Challenges in chaotic time series forecasting

Maximilian Hornung

May 1, 2019

Abstract

Predicting the future development of time series is of interest in different areas of computational biology. The time series in biological systems exhibit challenging characteristics such as chaotic or stochastic behavior. In this report, time series forecasting and generation is done on several kinds of biological time series using deep neural networks. We compare classical multi layer perceptrons with recurrent and convolutional architectures. In our evaluation, we identify challenges and limitations of neural networks for these tasks, and compare different architectures of deep neural networks with regard to their performance. We conclude that the coniguration of the optimization algorithm has strong impact on the results, independent of the architecture. The nature of the noise in the data has also strong influence. We also show that a generation of stochastic time series is not possible using deterministic neural networks, because only the mean of the probability distribution can be estimated.

1 Introduction

In various kinds of diseases, it is important to choose the correct treatment at the current time point. Since the human body is too complex to be described directly using mathematics, special features are modeled with mathematical systems such as the Mackey-Glass equations for respiratory and hematopoietic diseases [MG77]. These models make use of available data about previous information in order to reduce the uncertainty under which a decision is made. This is especially important when the consequences of a decision are severe, for example the decision of breast cancer treatment after radiation. This particular problem can be addressed by using a stochastic differential equation, as done by Oroji *et al* [OOY16].

In different kinds of biological models, even chaotic behavior is observed. The Mackey-Glass equations possess chaotic behavior for certain choices of their parameters, as shown by Farmer [Far82]. That means that even small measurement errors of the initial state lead to exponentially increasing errors of the time series prediction. Because of that, it is important that the time series prediction should be as robust as possible against noise.

This gets important in particular because biological data cannot be measured without the presence of noise. In this report, we show that different kinds of noise assumptions can impact the results of time series prediction. We evaluate all our investigated

problems with regard to the popular independent and identically distributed (*i.i.d.*) random noise as well as noise from a discretized Wiener process. Since mathematical models of biological systems cannot describe their characteristics perfect and error-free, it is assumed that the latter type of noise is more realistic. But unfortunately this kind of noise increases the difficulty of time series prediction, even for simple mathematical functions, as seen in Section 3.1.

The remaining report is structured as follows. First, we perform proper time series prediction on the sinus function and show how different noise models impact the prediction ability of our network models. After that, we show how different levels of chaos impact the possibility to forecast the development of time series at the example of the Mackey-Glass time series. We show how the variance and the type of noise impacts the possibility to predict deterministic chaotic time series at the example of this time series. Last but not least, we report fundamental limitations of neural networks to approximate stochastic time series at the example of biological oscillators.

2 Methods

In our analysis, we use three different architectures and neural networks and optimize their hyperparameters for the respective use case. Since Hornik *et al.* have shown that multi-layer perceptrons (*MLP*) are universal function approximators [HSW89], we evaluate this architecture with varying number of hidden nodes.

After that, we investigate the strength of improvement of using a recurrent neural network. This type of neural network has already been applied to time series prediction [CMA94], but suffers from the vanishing gradient problem when capturing long-term dependencies in a sequence. Because of that, we decide to evaluate only the Long Short-Term Memory (*LSTM*) network architecture [HS97], which was successfully applied in various time series prediction tasks like anomaly detection [MVSA15], stock price [FK18] and protein disorder prediction [HYPZ16].

In the last years, convolutional neural networks (*CNN*s) have improved the results in image classification [KSH12] and other computer vision tasks. A *CNN* learns features from the data in a hierarchical way, for example combining pixels to edges, edges to more complex forms etc. until a high-level classification can be done. The large success in computer vision has inspired researchears in time series prediction to also apply *CNN*s [CCC16, BBO17], so we also evaluate this architecture in our analysis. The same way as images are composed of hierarchical features (e.g. a face consisting of eyes, that consist of certain edges etc.) we assume that similar hierarchies of features can be found in time series data.

The loss is measured using the classical mean squared error (*MSE*) or the root mean squared error (*RMSE*), because López-Caraballo *et al.* [LCSL$^+$16] use this error function to compare various other neural network approaches for Mackey-Glass time series prediction. This function is computed in Equation 1.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{1}$$

In order to train ours models, we apply the *Adam* optimizer [KB14]. This famous

optimization algorithm (more than 20.000 citations in less than 5 years) is commonly used to train neural networks and improves the stochastic gradient descent algorithm. Other optimization algorithms are not considered in order to keep the hyperparameter tuning feasible.

