Data Driven Fuzzy Systems and Deep Neural Networks for Nonlinear Dynamic Processes Modeling

Daniel Leite
Department of Computer Science
Paderborn University
Paderborn, Germany
daniel.leite@uni-paderborn.de

Igor Skrjanc
Faculty of Electrical Engineering
University of Ljubljana
Ljubljana, Slovenia
Igor.Skrjanc@fe.uni-lj.si

Fernando Gomide
School of Electrical and Computer Engineering
University of Campinas
Campinas, Brazil
gomide@unicamp.br

***Abstract*—Data-driven modeling has become a predominant approach for learning and understanding systems. Models span a wide range of domains, including medical, health informatics, biological, environmental, meteorological, transportation, and economic systems, as well as complex dynamic virtual information, engineering, and hybrid systems. Computational intelligence techniques, including fuzzy systems, neural networks, evolutionary computation, and their hybridizations, are key driving forces in the current data-driven system modeling effort. This paper addresses data-driven fuzzy and neural modeling analytics and analyzes their performance in nonlinear dynamic systems modeling and prediction tasks. Data-driven fuzzy modeling and neural-based analytics are powerful paradigms that compete closely, often outperforming many alternative state-of-the-art methods, such as gradient boosting, kernel ridge regression, and Gaussian processes. In particular, the flexibility and approximation capabilities of data-driven fuzzy models, long-short-term memory, and convolutional neural networks are analyzed in two applications: (i) learning from a data stream produced by a synthetic nonlinear difference equation, and (ii) electricity load time series forecasting. The usefulness of the models is discussed in terms of their prediction accuracy and parametric complexity. In computational experiments, the recursive or incremental level-set fuzzy model provided both accuracy and interpretability with fewer parameters compared to the other methods, making it an attractive option for real-time modeling and forecasting tasks.**

***Index Terms*—fuzzy systems, machine learning, neural networks, dynamic systems.**

I. Introduction

Information systems and engineering processes often require quantitative analysis and employ many techniques to model, analyze, and design effective solutions. Typically, qualitative concepts, system descriptions, and perceptions need to be supplemented with quantitative models – devices that act as approximations of the behavior of actual systems. It is common in many information and engineering contexts that the modeling process relies on input-output data, under the assumption that some mathematical relationship exists between them. In dynamic systems, the output typically exhibits nonlinear dependence on past input values. Moreover, the output is time-dependent, and the input-output functional relationship is often too complex to be solved in closed form to yield an explicit solution for the output.

The input-output modeling process follows a black-box approach, where knowledge about the system is observed solely from the model output response to an input. The precise internal structure of the system is left unspecified. The task is to deduce the characteristics of the black box by experimenting with different inputs and observing the resulting outputs. Conversely, state-space modeling incorporates structural knowledge of the system, such as state variables, system order, and first-principles-derived parameters, which represent fundamental physical relationships. Both input-output and state-space modeling are model-based techniques in which the principles governing system behavior are combined with parameter estimation or system identification procedures [1].

Data-driven modeling relies on data, that is, a set of training instances that illustrate how the system behaves. The objective is to formulate a hypothesis that ideally mirrors the actual system behavior across the training instances. Data-driven modeling assumes that any hypothesis that approximates the system behavior over a sufficiently large set of input-output training instances will also generalize well to unobserved inputs [2], [3]. Machine learning and optimization techniques are often used to formulate such hypotheses, leveraging algorithms that can learn from large datasets and improve predictions over time. An intermediate modeling approach combines this data-driven learning process with a higher-level understanding of the system structure, thus generating a description of the system where the input-output response reflects both the system's structural insights and the input-output behavior gleaned from the training data.

Over the past few decades, the emphasis in data-driven modeling has shifted towards the use of computational intelligence and machine learning methods to build models that complement knowledge-driven models. Examples of such methods include statistical techniques, decision trees, support vector machines, ensemble methods, Bayesian networks, random forests, deep learning architectures, neural networks, fuzzy rule-based systems, and their combinations [8], [10],

[9]. Advances of data-driven modeling methods require the investigation of a variety of strategies because the modeling techniques should account for factors such as performance, interpretability, explainability, robustness, adaptability, scalability, usefulness, and the specific characteristics of the application domain.

