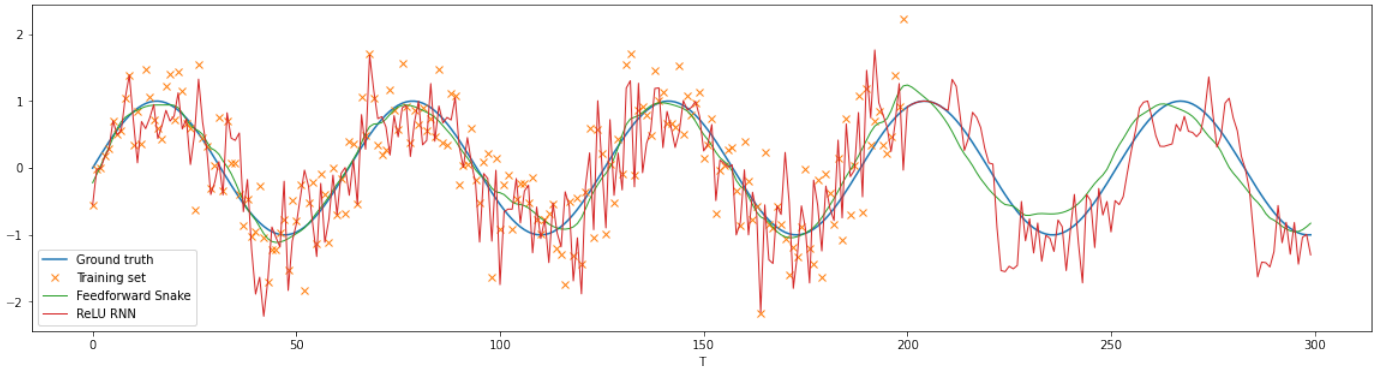


Figure 8: Predictions of a feedforward network with snake activation and a conventional RNN on $sin(0.1x)$

It is seen that because of the noisy data, even the predictions of the RNN are noisy, with a high generalization loss. The feedforward network, on the other hand, almost perfectly learns the function.

Also, the time taken by the snake network to regress the function is orders of magnitude lower than the time taken by the RNN, suggesting that snake networks may be more effective in modeling data that is known to be periodic.

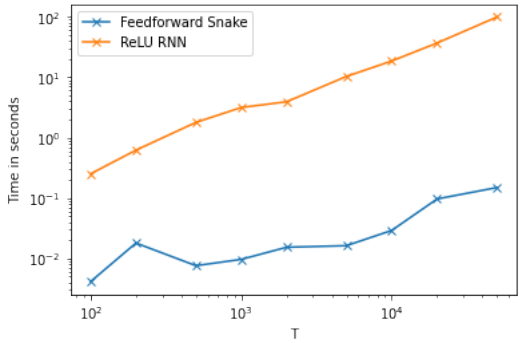


Figure 9: Time taken for a single epoch of training by an RNN and a snake feedforward network to train on $T$ timesteps of $sin(0.1x)$

### 4.4 Effect of $a$

In a series of experiments, the authors depict the effect the parameter $a$ has on the learning process. We reproduce one of these experiments for brevity. Simple neural networks are trained on the sinusoidal function $sin(x) + sin(4x)/4$.