We use two different types of noise. First, we consider *i.i.d.* gaussian noise which is added to the underlying function $f(t)$ at each point of time t, as seen in Equation 2. The other type of noise is a discretized Wiener process, as given in Equation 3. This kind of stochastic process is used in various areas, for example to study Brownian motion in Physics. For more details about Wiener processes, refer to Schilling *et al* [SP14].

$$\forall t: \quad x(t) = f(t) + y, \quad y \sim \mathcal{N}(0, \sigma^2) \tag{2}$$

$$\forall t: x(t) = f(t) + y(t), \;\; y(t) = y(t-1) + y, \;\; y \sim \mathcal{N}(0, \sigma^2), \;\; y(0) = 0 \tag{3}$$

All implementations are done using the `Python` programming language in version 3.6.7. The time series data is created and loaded in the `numpy` framework in version 1.16.1. In order to train the neural networks both fast and elegant, we use the `keras` framework (version 2.2.4) with the `tensorflow` backend in version 1.13.1. The reason for this choice is the tight integration between `keras` and `numpy` that simplifies and increases the speed of our software development. Before running the experiments, the random number generator of `numpy` is set to the seed 0 to ensure reproducibility of data generation. The time series data for the biological oscillator is generated using `gillespy` [ADHP16], which only works with `Python` in version 2. For these application, the version 2.7.15 of `Python` is used.

3 Evaluation

3.1 Time series forecasting of periodic functions

In the first part of the analysis, we analyze the capability of different neural network architectures to perform time series prediction on the periodic sinus function. This task can be considered simple, because the different parts of the sine wave occur multiple times in the data. It is therefore interesting how robust the network architectures are with regard to different kinds of noise.

In this problem, we apply a large grid search over several parameters given in Table 1. All models are trained using the *Adam* optimizer and 10 time sequence entries are used to predict 1 directly consecutive point. These assumptions are made in order to keep the grid search feasible. For training, testing and validation, a range from 0 to 6π is used. The data is split equally between the datasets, the first third used as the training and the second third used as the validation set. Expressed more formally, we provide information about the last 10 time sequence points to the network and want to predict the next point, as given in Equation 4. That is, we want to estimate $x(t)$ with a function $F(x(t), x(t-1), \cdots, x(t-9))$ that is computed using a neural network.

$$\hat{x}(t+1) \approx F(x(t), x(t-1), \cdots, x(t-9)) \tag{4}$$

Parameter	Values
Number of hidden nodes	[5, 10, 20]
std (σ) of noise	[0, 0.01, 0.1]
Noise type	["iid", "Wiener process"]
Learning rate	[0.001, 0.01, 0.1]
Epochs of training	[10, 100, 500]
Neural network architecture	["MLP", "LSTM", "CNN"]

Table 1: Parameter combinations used for the grid search. Each possible value of each parameter is combined with each possible value of each other parameter. Note that the *CNN*s are evaluated with 1, 2, or 3 hidden layers of with 8 filters instead.

Architecture	Average MSE
MLP	0.00285
LSTM	0.00066
CNN	0.07583

Table 2: Average MSE over all applied neural networks aggregated by the neural network architecture type.

As we expected based on our literature review, the *LSTM* yielded lowest test error on the noise free time series data as seen in Table 2. We use the MSE here and evaluate on the test error because we are interested in generalisation capability of the neural networks, which can not be evaluated based on the training error. After investigating the high error for the *CNN* architecture, we discover that this architecture needs a lower training rate than the *LSTM* architecture to yield good results – the learning rate of 0.1 is too high and ruins the results as seen in Table 3.

Next we investigate the influence of the type of noise and the standard deviation of the noise on the results based on our grid search. The average error is strongly impacted as seen in Table 4. We see that a higher σ increases the test error. But it seems that the neural networks are less capable of filtering noise from a Wiener process, and for a high σ of this type of noise completely lose the ability to approximate the underlying sinus function on the first look. A further analysis of the used networks architectures shows that this is merely caused by inluding *CNN*s trained with learning rate 0.1. After excluding them, the test error goes down to a high but reasonable value.

In Figure 1, we see the best neural networks found using the large grid search for a noise free time series. The approximation is remarkably accurate and no deviation can

Learning rate	Average MSE (CNN)	Average MSE (LSTM)
0.1	0.227052	$5.391024 \cdot 10^{-5}$
0.01	0.000155	$8.129514 \cdot 10^{-5}$
0.001	0.000159	0.000276

Table 3: Comparison of the average MSE when training *CNN* and *LSTM* architectures. We see that the *LSTM* can be trained using higher learning rates.