Dynamic systems models based on data streams have been extensively researched in many domains, particularly in engineering, control systems, and applied mathematics. Early work developed during the 1990s has explored neural networks [4], [5], fuzzy systems, and neuro-fuzzy methods [6], [7]. Subsequent decades witnessed progression encompassing neural networks, instance-based modeling, functional fuzzy modeling, regression trees, knowledge-based approaches [8], and first and second-order derivative function mechanisms for approximating nonlinear state-space models [11]. The use of approximation for modeling and solving optimal control problems has been pursued in [13] particularly using proper orthogonal decomposition and radial basis functions. Recent endeavors have addressed deep learning models for system identification [12]. Additionally, nonlinear autoregressive Gaussian processes with exogenous inputs, long short-term memory architectures, and convolutional neural networks to model nonlinear dynamics have been explored [14].

This paper addresses the utilization of the recursive incremental level set method (RLSM), long short-term memory (LSTM), convolutional neural network (CNN), and hybrid approaches in the development of dynamic processes models. These methodologies represent the state of the art within the fuzzy systems and neural computing analytics. Despite their sophistication, there has been a lack of analysis and performance evaluation of these methods in complex nonlinear dynamic processes modeling and prediction tasks. The results have shown that, in spite of their margin of approximation, these approaches compete closely from the point of view of accuracy, but in general the complexity of the fuzzy models are lower. Additionally, fuzzy models are more transparent than neural models due to their reduced count of parameters and reliance on partitions of the data space. These partitions, along with their ordering along the feature axes, can be accompanied by linguistic values and rules, thereby supporting human understanding and decision-making processes.

The structure of the paper is organized as follows. Following this introduction, Section II reviews the data-driven level set fuzzy modeling, and recalls two prominent deep learning architectures, the long short-term memory and convolutional neural network structures. Section III provides the computational results and the practicality of the models employed. Finally, Section IV concludes the paper, summarizing its contributions and proposing topics for future development.

II. Fuzzy Machine Learning and Deep Neural Nets

This section provides a concise overview of the data-driven fuzzy machine learning method using the notion of level sets. The section also gives a short reminder of the long short-term memory models and convolutional neural networks to ensure the paper conceptually and methodologically self-contained. The level set-based fuzzy model is a novel form of additive fuzzy rule-based model that is simpler and easier to understand compared to traditional linguistic and functional fuzzy models [7] [16]. Level set-based fuzzy models are computationally fast, efficient, and interpretable. Long short-term memory models constitute a class of recurrent deep neural networks whose structure is tailored to encode time-related data, a property that is particularly important to model dynamic systems [27]. Convolutional neural networks are multi-layer structures that have proved to be effective in a variety of dynamic systems modeling tasks due to their ability in capturing local trends and patterns in input-output data relationships [28]. These modeling methods are representative of the current state of the art in fuzzy and neural modeling.

A. Data-Driven Level Set Fuzzy Modeling

The fuzzy model discussed in this paper comprises a collection of fuzzy rules of the following form:

$$\mathcal{R}_i: \text{ if } x \text{ is } \mathcal{A}_i \text{ then } y \; is \; \mathcal{B}_i \tag{1}$$

where $i = 1, 2, \ldots, N$, and $\mathcal{A}_i$, and $\mathcal{B}_i$ are convex fuzzy sets with membership functions $\mathcal{A}_i(x) : \mathcal{X} \to [0, 1]$ and $\mathcal{B}_i(y) : \mathcal{Y} \to [0, 1]$. Given an input $x \in \mathcal{X}$, the steps of the level set method are as follows [15]:

1) Compute the activation level of each rule $\mathcal{R}_i$

$$\tau_i = \mathcal{A}_i(x) \tag{2}$$

2) For each τ_i, find the level set

$$\mathcal{B}_{\tau_i} = \{y | \tau_i \leq \mathcal{B}_i(y)\} = [y_{il}, y_{iu}] \tag{3}$$

3) Calculate the midpoint of the corresponding level set

$$m_i(\tau_i) = \frac{y_{il} + y_{iu}}{2} \tag{4}$$

4) Compute the model output $\hat{y}$ as

$$\hat{y} = \frac{\sum_{i=1}^{N} \tau_i \; m_i(\tau_i)}{\sum_{i=1}^{N} \tau_i} \tag{5}$$

Notice that $\tau_i > 0$ for at least one i. In most practical scenarios, the fuzzy sets $\mathcal{A}_i$ and $\mathcal{B}_i$ are continuous. If the fuzzy set $\mathcal{B}_i$ is discrete, then m_i represents the average of the elements of $\mathcal{B}_{\tau_i}$. Equation (5) can generally be interpreted as a mapping in the form $\mathcal{F} : [0, 1] \to \mathcal{Y}$. In other words, $\mathcal{F}$ denotes a mapping from membership degrees to elements within the output domain $\mathcal{Y}$. This differs significantly from the nature of fuzzy linguistic and fuzzy functional models, and their associated data processing mechanisms.

