Predicting Evolving Chaotic Time Series with Fuzzy Neural Networks

Frank Z. Xing and Erik Cambria
School of Computer Science and Engineering
Nanyang Technological University
Singapore
Email: {zxing001, cambria}@ntu.edu.sg

Xiaomei Zou
Department of Computer Science and Technology
Harbin Engineering University
China
Email: zouxiaomei@hrbeu.edu.cn

***Abstract*—This work tackles the seldom discussed task of predicting chaotic time series generated by dynamic systems with evolving parameters. Representative chaotic time series produced by different system dimensions are introduced with a critical parameter linearly depending on time. The evolving character of systems are qualitatively studied by phase portraits. We assess the predictability of different fuzzy neural network (FNN) architectures on several evolving chaotic time series. Experiments illustrate that FNN models can generally better approximate evolving chaotic systems comparing to the autoregression method as benchmark. Moreover, certain FNN types, e.g., NEFCON and DENFIS, are more robust to changing system parameters. In spite of the performance, some FNN models are more vulnerable and incline to be destabilized by high order chaotic systems. This work also casts light on composing FNN structures to capture evolving characters of chaotic time series in the future.**

I. Introduction

Time series prediction comprises diversified methods to formulate a broad range of forecasting problems, such as financial modeling, weather forecasting, transportation planning and production management to name a few. Based on historical observations of a sequence of variables x_t depending on time t, a hidden process P is believed to exist, and preserves its characteristics at least for a period as the cause of patterns presented in x_t. For a long period in history, linear model is believed capable of, and applied for approximating the hidden stochastic process P. Specific techniques are developed, either autoregressive or depending on impact factors, either scalar or assuming vectorial correlation, or including doubly modeling of heteroscedasticity [7].

Whereas from the last century, a large amount of chaotic systems have been discovered. Some recognized attributes of chaotic system include the sensitive dependency on initial conditions and densely distributed periodic points [20]. These effects, along with the inevitable noise during data collection, make the long term prediction of a chaotic system impossible. However, by estimating Maximal Lyapunov Exponent (MLE), many empirical studies suggest that chaos is ubiquitous in real life time series, such as stock index [27], exchange rate [6], and climate change [10]. This fact proposes the necessity for modeling chaotic time series. A popular method to model chaotic systems is to reconstruct the original state space and adapt to the local quasi-linearity with a linear system [3], [4].

If we further expand this approximation idea to a higher dimension, the problem of adaptating to a nonlinear system will be equivalent to another regularly employed method named Fuzzy Neural Network (FNN). FNN is a rather inclusive terminology that covers several variants of combination of fuzzy logic and Artificial Neural Network, such as linguistic fuzzy [28], type-2 fuzzy [15], and high order fuzzy [1], [5]. Due to the fact that fuzzy logic is good at representing structural knowledge and embedding experience with linguistic rules, FNN is surmised to combine and take advantages of both the learning characters of neural networks [26], [29] and the interpretability of fuzzy logic [35]. Previous research [22] also indicate that FNN can better depict the statistics of chaotic systems.

The contrast between the satisfying performance reported in the literature and the constrained predicability of FNNs in practice leads us to question whether the real life time series can be formulated as an observation of static chaotic systems. In other words, the hidden chaotic process P, which is often assumed to be a set of differential equations with physical meaning, may not be time independent. Therefore, we investigate some computer generated time series depending on an evolving setting of equation parameters. The reason for using this setting is that we believe the changing of parameter is more frequent than of the paradigm itself.

The main contribution of this work is that we investigate the impact of evolving parameter on the predicting performance of different type of FNN architectures. This helps to better understand the dynamics of FNN learning process. Furthermore, the influence of system dimension on predictability is also never explicitly reported before to the best of our knowledge. We discover this phenomenon through conducting experiments on Fuzzy Inference Systems.

In this proposed work, we formulate the problem of predicting evolving chaotic time series. Paradigm shift is not discussed for the sake of conciseness. The rest of this paper is organized as follows: Section II provides the conception of evolving chaotic time series and an example of the archetypal chaos derived from logistic map; Section III introduces some fundamental architecture types of FNN, their working mechanisms and learning algorithms; Section IV discusses the predicting performance of FNN on some evolving benchmark chaotic time series. Comparison with autoregression models is presented at the same time; Section V summaries.

II. Evolving Chaotic Time Series

Chaotic time series [21] can either take the discrete form as $x_{n+1} = f(x_n)$, or continuous form as $\frac{dx(t)}{dt} = F(x(t))$. If x is a scalar representation, this time series can also be regarded as one dimensional projection of some certain aspects from chaotic systems, which are featured by infinite unstable periodic orbits. This character brings out extremely difficulty in analyzing patterns of chaotic time series. After a small perturbation, the future values for the same system deviate fast. Fig. 1 provides a deviation example for time series generated by the same chaotic system with a less than 5% random error on timestep 50. The values after timestep 250 are scarcely relevant.

Previous studies, for instance [8], usually indicate reconstruction of state space as a necessary step. Some properties, such as trajectory topology and strange attractors, and statistics, such as Lyapunov exponent, Kolmogorov entropy and fractal dimensions, are believed to be preserved in both embedded space and the real phase space with a subtle selection of parameters. According to Takens [32], embedding dimension m should theoretically satisfy $m \geqslant 2d+1$ to enable a well-formed reconstruction, where d denotes the dimension of strange attractor of original dynamic system. Though in practice, neither identifying attractor nor calculating system dimension is easy.