Type of noise	MSE	MSE ($\sigma = 0.01$)	MSE($\sigma = 0.1$))
i.i.d noise	0.015	0.008	0.022
Wiener process	24.153	0.0524	48.254 (1.322)

Table 4: Average MSE aggregated by the type of noise and the noise standard deviation.

Noise type (standard deviation)	Net type	Epochs	Hidden Units	L.R.
no noise	MLP	500	5	0.001
i.i.d. (0.01)	LSTM	500	5	0.01
i.i.d. (0.1)	LSTM	100	5	0.001
Wiener (0.01)	CNN	500	1 layer	0.01
Wiener (0.1)	MLP	500	10	0.001

Table 5: Best performing neural network architectures for different noise settings. The *LSTM* are well suited to the *i.i.d.* distributed Gaussian noise, while the other architectures should be used for noise from a Wiener process.

be found. If we add different kinds of noise, the grid search is still able to find suitable architectures, as seen in Figure 2. In Figure 3 we demonstrate that only a few outliers caused the average error for Wiener process noise with high σ to increase so strongly.

After investigating the architectures of the best neural networks in Table 5, we conclude that *LSTM* networks are best suitable to situations in which *i.i.d.* distributed noise occurs. But also a classical *MLP* architecture performed best in presence of high Wiener noise, while for a small amount of Wiener noise, the *CNN* architecture is advantageous. From the grid search, we conclude furthermore that the default parameters in the optimization algorithms can be improved in various cases. The longest training is not always delivering the best results, which indicates the importance of the Early Stopping technique. Using the grid search on a various number of parameters, it was possible to find a suitable architecture for each of the considered situations.

3.2 Mackey-Glass time series forecasting

In order to model diseases related to dynamic respiratory and hematopoietic diseases, Mackey *et al.* proposed the Mackey-Glass equations, a kind of first-order nonlinear differential delay equations to model the number of white blood cells over time [MG77]. The solutions to these equations given in Equation 5 exhibit chaotic behavior under certain conditions [Far82]. For fixed parameters $a = 0.2$, $b = 0.1$ and $c = 10$ this system

Architecture	Noisefree data	*i.i.d.* noise	memorizing noise
MLP	0.0021	0.0249	0.0174
LSTM	0.0021	0.0127	0.5554
CNN	0.0021	0.0136	0.0234

Table 6: Validation loss of different neural network architectures for time series prediction on the sinus function stated in RMSE.

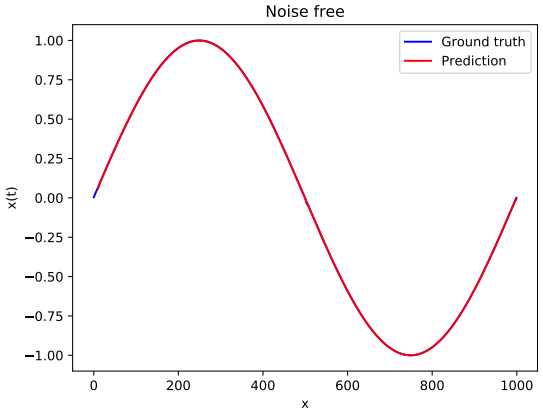

Figure 1: Demonstration of grid search results. For the noise free case, a nearly perfect approximation is achieved.