Let $\mathcal{D} = \{(x^k, y^k)\}$ be a dataset, where $x^k \in \mathbb{R}^p$ and $y^k \in \mathbb{R}$, such that $y^k = f(x^k)$, $k = 1, 2, \ldots, K$. In data-driven fuzzy level set modeling, the objective is to construct a fuzzy model $\mathcal{F}$ to approximate the function f using $\mathcal{D}$ [16].

In fuzzy rule-based modeling, determining the number of rules, the type of membership functions for the antecedent and

consequent of each rule, and their corresponding parameters is essential. This specification is achieved through the granulation of the input and output spaces, which is autonomously performed by a machine learning method using $\mathcal{D}$. Clustering is the preferred method for granulating the input-output space and defining membership functions. Input-output clustering also aids interpretability due to the localized nature of the rules. Cluster centers are often used as the modal values for normal membership functions, which define the antecedent terms $\mathcal{A}_i$. A normal membership function is one in which the highest degree of membership is 1, i.e., there exists some x_0 such that $\mathcal{A}_i(x_0) = 1$. Additionally, cluster validity methods assist in determining the appropriate number N of rules [7].

Once the membership functions $\mathcal{A}_i$ are specified, the data-driven level set fuzzy modeling method requires estimates of the corresponding local output functions $\mathcal{F}_i$, $i = 1, 2, \ldots, N$. While numerous options exist, and theoretically any type of function approximator could be chosen, this paper adopts the simplest approach, which is affine output functions $\mathcal{F}_i$ of the form

$$\mathcal{F}_i(\tau_i) = m_i(\tau_i) = v_i\tau_i + w_i \tag{6}$$

The values of the parameters v_i and w_i can be estimated from the data using any suitable procedure, such as least squares-based learning procedures. For instance, the pseudo-inverse based solution proceeds as follows.

For each data pair (x^k, y^k), compute the activation level $\tau_i^k = \mathcal{A}_i(x^k)$, $i = 1, 2, \ldots, N$, and let $s^k = \sum_{i=1}^{N} \tau_i^k$. From (5) and (6), the corresponding output is

$$z^k = \frac{\tau_1^k(v_1\tau_1^k + w_1)}{s^k} + \ldots + \frac{\tau_N^k(v_N\tau_N^k + w_N)}{s^k} \tag{7}$$

Let $\mathbf{d}^k = [(\tau_1^k)^2/s^k,\ \tau_1^k/s^k,\ \ldots,\ (\tau_N^k)^2/s^k,\ \tau_N^k/s^k]$ and the vector of parameters be $\mathbf{u} = [v_1, w_1, \ldots, v_N, w_N]^T$. Equation (7) becomes

$$z^k = \mathbf{d}^k \cdot \mathbf{u}, \quad k = 1, \ldots, K \tag{8}$$

Let $\mathbf{z} = [z^1, \ldots, z^K]^T$, $\mathbf{D} = [d^{1T}, \ldots, d^{KT}]^T$, and $\mathbf{y} = [y^1, \ldots, y^K]^T$. Thus, the set of equations (8) can be succinctly expressed as $\mathbf{z} = \mathbf{D}\mathbf{u}$. The parameter vector $\mathbf{u}$ is the solution of $min_u||\mathbf{y} - \mathbf{z}||^2$,

$$\mathbf{u} = \mathbf{D}^+\mathbf{z} \tag{9}$$

where $\mathbf{D}^+$ is the Moore-Penrose pseudo-inverse of $\mathbf{D}$ [17]. Therefore, given an input x, the corresponding model output is obtained from

$$\hat{y} = \mathbf{d}\mathbf{u} \tag{10}$$

which is the vector equivalent to that of (5). The number of parameters L of a level set fuzzy model with n inputs and p antecedent membership function parameters is given by $L = (np + 2)N$. The computation of the output is bounded by the computation of $\mathbf{D}$. If $\mathbf{D}$ is an $(m \times n)$ matrix, then the pseudo-inverse computation based on singular value decomposition is $\mathcal{O}(nm\ \min(m, n))$ [18].

The least squares estimator (9) can be expressed as

$$\mathbf{u}^k = \mathbf{D}^+\mathbf{z} \tag{11}$$

where the superscript k is added to denote the number of data pairs used to estimate $\mathbf{u}$. It may be desirable to compute the least squares estimate for different values of $k \to \infty$, and to systematize the parameter estimation computations in a way that the result obtained for k data pairs can be used to obtain the estimate for $k+1$ data pairs. The recursive form of parameter estimation can also be convenient when the number of parameters is not known in advance, which is particularly suitable for level set-based modeling when output functions of interest are other than affine. Recursive parameter estimation has been studied in the literature [1], [6], [19]. In the level set fuzzy modeling method, the recursive (incremental) steps are as follows (see [20] for details).