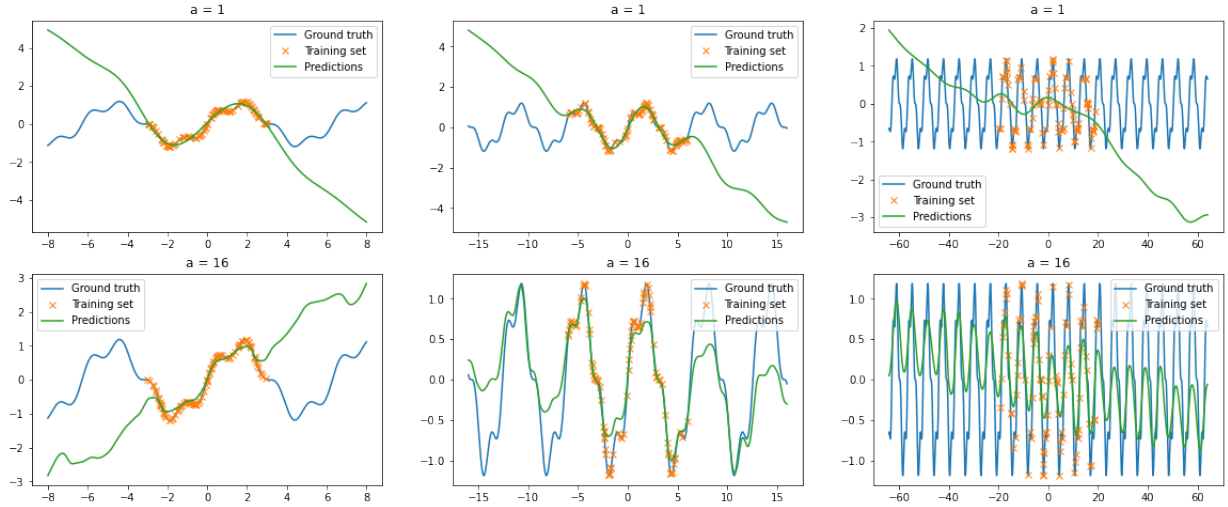


Figure 10: Features learned by snake neural networks at different $a$

It is seen that larger $a$ encourages the model to learn features with higher frequency. With $a = 1$, the higher frequency modulation is considered noise, while the $a = 16$ model learns both the signals.

### 4.5 Results beyond original paper

The original paper demonstrated the ability of neural networks with the snake activation function to learn periodic functions and that the performance on everyday tasks like image classification is similar to that of conventional activation functions. We extend this study to more subfields of deep learning.

We train a deep convolutional generative adversarial network (DCGAN) [17] to generate samples of the MNIST dataset. All the activations in the generator and discriminator subnetworks are replaced with the specified nonlinearity. We see that while the initial training is slow for the snake GAN, it produces reasonably realistic samples after training is complete. $a$ was a learnable parameter in this experiment.

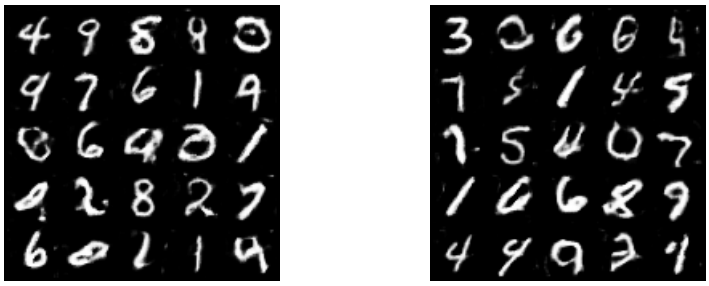

Figure 11: Samples output by snake GAN (left) and LeakyReLU GAN (right) after training for 50 epochs

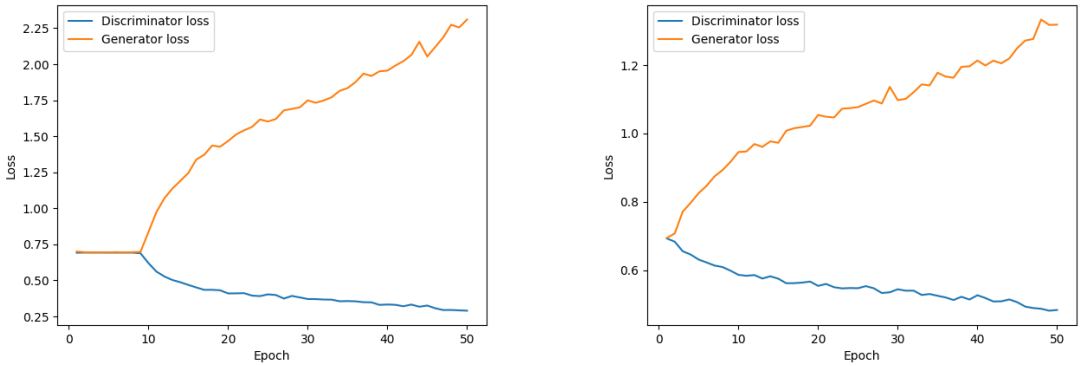


Figure 12: Losses observed over the course of training snake GAN (left) and LeakyReLU GAN (right)

Finally, we use the snake activation function in a Long Short Term Memory (LSTM) network for sentiment analysis on the IMDB movie reviews dataset. The typical tanh activation used to output the value $h_t = o_t * tanh(C_t)$ is replaced by the snake activation, such that $h_t = o_t * snake(C_t)$. We observed that the snake LSTM network took much longer to converge to the optimum. Also, in many cases, the network got stuck in local minima, necessitating a restart of training.

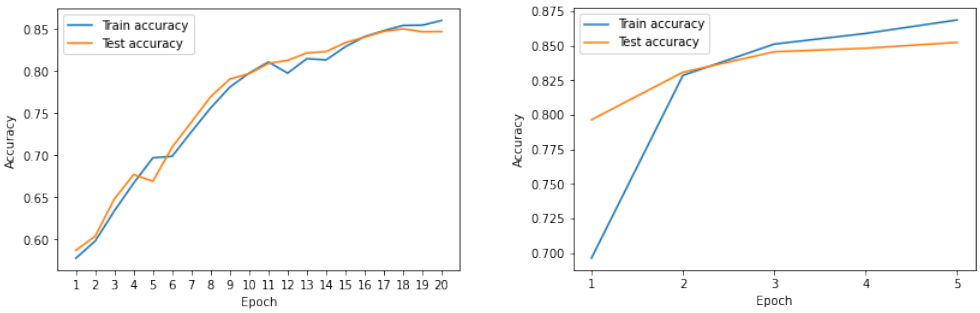


Figure 13: Training and testing accuracies versus epochs for snake LSTM (left) and tanh LSTM (right)