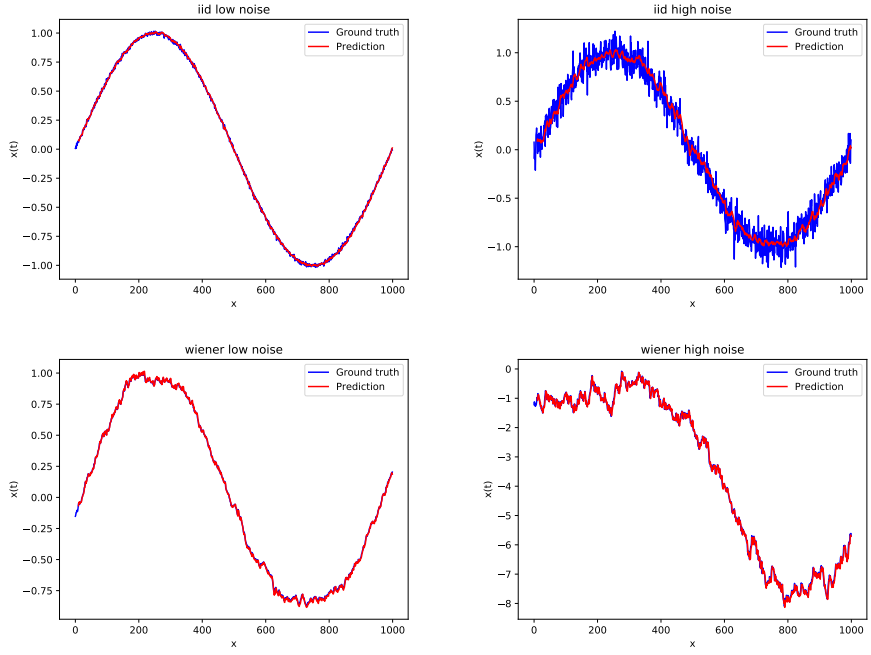

Figure 2: Demonstration of grid search results. The low noise has a standard deviation of $\sigma = 0.01$ and the high noise uses $\sigma = 0.1$, independent of the type of noise.

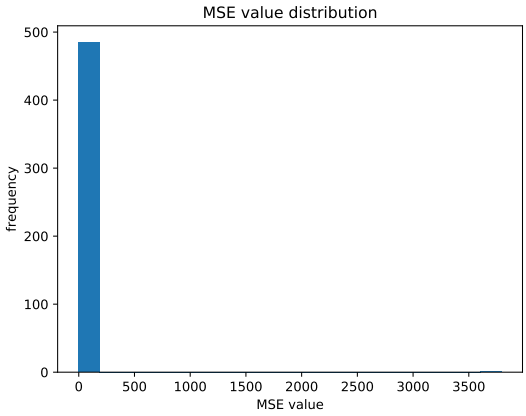

Figure 3: Demonstration of grid search results. A few networks with high test error have to be considered during the evaluation to get more robust interpretations. The majority of trained networks yield smaller test error values.

has a stable fixed point attractor for $\tau < 4.53$. With increasing delay time, the system gets less stable. For delay times of $4.53 < \tau < 13.3$ there is a limit cycle attractor whose period raises for $13.3 \leq \tau \leq 16.8$. For delay time $\tau > 16.8$ the system shows chaotic behavior.

$$\frac{dx}{dt} = \frac{a \cdot x(t-\tau)}{1 + x(t-\tau)^c} - b \cdot x(t) \tag{5}$$

By using the Euler method to discretize the Mackey-Glass time series shown in Equation 6, we can see that the chaotic behavior gets stronger for increasing $\tau > 16.8$. In Figure 4, we simulate the solution of the Mackey-Glass time series for slightly different initial conditions. It is clearly visible that for $\tau = 17$, the time series diverges slower than for $\tau = 25$. In the phase plot in that figure we can see that the trajectory of the time delay $\tau = 17$ looks still similar to the non-chaotic trajectory of the time delay $\tau = 15$. But for $\tau = 25$, the chaos is so strong such that no similarity to the non-chaotic trajectories can be recognized.

The first approach to predict the short-time behavior of chaotic time series was done by Farmer *et al.* who proposed a `local approximation` technique [FS87]. After improval of predictions using support vector machines by Müller *et al.* [MSR$^+$97], the focus in research shifted towards artifical neural networks which enabled even better predictions. Two of the latest developments are the usage of Wavelet Networks [AZ13] and particle swarm optimization [LCSL$^+$16].

$$x(t+1) = x(t) + \frac{\beta x(t-\tau)}{1 + x^n(t-\tau)} - \gamma x(t) \tag{6}$$

In accordance with the approach used by López-Caraballo *et al.* [LCSL$^+$16], we predict $x(t+6)$ based on the information of the time series points in $x(t)$, $x(t-6)$,