We define evolving chaotic time series as chaotic time series that take a set of parameters that are functions of time t. Consider logistic map as a simple polynomial mapping case that can generate complicated behavior,

$$x_{t+1} = r(t)x_t(1 - x_t)$$

It produces chaotic time series for the single parameter r in range $3.57 < r < 1 + \sqrt{8}$. If r increases linearly along time, while stay in this chaotic range, for example, $r(t) = 3.6 + 0.002t$, the phase portrait of x will become more complicated. Fig. 2 gives a comparison between phase portrait of a static r and an evolving r. We can notice from Fig. 2 that the moving trajectory of attractor is not linear (parabolic). For the remainder of this section, we investigate into three well-known chaotic systems order by dimensionality. Details about how the evolving chaotic time series are generated are described.

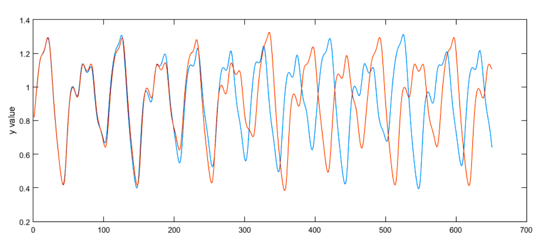

Fig. 1. Deviation of future values after a small perturbation on step 50

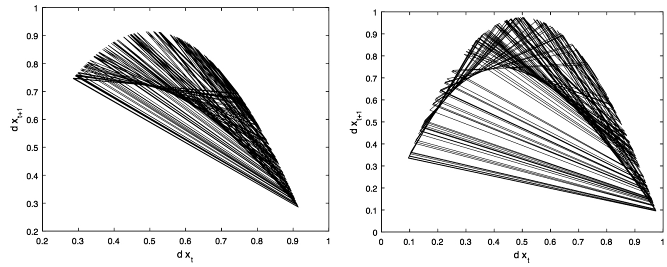

Fig. 2. Phase portrait of logistic maps, to the left with a static parameter, to the right with a linearly evolving parameter

A. Duffing Chaotic Time Series

Duffing equation is a nonlinear second-order differential equation used for modeling damped and driven oscillators, which do not obey Hooke's law.

$$\frac{d^2x(t)}{dt^2} + 2\gamma\frac{dx(t)}{dt} + \alpha x(t) + \beta x^3(t) = \delta\cos(\omega t)$$

Moreover, this equation is occasionally formulated as simultaneous equations to exhibit physical meanings more explicitly,

$$\begin{cases} \dot{x_1} = x_2 \\ \dot{x_2} = -\alpha x_1 - 2\gamma x_2 - \beta {x_1}^3 + \delta\cos(\omega t) \end{cases}$$

here x_1 stands for the length of a spring, x_2 is the first order derivative of x_1 that stands for velocity, γ denotes the friction coefficient for the spring, δ represents the exogenic driven force. This expression also makes constructing training samples convenient, as the discrete form $(x_t, y_t) = (x_2(t-1), x_1(t-1), t, x_2(t))$. The parameter settings to generate a chaotic Duffing system are discrete. The method to determine exact parameter values and to truncate initial transient is called Poincaré selection, which cannot be elaborated here. We consider a simplified case with parameters other than δ fixed: $\alpha = -1, \beta = 1, \gamma = 0.05, \omega = 1.4$. Fig. 3 shows the phase portrait of $(\dot{x_1}, \dot{x_2})$ when $\delta = 0.35$. Let $\delta(t) = 0.35 - 0.01t$, the system will produce evolving chaotic time series.

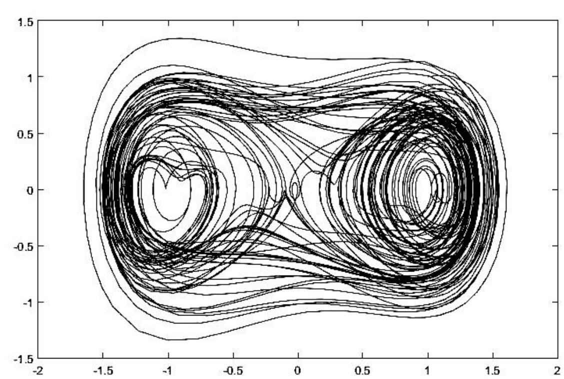

Fig. 3. Phase portrait of chaotic Duffing system ($\delta = 0.35$)

B. Mackey-Glass Chaotic Time Series

The Mackey-Glass (MG) Differential Equation is one of the most referred benchmark nonlinear time-delay system that generates chaotic time series with the following parameter: $\beta = 0.2, \gamma = 0.1, n = 10$, and $\tau > 16.8$,

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

Let $\tau(t) = 17 + \lfloor 0.01t \rfloor$, the system will produce evolving chaotic time series. The fourth-order Runge-Kutta method is used to find the numerical value for discrete integer points. Notation $\lfloor x \rfloor$ stands for the largest integer less than or equal to x, so periodic delay τ will jump after equal time interval.