## 5 Discussion

As the authors had not made their code available and only included brief descriptions of the network architectures used in their experiments, exact replication of their experimental results was not possible. However, the nature of the experiments meant that only the relative performance of snake in comparison to other activation functions was of interest. We were able to uphold the claim that standard neural networks with standard activation functions are insufficient to learn periodic functions outside the bounded region where data points are present as well as the claim that the proposed activation function performs as well as standard activation functions such as ReLU, tanh, leaky-ReLU over a wide range of tasks, by replicating the experiments in the original paper and conducting some additional ones

ourselves. Future work could focus upon providing theoretical justifications for the behavior of snake and developing more suitable optimization algorithms.

### 5.1 What was easy

Detailed description of the neural network architecture used for experiments such as training on the MNIST dataset and human body temperature was provided, allowing us to replicate the experiments closely. Links to datasets for all experiments, barring one, were also included in the paper itself. An extensive appendix sections listed additional experiments comparing the performance of snake with different $a$. Every experiment was supported by graphs showcasing the performance of snake with other activation functions, giving us a clear metric with which we could compare the results of our reproductions.

### 5.2 What was difficult

The link to the original source code was not provided and we had to rely on the descriptions of hyperparameters and guesswork while attempting to exactly replicate the results. Data for the human body temperature experiment was not available. Theoretical justification for variance correction and the results of this variance correction using ResNet101 on CIFAR-10 were provided, but implementation details were not included. The section on Comparison with RNN on Regressing a Simple Periodic Function simply states that snake was deployed on a feedforward network, without any additional details of the hyperparameters used. The dataset also had to be inferred from the graphs of the results, and since white noise had been added to the data, exact replication of the experimental setup was not possible.

### 5.3 Communication with original authors

Liu Ziyin, one of the authors, was contacted to provide the dataset used for the human body temperature experiment, elaborate upon the implementation of variance correction and provide the implementation of RNNs using snake.

On being contacted, he provided the Github link to his code for the human body temperature, market index, and extrapolation experiments. He also provided an explanation on how to implement variance correction. While the code for the RNN implementation using Snake activation was not made public, he provided a screenshot of the same.

## References


[1] Liu Ziyin, Tilman Hartwig, and Masahito Ueda. Neural Networks Fail to Learn Periodic Functions and How to Fix It. *arXiv:2006.08195*, 2020.

[2] Isaac Newton *Philosophiae naturalis principia mathematica*. William Dawson & Sons Ltd., London, 1687.

[3] Jung-Hua Wang and Jia-Yann Leu. Stock market trend prediction using arima-based neural networks. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 4, pages 2160–2165. IEEE, 1996.

[4] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Swish: a self-gated activation function. *arXiv preprint arXiv:1710.05941*, 7, 2017.

[5] Giambattista Parascandolo, Heikki Huttunen, and Tuomas Virtanen. Taming the waves: sine as activation function in deep neural networks, 2016.

[6] Denise R. Osborn and Marianne Sensier. The prediction of business cycle phases: Financial variables and international linkages. *National Institute Economic Review*, 182(1):96–105, 2002.

[7] Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. Stock price prediction via discovering multi-frequency trading patterns. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 2141–2149, New York, NY, USA, 2017. Association for Computing Machinery.

[8] Norio Ishida, Maki Kaneko, and Ravi Allada. Biological clocks. *Proceedings of the National Academy of Sciences*, 96(16):8819–8820, 1999.

[9] Roberto Refinetti and Michael Menaker. The circadian rhythm of body temperature. *Physiology & behavior*, 51(3):613–637, 1992.

[10] Djork-Arne Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[11] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. *In Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[12] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[13] Adrian Silvescu. Fourier neural networks. In *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339)*, volume 1, pages 488–491. IEEE, 1999.

[14] Abylay Zhumekenov, Malika Uteuliyeva, Olzhas Kabdolov, Rustem Takhanov, Zhenisbek Assylbekov, and Alejandro J. Castro. Fourier neural networks: A comparative study, 2019.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[16] Liu Ziyin, Zhikang T Wang, and Masahito Ueda. Laprop: a better way to combine momentum with adaptive gradient. arXiv preprint arXiv:2002.04839, 2020.

[17] Radford, A., Metz, L. and Chintala, S., 2021. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. [online] arXiv.org. Available at: <https://arxiv.org/abs/1511.06434>