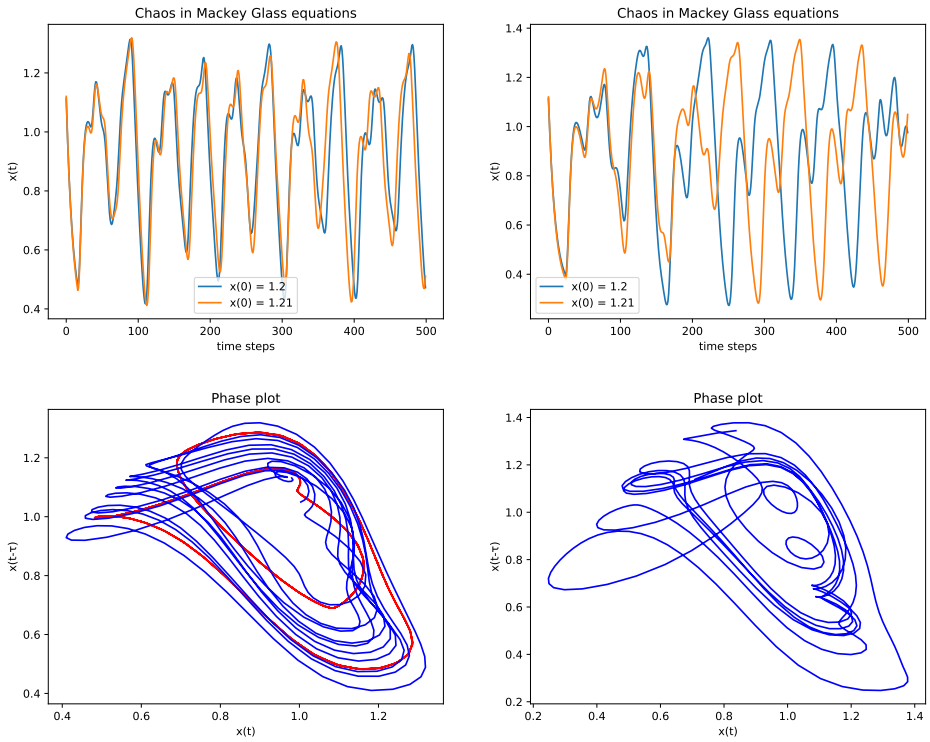

Figure 4: Influence of time delay parameter τ on the chaotic behavior of the Mackey-Glass equation. The top left side of the figure uses $\tau = 17$, the right side $\tau = 25$. The phase plot for $\tau = 17$ shows additionally the phase plot for $\tau = 15$ to visualize the similarity with a time series which is not chaotic, but has a limit cycle attractor instead.

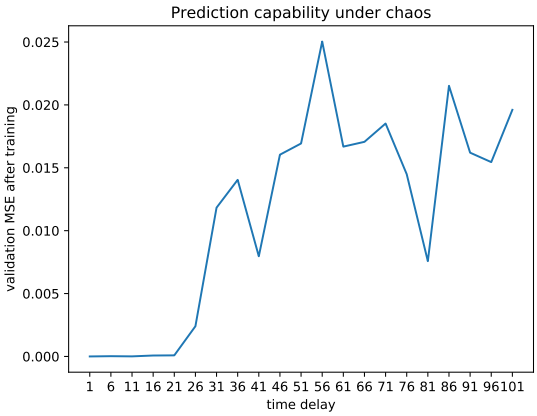

Figure 5: Validation error of the *LSTM* network for different time delay values τ of the Mackey-Glass equation. The error increases strongly for values of $\tau > 20$, indicating a strong chaotic behavior from this point.

$x(t-12)$, and $x(t-18)$. We investigate how an increase in chaotic behavior impacts the prediction capability of the different neural network architectures. The results for the *LSTM* are depicted in Figure 5. For this plot, a 2-layer LSTM architecture is used, using 10 hidden nodes in the LSTM layer, followed by a feedforward layer to sum up for the output value. The first 500 time points of the time series are used for training, the 500 following points for validation. Additional 500 points are not used during the training procedure at all and form the test set. For training our models, we train the models for 100 epochs. The learning rate of 0.01 for the Adam optimizer, because it outperforms the default learning rate in this situation.

It can be seen that for non-chaotic solutions of the macke-glass time series the model is able to predict precisely, and that for values of $\tau > 20$ the chaotic behavior is strong enough to impact the prediction. We explain the low error for $\tau = 21$ in the ability of the network to compensate for the amount of chaos up to this point.