C. Lorenz Chaotic Time Series

The Lorenz system is a famous climate model consists of three ordinary differential equations (ODE).

$$\begin{cases} \dfrac{dx(t)}{dt} = \sigma[y(t) - x(t)] \\ \dfrac{dy(t)}{dt} = x(t)[\rho - z(t)] - y(t) \\ \dfrac{dz(t)}{dt} = x(t)y(t) - \beta z(t) \end{cases}$$

It is derived from Navier-Stokes equation, which is used to describe fluid mechanics. Lorenz firstly used the parameter setting $\sigma = 10, \beta = 8/3, \rho = 28$ to exhibit chaotic behavior [10]. It is then testified when $\rho \geqslant 25$, chaotic phase portrait (Fig. 4) with two strange attractors is observed.

Let $\rho(t) = 25 + 0.1t$, the system will produce evolving chaotic time series. Especially, we find out that not any $\rho(t) \geqslant 25$ can lead the system to chaos. When $\frac{d\rho(t)}{dt}$ is a very large number, the Lorenz system will escape from chaotic state and fall into a periodic movement that resembles a limit cycle.

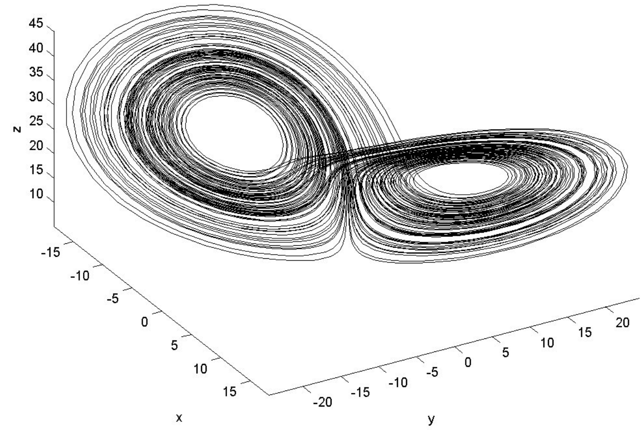

Fig. 4. Phase portrait of chaotic Lorenz system ($\rho = 28$)

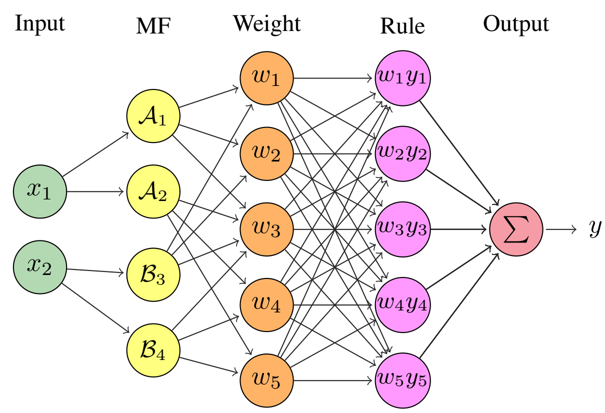

Fig. 5. Architecture of Artificial Neuro-Fuzzy Inference System

III. Fuzzy Neural Networks

Fuzzy Neural Networks, or Neuro-Fuzzy Systems, are a group of models that apply fuzzy logic and neural network together. The idea behind fuzzy logic is that, by providing a set of linguistic rules with the form of:

$$\begin{aligned} &\textit{IF } x_1 = \mathcal{A}_{1j} \textit{ and } x_2 = \mathcal{A}_{2j} \textit{ ... and } x_n = \mathcal{A}_{nj} \\ &\quad \textit{THEN } y_j = f(x_1, x_2, ..., x_n) \end{aligned}$$

the fuzzy aggregation of y_j can approximate nonlinear function $y = f'(\boldsymbol{x})$. Traditionally, these rules in fuzzy control systems are manually tuned by experts. However, rules can also be produced from a learning perspective.

Depending on the stage where neural networks are used, FNN can either be cooperative, concurrent, or hybrid [33]. For cooperative FNN, neural networks are removed once the fuzzy rules are generated, while for concurrent FNN, they take the output from each other.

Only hybrid FNN is a thorough combination of fuzzy logic and neural network in the strict sense. Therefore, we will investigate into four different while popular types of hybrid FNN, as well as their performance on predicting evolving chaotic time series. For the convenience of comparison, we focus on the basic version of each type.

A. Artificial Neuro-Fuzzy Inference System

Artificial Neuro-Fuzzy Inference System (ANFIS) [16] contains first-order Takagi-Sugeno-Kang (TSK) type fuzzy rules, while assume the form of $f(\cdot)$ to be first-order polynomial:

$$f(x_1, x_2, ..., x_n) = \sum_{i=1}^{n} b_i x_i + c_i,$$

where b_i and c_i are parameters to be estimated. In the five-layer architecture shown in Fig. 5. Input $\boldsymbol{x}$ is mapped through stiff membership functions $\mathcal{A}_j = \mu_{ij}(x_i)$, which require prior knowledge to craft.

The outcome weight is calculated with a product t-norm $\top_{\text{prod}}(a, b) = a \cdot b$, hence we have:

$$w_j = \prod_{i=1}^{n} \mu_{ij}(x_i)$$

Each node in the rule layer denotes a piece of rule with a certain degree that should be applied to $\boldsymbol{x}$. The final output is a weighted average of all rule outputs. Because the output values of TSK rules are crisps, defuzzification process is not required here.

$$y = \frac{\sum_{j=1}^{r} w_j y_j}{\sum_{j=1}^{r} w_j}$$

Our implementation of ANFIS learns function parameters by back propagation and iteratively modification to minimize the error function, which is defined as a least mean square:

$$Error = \frac{1}{2}\sum_{k=1}^{n}[y_k(w) - y_k^*(w)]^2$$

the iteration for updating weights $\boldsymbol{w} = (w_1, w_2, ..., w_n)$:

$$\boldsymbol{w}(k) = \boldsymbol{w}(k-1) - \eta\Delta\boldsymbol{w}$$

where η is the learning rate. The convergence of this algorithm is mathematically guaranteed [25].

B. Neural Fuzzy Controller

The architecture of Neural Fuzzy Controller (NEFCON) is more succinct comparing to ANFIS and similar Neuro-Fuzzy Inference Systems. It implements a three-layer structure of Mamdani type fuzzy perceptrons.

As shown in Fig. 6, two layers of Membership Functions (MF) are used instead of weights. Rule nodes here take the form of:

$$\begin{aligned}&\mathit{IF}\ x_{1j} = \mu_{1j}\ \mathit{and}\ x_{2j} = \mu_{2j}\ ...\ \mathit{and}\ x_{nj} = \mu_{nj}\\ &\quad \mathit{THEN}\ y_{1j}^{+} = \nu_{1j}\ \mathit{and}\ y_{2j}^{+} = \nu_{2j}\ ...\ \mathit{and}\ y_{nj}^{+} = \nu_{nj}\\ &\quad y_i = \sum_j y_{ij}\end{aligned}$$

Fuzzy error back-propagation learning algorithms are frequently employed to adapt both the MF parameters and rule nodes. The conventional triangular MF for NEFCON input and output variables have three or two parameters respectively:

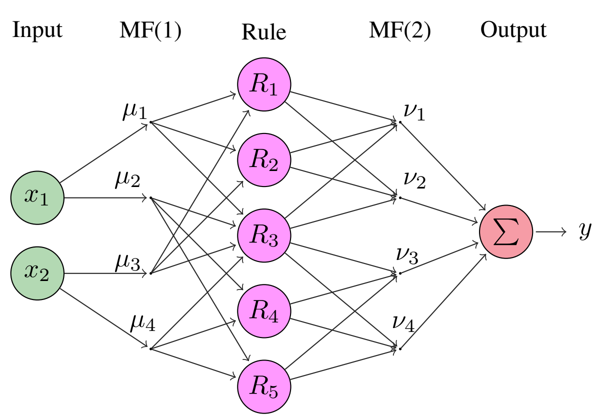

Fig. 6. Architecture of Neural Fuzzy Controller

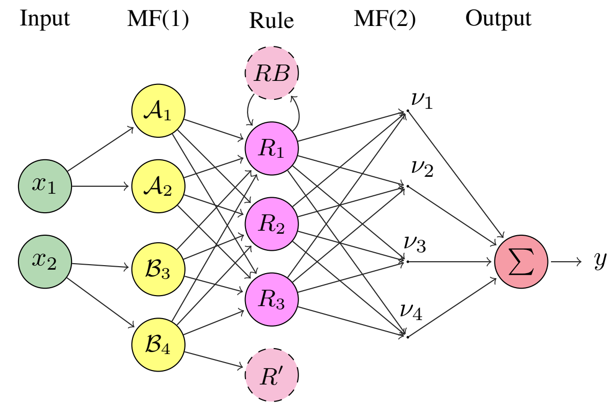

Fig. 7. Architecture of Evolving Fuzzy Neural Network

$$\mu_{ij}(x) \overset{def}{=\!=} \begin{cases} \dfrac{x - a_{ij}}{b_{ij} - a_{ij}}, & if\ x \in [a_{ij}, b_{ij}] \\ \dfrac{c_{ij} - x}{c_{ij} - b_{ij}}, & if\ x \in [b_{ij}, c_{ij}] \\ 0, & otherwise \end{cases}$$

where a_{ij}, b_{ij}, $c_{ij} \in R$, $a_{ij} \leq b_{ij} \leq c_{ij}$.

$$\nu_j(y) \overset{def}{=\!=} \begin{cases} \dfrac{d_j - y}{d_j - e_j}, & if\ y \in [\min\{d_j, e_j\}, \max\{d_j, e_j\}] \\ 0, & otherwise \end{cases}$$

where $d_j, e_j \in R$.

Output o_y of NEFCON is calculated by

$$o_y = \frac{\sum_R o_R \cdot t_R}{\sum_R t_R}$$

where the output of activated rule can be represented by $o_R = \max\{\mu_{1,R}(x_1), \cdots, \mu_{n,R}(x_n)\}$, and desired output represented by $t_R = \nu_R^{-1}(o_R)$. Then, the fuzzy rule error can be applied to adjust parameters $a, b, ..., e$ with a learning rate similar to the one used for tuning ANFIS. The single output architecture we use for prediction can also be regarded as a special case of NEFPROX[1] [24].

C. Evolving Fuzzy Neural Networks

Evolving Fuzzy Neural Networks (EFuNN) was firstly proposed as an instantiated FNN that evolves according to the connectionism idea expatiated in ECOS [18]. The four-layer structure is illustrated in Fig. 7.