In the next step, we compare our *MLP*, our *LSTM* and our *CNN* architectures with the results stated by López-Caraballo *et al.* [LCSL$^+$16] in Table 7. We compare our results with a linear model, a cascade correlation neural network and a radial basis function (*RBF*) neural network additional to their approach to combine an artificial neural network (*ANN*) with particle swarm optimization (*PSO*). By using the *ReLU* (Rectified Linear Unit) activation function and two hidden layers with 32 and 16 nodes, respectively, we outperform the backpropagation network reported by López-Caraballo *et al.* with our MLP architecture. From our considered models, the *CNN* achieves the lowest RMSE value and allows for accurate prediction, both for validation and for test data as seen in Figure 6. This *CNN* uses three convolutional layers with 1D convolutions, kernel size of 3, "same" padding and 8 filters each. A classical feedforward layer without activation function is used to extract the output value for the time series prediction. We avoid overfitting successfully, since the test data of the chaotic time

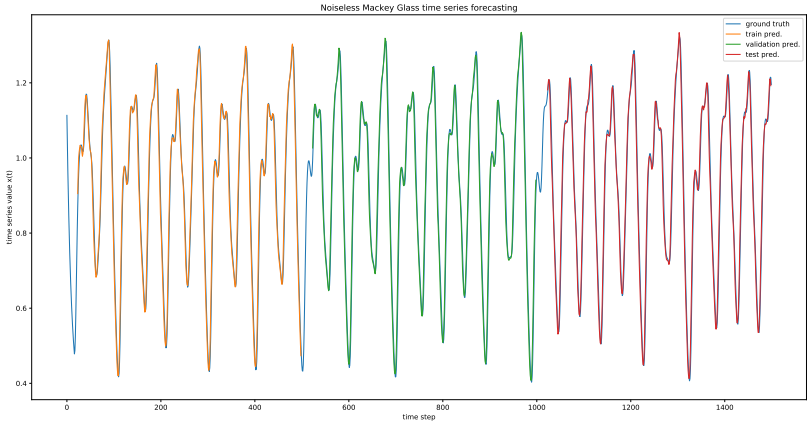

Figure 6: The prediction of the noisefree Mackey-Glass time series using a *CNN* with 3 hidden layers. The precise predictions for training, validation and testing points show the accurate approximation of the chaotic time series.

Method	$RMSE_{x(t+6)}$
Linear model	0.5503
Cascade correclation NN	0.0624
proposed LSTM	0.0131
proposed MLP	0.0125 (vs. 0.0262 [LCSL$^+$16])
RBFNN	0.0114
proposed CNN	0.0075
ANN + PSO [LCSL$^+$16]	0.0053

Table 7: Mackey-Glass time series forecasting results using different neural network architectures.

series was never presented to the neural network, but is predicted accurately.

The *LSTM* employs 10 LSTM nodes, followed by a feedforward layer to sum up the results. A variety of other layer configurations has been applied on the problem, but all consistently perform worse than the other neural network architectures. We assume that the chaotic behavior of the time series confuses the *LSTM* which is trying to account for temporal dependencies between the time series points.

Now we apply noise on the Mackey-Glass time series to account for the imperfect data typical for biological measurements. The results for these experiments are stated in Table 8. We see that an increased standard deviation of the added noise increases the test error, measured in RMSE. This applies to all neural network architectures.

We interpret the results of Table 8 as the neural networks are capable of dealing with a certain amount of noise. For example the *CNN* has approximately 9 times higher error rate when increasing the standard deviation of the Wiener process from

Architecture	noise type	σ	RMSE
CNN	iid	0.01	0.0196
CNN	iid	0.1	0.0806
CNN	wiener	0.01	0.0242
CNN	wiener	0.1	0.2172
MLP	iid	0.01	0.0249
MLP	iid	0.1	0.0949
MLP	wiener	0.01	0.0348
MLP	wiener	0.1	0.2162
LSTM	iid	0.01	0.0432
LSTM	iid	0.1	0.0862
LSTM	wiener	0.01	0.0347
LSTM	wiener	0.1	0.2044

Table 8: Results of the Mackey-Glass time series forecasting using different types of neural networks, noise levels and types.

0.01 to 0.1.

But more interestingly, the negative effect of the noise is stronger if the noise is memorizing, i.e. if the effect of the noise in time step t impacts the time series in time step $t+1$. We explain this with an increase in chaotic behavior induced by this type of noise to the Mackey-Glass time series. Since chaotic time series by definition deviate exponentially based on small variations in the initial conditions, it seems logical that this divergence is increased by introducing perturbations of the time series in every time step.

3.3 Biological oscillator time series forecasting

The next section deals with time series generation of biological oscillators, as described by Novak *et al* [NT08]. For different aspects of cell physiology, e.g. the DNA synthesis, this kind of oscillators can be observed. Novak *et al.* show that oscillations in simple ODEs can be produced by incorporating an explicit time delay [NT08], similar to the Mackey-Glass equations [MG77].