EFuNN implements Mamdani type fuzzy rules. Therefore, input variables are directly mapped to rules through fuzzy membership functions without weights layer. There are richer connections from MF(1) to rule nodes in EFuNN architecture than in NEFCON. Although, after the rule layer EFuNN has a technically similar structure as NEFCON, the underlying considerations are quite different. MF(2) in fact represents the

[1]We use an encapsulated .NET implementation with GUI developed by Sahal Arafat Zain called NefProx.NET version 1.0

actions to be activated. The most interesting fact about EFuNN is that the rule nodes maintains two vectors of connection weights in a dynamic manner. Rule nodes can be created or interact with a rulebase (RB) by machine learning techniques [13]. For each rule node R_i, weights associated to input variables $\boldsymbol{w_1}(R_i)$ and weights associated to output variables $\boldsymbol{w_2}(R_i)$ are updated by:

$$\boldsymbol{w_1}(R_i^{t+1}) = \boldsymbol{w_1}(R_i^t) + \eta_{i,1}(\boldsymbol{w_1}(R_i^t) - \boldsymbol{x}_{fuzzy})$$
$$\boldsymbol{w_2}(R_i^{t+1}) = \boldsymbol{w_2}(R_i^t) + \eta_{i,2}(A_2 - \boldsymbol{y}_{fuzzy})A_1(R_i^t)$$

where $\eta_{i,1}$ and $\eta_{i,2}$ are R_i's learning rates for its input layer and output layer connections. $A2 = f_2(\boldsymbol{w_2}A1)$ is the activation vector of fuzzy output neurons. $A_1(R_i^t) = f_1(D(\boldsymbol{w_1}(R_i^t), \boldsymbol{x}_{fuzzy}))$ is the activation function of rule node R_i^t, where D is a function to measure a local normalized fuzzy distance between two fuzzy membership vectors MF(1) and MF(2) [17].

During the learning process, when a new example arrived at a rule node R_i, its radius r_i and sensitivity threshold s_i can be updated using:

$$r_i^{t+1} = r_i^t + D(\boldsymbol{w_1}(R_i^{t+1}), \boldsymbol{w_1}(R_i^t))$$
$$s_i^{t+1} = s_i^t - D(\boldsymbol{w_1}(R_i^{t+1}), \boldsymbol{w_1}(R_i^t))$$

which makes the structure always optimized at the current stage. As a result, EFuNN is considered advantageous for online learning problems.

D. Dynamic Evolving Neuro-Fuzzy Inference System

Dynamic Evolving Neuro-Fuzzy Inference System (DENFIS) [17] implements TSK type fuzzy inference engine. The link between antecedents x_i and fuzzy sets $\mathcal{A}_i$ is forged more randomly than in EFuNN. Each time m fuzzy rules fire together to produce the output. Therefore, the inference process can be represented as:

$$\textit{IF } x_1 = \mathcal{A}_{m1} \textit{ and } x_2 = \mathcal{A}_{m2} \textit{ ... and } x_n = \mathcal{A}_{mn}$$
$$\textit{THEN } y = f_m(x_1, x_2, ..., x_n)$$

From the holistic perspective, DENFIS can be recognized as an improved version of EFuNN. The main difference is on learning phase. One modification is that DENFIS introduces an online learning algorithm under the name of Evolving Clustering Method (ECM) to continuously change the parameters of triangular membership functions:

$$\begin{aligned}\mu(x) &= f(x, a, b, c) \\ &= \max(\min(\frac{x-a}{b-a}, \frac{c-x}{c-b}), 0)\end{aligned}$$

where b is the value of cluster center of x, a and c are dependent on b. The defuzzification process is the same as ANFIS.

Another exclusive feature for DENFIS is that forgetting factor is taken into account during fuzzy rule learning. This feature enables DENFIS to be more robust than other FNN types on condition that the principle behind modeling data is evolving, which is attested in Section IV.

IV. Simulation

It is notable that there have been many discussion about how to decide the time-delay and embedding dimension for a time series. Some studies heuristically tune the parameters based on the predicting result [22], while others [9] [19] introduce some criteria. In our method, we take three factors into consideration. Firstly, we attach importance to the classic methods like first minimum mutual information step and Cao's embedding theorem [2]. Secondly, we align the same method as popular in literature, for instance G-P algorithm [12]. Thirdly, we respect the convention proposed intuitively [34] or without explicit explanation. Kaplan-Yorke dimensions are referred for consideration as well [31], though they can jump as parameters evolve.

The benchmark autoregression (AR) method determines the appropriate time lags with the help of plotting partial autocorrelation function (PACF). For chaotic Duffing time series, the proper time delay in our data sampling timestep is unknown.

Therefore, we resort to trial-and-error. Experiments suggest the optimum RMSE is achieved when $k = 3$. Finally we arrive at the sampling method according to information delivered by Table I,

TABLE I
SCALE PROPERTIES APPROXIMATED FOR SOME CHAOTIC SYSTEMS

Chaotic system	Time delay	Embedding dims.	System dims.
Logistic map	1	1	0.5
Duffing	k	2	1
Mackey-Glass	6	4	2
Lorenz	3	6	3

To make our result comparable to previous studies, we adopt the result evaluation criteria as in [11] and [14]. In detail, three measurements addressing different aspect of the performance are selected: root mean square error (RMSE), non-dimensional error index (NDEI), and normalized mean square error (NMSE). MSE is not presented, because it add no more information other than the square of RMSE. When the number of sampling n is large, NMSE is roughly the square of NDEI in a single round of experiment.

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

$$NMSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{\sum_{i=1}^{n}(y_i - \bar{y_i})^2}$$

$$NDEI = \frac{RMSE}{\sigma(y_1, y_2, ..., y_n)}$$

where y, $\hat{y}$, and $\bar{y}$ stand for the values observed from chaotic time series, prediction made by FNN and average of observed data respectively.

TABLE II
PERFORMANCE OF DIFFERENT TYPE OF FNNS FOR PREDICTING EVOLVING CHAOTIC TIME SERIES

Model	Error	White noise	Logistic map		Duffing		Mackey-Glass		Lorenz	
			static	evol.	static	evol.	static	evol.	static	evol.
AR	RMSE	1.01E+0	4.09E-1	4.85E-1	1.12E-1	1.44E-1	1.26E+0	3.92E+0	3.35E+0	5.47E+0
	NMSE	1.03E+0	3.45E+0	4.87E+0	1.27E-2	1.91E-2	2.67E+1	7.88E+1	3.94E+0	6.36E+0
	NDEI	1.01E+0	1.85E+0	2.21E+0	1.22E-1	1.38E-1	5.13E+0	9.01E+0	1.98E+0	2.35E+0
ANFIS	RMSE	1.17E+0	1.83E-4	1.01E-3	7.01E-2	6.67E-2	8.47E-2	1.65E-1	9.38E+0	1.44E+2
	NMSE	1.26E+0	2.82E-6	2.09E-4	1.20E-1	1.20E-1	8.76E-2	1.73E+0	7.83E-1	3.64E+1
	NDEI	1.18E+0	**7.71E-4**	**3.38E-3**	**7.48E-2**	**6.95E-2**	3.69E-1	6.19E-1	8.55E-1	6.92E+0
NEFCON	RMSE	1.00E+0	7.74E-2	2.02E-1	3.62E-1	3.43E-1	1.35E-1	2.07E-1	9.00E+0	2.73E+1
	NMSE	1.03E+0	1.25E-1	4.77E-1	1.41E-1	1.22E-1	3.14E-1	6.12E-1	5.08E-1	1.40E+0
	NDEI	**1.01E+0**	3.50E-1	6.86E-1	3.76E-1	3.48E-1	5.59E-1	7.64E-1	7.12E-1	1.18E+0
EFuNN	RMSE	1.39E+0	3.03E-1	4.43E-1	9.90E-2	9.91E-2	6.52E-2	2.16E-1	6.12E+0	1.44E+1
	NMSE	1.94E+0	1.89E+0	2.26E+0	1.10E-2	2.39E-2	7.66E-2	7.25E-1	4.39E-1	4.35E-1
	NDEI	1.39E+0	1.37E+0	1.50E+0	1.03E-1	1.34E-1	2.74E-1	8.30E-1	4.70E-1	6.57E-1
DENFIS	RMSE	1.06E+0	3.32E-2	3.78E-2	9.18E-2	9.16E-2	1.81E-2	1.37E-1	2.55E+0	7.13E+0
	NMSE	1.12E+0	3.19E-2	3.81E-2	9.06E-3	1.81E-2	6.22E-3	3.02E-1	4.85E-2	1.02E-1
	NDEI	1.06E+0	1.50E-1	1.07E-1	9.48E-2	1.19E-1	**1.45E-1**	**5.32E-1**	**2.14E-1**	**3.13E-1**

Around 1500 simulated values are generated and sampled according to the following formulas from each evolving chaotic time series. Lm, D, MG and L are the acronyms for corresponding types of chaotic system.

$$Lm_i = [Lm(t-1), Lm(t)]$$
$$D_i = [D(t-3), D(t), D(t+3)]$$
$$MG_i = [MG(t-18), MG(t-12), ..., MG(t), MG(t+6)]$$
$$L_i = [L(t-15), L(t-12), ..., L(t), L(t+3)]$$

The first 100 data points are leftover to suppress early disordered behavior. In this early stage, the system is adapting to the chaotic characteristics, but the influence of initial value is still not negligible. The rest of data are partitioned into around 1000 samples for training and the remaining samples for testing. For each type of chaotic time series, experiments are conducted 5 times for evolving parameters and 5 times for static parameters, with various FNN structures. Due to the limit of space, we only report the average RMSE, NMSE and NDEI in Table II. The standard deviations, assuming error measurements are normally distributed, are not included.

Autoregression (AR as in Table II) is provided as a benchmark method. We have selected the model with appropriate order and timestep lag to minimize errors. White noise with a standard deviation $\sigma = 1$ is experimented as well. If all the predicting errors are significantly larger than errors on white noise, we can conclude that this model is ineffective on forecasting this type of time series.

Furthermore, we compare the performance of our implementation and results from other studies on predicting static parameter Mackey-Glass chaotic time series with ANFIS in Table III. The comparison shows that these measurements can vary greatly from case to case. One possible reason is these studies use different FNN configurations, for example the maximum number of rules, initialization of time series etc. Another factor to consider is the stability of the model itself.

TABLE III
PERFORMANCE OF PREDICTING MG TIME SERIES WITH ANFIS

Reference	RMSE	NMSE	NDEI
Maguire et al. [22]	1.05E-2	–	–
Gholizade et al. [11]	1.80E-3	2.90E-5	–
Heydari et al. [14]	3.25E-2	–	1.44E-1
Our Simulation	8.47E-2	8.76E-2	3.69E-1

As a consequence, the conclusion that one implementation is better than another should be drawn prudently as long as errors belong to the same order of magnitude.