Recently, Strömstedt *et al.* analyzed the capability of neural network architectures to model stochastic time series [SSH18] at the example of time series produced according to the characteristics of Novak *et al* [NT08]. In this section, we reproduce their results for the LSTM architecture and state fundamental limits for neural networks to approximate stochastic time series.

The `gillespy` framework is used to model a stochastic reaction system with 2 states, described in Equation 8. One protein X is produced more the less of the other protein Y exists. X and Y are both continuously reduced but Y gets reduced stronger for smaller values of Y in a non-linear way. A set of 6 parameters $(k_{dx}, k_t, k_{dy}, a_0, a_1, a_2)$ is used to parametrize the system.

Processor	Intel(R) Core i5-7200U CPU @ 2.50GHz
RAM	7859 MB

Table 9: Hardware configuration used in experiments.

$$\frac{dX}{dt} = \frac{1}{1+Y^2} - k_{dx} \cdot X \tag{7}$$

$$\frac{dY}{dt} = k_t \cdot X - k_{dy} \cdot Y - \frac{Y}{a_0 + a_1 \cdot Y + a_2 \cdot Y^2} \tag{8}$$

The motivation for using neural networks to generate time series is the computational expensiveness of analytical solutions like the Gillespie algorithm [Gil77]. The runtime of an experiment on our hardware (see Table 9 for details) takes more than 6 seconds runtime to simulate this biological oscillator with two states for 100 time steps (yielding only one trajectory of the stochastic system). This makes grid search approaches on the large number of parameter combinations intractable. Using the 2-layer LSTM approach, predictions for short time series can be done in less than 0.2 seconds on our hardware.

The goal here is to train a LSTM based model to convert a set of input parameters into a time series. The first LSTM layer takes 7 input values and transforms them into n sequences of 7 values, with n being the desired length of the output sequence. The second LSTM layer takes this intermediate result and converts it to a sequence of length n.

The analysis of the resulting time series as depicted in Figure 7 shows the general limitation of using a deterministic neural network for predicting a stochastic time series. For one possible parameter choice for the biological oscillator, the analytical solution was computed under 100 (stochastic) trajectories. After that, the network was trained to predict the time series based on the (unchanged) parameters. We see that the short-term predictions are accurate and have the same magnitude as the ground truth signal – but the more time steps are simulated, the larger is the difference between the ground truth trajectories due to stochasticity. In other words, the neural network predicts the mean of the signals and the amplitude decreases due to the increasing variance of the signal over time. It should be noted that the trends in the time series are still recognized, just with a smaller amplitude.

4 Conclusion

In this report, we investigate time series forecasting using neural networks. By applying a grid search on the time series forecasting of a continuous function, we find suitable architectures to produce accurate predictions for all considered types of noise. We discover that it is necessary to vary the configuration of the optimization algorithm w.r.t. the learning rate in order to find the optimal results. It is remarkable that even the time series forecasting for a function with noise from a Wiener process yielded good

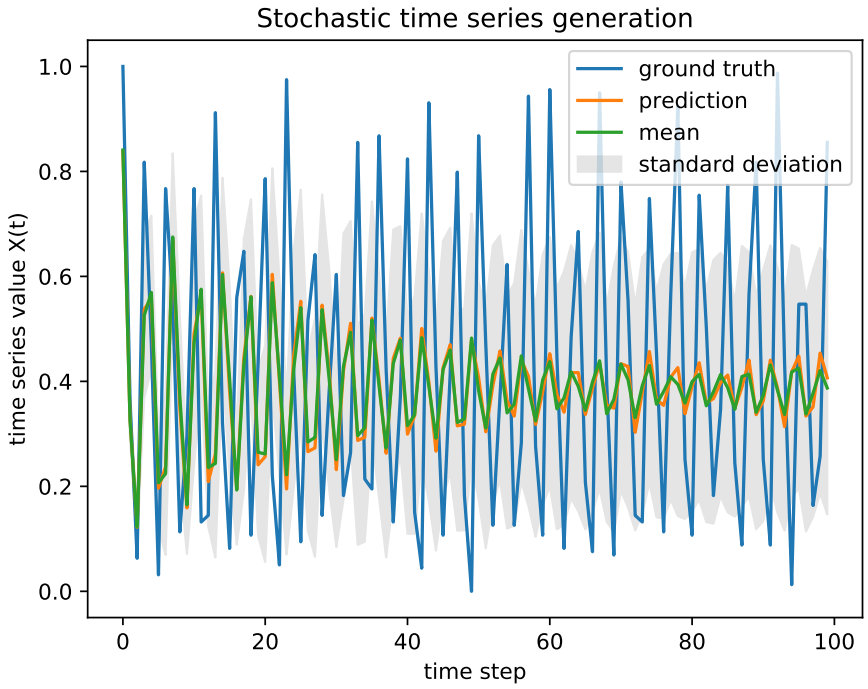

Figure 7: Prediction of stochastic time series using deterministic **LSTM** network. Sampled ground truth data are unseen for the network. Mean computed on all stochastic trajectories. The standard devation increases from the given starting point, because the stochastic trajectories change their values in different ways over time.