As Table II implies [2], most FNN models are more capable for depicting chaotic time series than the classic autoregression method. In most cases, evolving chaotic time series are more difficult to approximate, but how much the predicting error would be worse is not clear. Moreover, the predictability diminishes as the dimension of chaotic system increase, regardless of the specific model. For time series generated by an evolving chaotic Lorenz system, using ANFIS or NEFCON for prediction is nonsensical.

We have noticed that ANFIS is especially sensitive to the parameter evolvement. This may be observed owing to the rather dense connections between different layers. According to our understanding, this feature makes the adjustment of weights more clumsy in an offline learning environment. Therefore, the model loses its stability and consistency in the end.

Fig. 8 illustrates the overreaction of ANFIS model more explicitly when it is used to predict an evolving Lorenz chaotic time series. In the first half part, the prediction closely follow the target chaotic time series; while in the second half, the prediction is over-sensitive to the realized values.

[2]The bold font in Table II highlights the best performance of models experimented for every data column, using average NDEI as the primary measurement

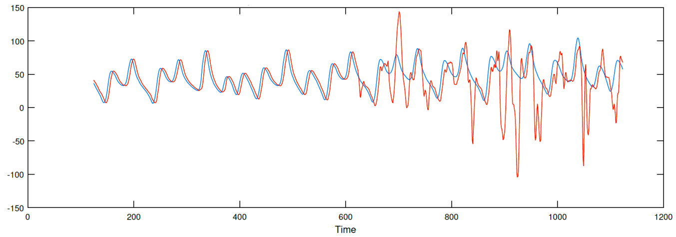

Fig. 8. Destabilization of ANFIS model due to parameter evolvement

Among the experimented models, NEFCON has the best consistency. The differences between each round of experiments are small. NEFCON model also provides the closest estimation of white noise statistics. In addition, the impact of parameter evolvement is not conspicuous comparing to other models.

Despite the aforementioned disadvantages, ANFIS and DENFIS can universally adapt to different chaotic systems. This phenomenon seems to indicate that TSK type Neuro-Fuzzy Inference Systems are more powerful for predicting chaotic time series because they establish direct link between input and output variables. As the complexity of chaotic system grows, the frequency domain spectrum evolves more rapidly with the parameter change. Accordingly, the prediction model requires algorithms to obtain priors, for example ECM, to emphasize the current state.

This theory is supported by the fact that, ANFIS predicts Logistic map and Duffing system more accurately, but for Mackey-Glass and Lorenz time series, DENFIS produces better results. Although, whether the after-coming learning phase should response to the detected parameter change or not [23] remains a question untouched.

V. Conclusion

In this paper, we introduced the concept of evolving chaotic time series. Four example systems that potentially exhibit chaotic behavior are provided with details on dimension properties and linear time dependency of parameters. Four FNN implementations, namely ANFIS, NEFCON, EFuNN, and DENFIS are used to predict the simulated chaotic time series. Experimental results suggest that DENFIS skillfully trades the prediction accuracy off against model stability and consistency. Therefore, we consider TSK type Neuro-Fuzzy Inference System to have better predictability among numerous primary FNN architectures.

Further work will chiefly focus on answering two questions. First, how would the phase portrait and other properties of a chaotic time series change, if the system behind it has nonlinearly evolving parameters or paradigm shift. Second, what online learning techniques and trigger conditions can be used to confer better prediction accuracy on TSK type Neuro-Fuzzy Inference Systems. It is also of sufficient interest to apply our methods on real-life financial data, for which the dynamics of chaotic system in behind is challenging and not widely agreed.

References

[1] C. H. Aladag, U. Yolcu, and E. Egrioglu, "A high order fuzzy time series forecasting model based on adaptive expectation and artificial neural networks," *Mathematics and Computers in Simulation*, Vol. 81(4), pp. 875–882, 2010.

[2] L. Cao, A. Mees, and K. Judd, "Dynamics from multivariate time series," *Physica D: Nonlinear Phenomena*, Vol. 121, pp. 75–88, 1998.

[3] M. Casdagli, "Nonlinear prediction of chaotic time series," *Physica D: Nonlinear Phenomena*, Vol. 35, No. 3, pp. 335–356, 1989.

[4] M. Casdagli, S. Eubank, J. D. Farmer, and J. Gibson, "State space reconstruction in the presence of noise," *Physica D: Nonlinear Phenomena*, Vol. 51, pp. 52–98, 1991.

[5] T.-L. Chen, C.-H. Cheng, and H.-J. Teoh, "High-order fuzzy time-series based on multi-period adaptation model for forecasting stock markets," *Physica A: Statistical Mechanics and its Applications*, Vol. 387(4), pp. 876–888, 2008.

[6] P. DeGrauwe and H. Dewachter, "Deterministic assessment of nonlinearities in models of exchange rate determination," *Review of Economic Studies*, Vol. 58, No. 3, pp. 603–619, 1991.

[7] R. F. Engle, "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation," *Econometrica*, 50 (4): 987–1007, 1982.

[8] J. D. Farmer and J. J. Sidorowich, "Predicting Chaotic Time Series," *Physical Review Letters*, 59 (8): 845–848, 1987.

[9] A. M. Fraser and H. L. Swinney, "Independent coordinates for strange attractors from mutual information," *Physical Review A*, 33(2): 1134–1140, 1986.

[10] A. Gholipour, C. Lucas, B.N. Araabi, M. Mirmomeni, and M. Shafiee, "Extracting the main pattern of natural time series for long-term neuro fuzzy prediction," *Neural Computing & Application*, Vol. 16, pp. 383–393, 2006.

[11] H. Gholizade-Narm and M. R. Shafiee-Chafi, "Using repetitive fuzzy method for chaotic time series prediction," *Journal of Intelligent & Fuzzy Systems*, vol. 28, no. 4, pp. 1937–1946, 2015.

[12] P. Grassberger and I. Procaccia, "Characterization of strange attractors," *Physics Review Letters*, vol. 50, pp. 346–349, 1983.

[13] J. A. M. Hernyndez, F. G. Castayeda, and J. A. M. Cadenas, "An Evolving Fuzzy Neural Network Based on the Mapping of Similarities," *IEEE Transactions on Fuzzy Systems*, Vol. 17, No. 5, pp. 1379–1396, 2009.

[14] G. Heydari, MA. Vali, and A. A. Gharaveisi, "Chaotic time series prediction via artificial neural square fuzzy inference system," *Expert Systems with Applications*, Vol. 55, pp. 461–468, 2016.

[15] K. Huarng and H.-K. Yu, "A Type-2 fuzzy time series model for stock index forecasting," *Physica A: Statistical Mechanics and its Applications*, Vol. 353, pp. 445–462, 2005.

[16] J.-S. R. Jang, "ANFIS: Adaptive-Network-based Fuzzy Inference Systems," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 23, No. 3, pp. 665-685, 1993.

[17] N. Kasabov, "Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 31, No. 6, pp. 902-918, 2001.

[18] N. Kasabov, "ECOS: Evolving Connectionist Systems And The ECO Learning Paradigm," *manuscript retrieved from CiteSeer, doi=10.1.1.44.8509.*, 2016

[19] M. B. Kennel, R. Brown, and H. D. Abarbanel, "Determining embedding dimension for phase-space reconstruction using a geometrical construction," *Physical Review A*, 45(6): 3403–3411, 1992

[20] T.-Y. Li and J. A. Yorke, "Peroid Three Implies Chaos," *The American Mathematical Monthly*, Vol. 82, No. 10, pp. 985–992, 1975.

[21] Z. Liu, "Chaotic Time Series Analysis," *Mathematical Problems in Engineering*, Article ID 720190, 2010.

[22] L.P. Maguire, B. Roche, T. M. McGinnity, and L. J. McDaid, "Predicting a chaotic time series using a fuzzy neural network," *Information Sciences*, vol. 112, no. 1, pp. 125–136, 1998.

[23] C. Monteleoni and T. Jaakkola, "Online Learning of Non-stationary Sequences," in *Advances in Neural Information Processing Systems*, 2004.

[24] D. Nauck and R. Kruse, "Neuro-fuzzy systems for function approximation," *Fuzzy Sets and Systems*, vol. 101, pp. 261–271, 1999.

[25] A. B. Novikoff, "On convergence proofs on perceptrons," in *Symposium on Mathematical Theory of Automata*, 1962.

[26] L. Oneto, F. Bisio, E. Cambria, and D. Anguita, “Statistical learning theory and ELM for big social data analysis.” *IEEE Computational Intelligence Magazine*, vol. 11(3), pp. 45–55, 2016.

[27] E. E. Peters, “A Chaotic Attractor for the S&P 500,” *Financial Analysts Journal*, vol. 47, no. 2, pp. 55–81, retrieved from http://www.jstor.org/stable/4479416, 1991.

[28] S. Poria, E. Cambria, A. Gelbukh, F. Bisio and A. Hussain, “Sentiment Data Flow Analysis by Means of Dynamic Linguistic Patterns,” *IEEE Computational Intelligence Magazine* 10(4), pp. 26–36, 2015.

[29] S. Poria, E. Cambria, and A. Gelbukh, “Aspect extraction for opinion mining with a deep convolutional neural network.” *Knowledge-Based Systems*, vol. 108, pp. 42–49, 2016.

[30] A. Rotshtein, L. Pustylnik, and Y. Giat, “Fuzzy Logic and Chaos Theory in Time Series Forecasting,” *Journal of Intelligent & Fuzzy Systems*, vol. 31, no. 11, pp. 1056–1071, 2016.

[31] J. C. Sprott, “Chaos and Time-Series Analysis,” *Oxford University Press*, ISBN 0-19-850839-5, 2003.

[32] F. Takens, “Detecting strange attractors in turbulence,” in *Dynamical Systems & Turbulence, Lecture Notes in Mathematics*, vol. 898, Springer-Verlag, pp. 366–381, 1981.

[33] J. Vieira, D. F. Morgado, and A. Mota, “Neuro-Fuzzy Systems: A Survey,” *IEEE Transactions on Systems, Man, and Cybernetics*, Udine, Italy, 2004.

[34] L. X. Wang and J. M. Mendel, “Generating fuzzy rules by learning from examples,” in *Proceedings of the 5th WSEAS NNA International Conference on Neural Networks and Applications*, 22(6): 1414–1427, 1992.

[35] L. A. Zadeh, “Outline of a new approach to the analysis of complex systems and decision processes,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no.1, pp. 28–44, 1973.