results, since this type of noise makes the function indifferentiable at any position. Furthermore, we applied neural networks to predict the chaotic Mackey-Glass time series. We can show that the prediction gets more challenging for the neural network with a higher delay parameter τ, which strenghtens chaotic characteristics. When applying noise from a Wiener process, the error rate of the prediction increases by approximately one order of magnitude. The Wiener process perturbates the chaotic system and influences the subsequent steps of the time series, which enlarges the chaotic characteristics of the time series. Last but not least, we show at the example of biological oscillators that it is not possible to generate stochastic time series based on a set of scalar parameters with a deterministic neural network. The neural network approximator can only predict the mean of the probability distribution generating the noise, and therefore be used to predict trends in the time series. But since the amplitude declines below the level of the original time series, some features like spikes can not be generated.

References

[ADHP16] John H Abel, Brian Drawert, Andreas Hellander, and Linda R Petzold. Gillespy: a python package for stochastic model building and simulation. *IEEE life sciences letters*, 2(3):35–38, 2016.

[AZ13] Antonios K Alexandridis and Achilleas D Zapranis. Wavelet neural networks: A practical guide. *Neural Networks*, 42:1–27, 2013.

[BBO17] Anastasia Borovykh, Sander Bohte, and Cornelis W Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017.

[CCC16] Zhicheng Cui, Wenlin Chen, and Yixin Chen. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995*, 2016.

[CMA94] Jerome T Connor, R Douglas Martin, and Les E Atlas. Recurrent neural networks and robust time series prediction. *IEEE transactions on neural networks*, 5(2):240–254, 1994.

[Far82] J Doyne Farmer. Chaotic attractors of an infinite-dimensional dynamical system. *Physica D: Nonlinear Phenomena*, 4(3):366–393, 1982.

[FK18] Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

[FS87] J Doyne Farmer and John J Sidorowich. Predicting chaotic time series. *Physical review letters*, 59(8):845, 1987.

[Gil77] Daniel T Gillespie. Exact stochastic simulation of coupled chemical reactions. *The journal of physical chemistry*, 81(25):2340–2361, 1977.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[HSW89] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[HYPZ16] Jack Hanson, Yuedong Yang, Kuldip Paliwal, and Yaoqi Zhou. Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks. *Bioinformatics*, 33(5):685–692, 2016.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[LCSL+16] CH López-Caraballo, I Salfate, JA Lazzús, P Rojas, M Rivera, and L Palma-Chilla. Mackey-glass noisy chaotic time series prediction by a swarm-optimized neural network. In *Journal of Physics: Conference Series*, volume 720, page 012002. IOP Publishing, 2016.

[MG77] Michael C Mackey and Leon Glass. Oscillation and chaos in physiological control systems. *Science*, 197(4300):287–289, 1977.

[MSR+97] K-R Müller, Alexander J Smola, Gunnar Rätsch, Bernhard Schölkopf, Jens Kohlmorgen, and Vladimir Vapnik. Predicting time series with support vector machines. In *International Conference on Artificial Neural Networks*, pages 999–1004. Springer, 1997.

[MVSA15] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *Proceedings*, page 89. Presses universitaires de Louvain, 2015.

[NT08] Béla Novák and John J Tyson. Design principles of biochemical oscillators. *Nature reviews Molecular cell biology*, 9(12):981, 2008.

[OOY16] Amin Oroji, Mohd Omar, and Shantia Yarahmadian. An ito stochastic differential equations model for the dynamics of the mcf-7 breast cancer cell line treated by radiotherapy. *Journal of theoretical biology*, 407:128–137, 2016.

[SP14] René L Schilling and Lothar Partzsch. *Brownian motion: an introduction to stochastic processes*. Walter de Gruyter GmbH & Co KG, 2014.

[SSH18] Adam Lindell Simon Strömstedt Hallberg. Black box time series modeling, 2018.