For each data pair (x^k, y^k), $k = 1, \ldots$

1) Compute the activation levels $\tau_i^k = \mathcal{A}_i(x^k)$, $i = 1, \ldots, N$
2) Let
$$\mathbf{d}^k = [(\tau_1^k)^2/s^k,\ \tau_1^k/s^k,\ \ldots,\ (\tau_N^k)^2/s^k,\ \tau_N^k/s^k]$$
$$s^k = \sum_{i=1}^{N} \tau_i^k$$
3) Calculate the gain matrix
$$\mathbf{P}^k = \frac{1}{\lambda}\left(\mathbf{P}^{k-1} - \frac{\mathbf{P}^{k-1}\mathbf{d}^{kT}\mathbf{d}^k\mathbf{P}^{k-1}}{\lambda + \mathbf{d}^k\mathbf{P}^{\mathbf{k-1}}\mathbf{d}^{kT}}\right)$$
4) Update the output functions parameters
$$\mathbf{u}^k = \mathbf{u}^{k-1} + \mathbf{P}^k\mathbf{d}^{kT}(y^k - \mathbf{d}^k\mathbf{u}^{k-1})$$
5) Compute the model output $\hat{y}^k$
$$\hat{y}^k = \mathbf{d}^k\mathbf{u}^k$$

The recursive computation of $\mathbf{u}$ requires an initial estimate $\mathbf{u}^0$ of the parameters, and an initial gain matrix $\mathbf{P}^0$. In practice, it is common to set $\mathbf{u}^0 = 0$, and the gain matrix $\mathbf{P}^0 = \alpha\mathbf{I}$, where α is a sufficiently large number [19], e.g., $\alpha = 10^3$. The constant $\lambda \in [0, 1]$ is a forgetting factor that modulates the emphasis on the data sequence. If λ is large, recent data are heavily weighted, and the estimation algorithm tracks time-varying behaviors faster. However, for large values of λ, the approach becomes more susceptible to outliers and retains less memory of typical past behavior [6]. A default value of $\lambda = 0.95$ is used.

B. Long-Short-term Memory

Short-Term Memories (LSTM) are a class of recurrent deep neural networks carefully designed to capture time-dependent, sequential relationships crucial in dynamic systems modeling. Their proficiency is a result of their capacity to

retain information from previous inputs and outputs [27]. The key idea is to form a computational block comprising a memory unit that keeps its state across multiple time steps, and a nonlinear gate unit that regulates the flow of input and output information within the memory unit. The basic LSTM neural network block features three gates, respectively, input, output, and forget, with sigmoid activation functions, a hyperbolic tangent function as the block input and the output activation function, and a single memory cell [21]. A general LSTM neural network has a sequential structure, consisting of a series of such blocks, which can, in turn, be further integrated with dense (fully connected) multilayer neural networks.

The training of long short-term memory networks is conceptually similar to the typical error backpropagation methods used for parameter adjustment. LSTM training follows a two-pass learning scheme, which can process the dataset in mini-batches or iterate through the data, one instance at a time. The first pass is a forward pass that, given the input data and the weights connecting the blocks, computes the block input, the input gate, the forget gate, and the memory unit state. Additionally, signals are output from the output gate and the block output. In the second pass, a backpropagation-through-time procedure is carried out. It involves evaluating the gradients of the loss function with respect to the weights, and subsequently updating the weights that connect the memory blocks.

A comprehensive description of the LSTM neural network structure, the learning algorithm equations and steps can be found in [21] and references therein. An open source LSTM implementation in Python, which is utilized in this paper, is available in [22]. The number of parameters M in a standard LSTM network with one cell in each memory unit, ignoring biases, is $M = 4n_c^2 + 4n_i n_c + n_c n_o + 3n_c$. Here, n_c represents the number of memory cells, n_i denotes the number of input units, and n_o stands for the number of output units. Therefore, the computational complexity per time step is $\mathcal{O}(M)$. For a small number of inputs, the complexity is dominated by $n_c(n_c + n_o)$. For tasks requiring a large number of output units and a large number of memory cells to store temporal dependencies, learning may become computationally expensive [23].

C. Convolutional Neural Network

Convolutional Neural Networks (CNN) share similarities with feed-forward multilayer (MLP) neural networks. They consist of neurons with adjustable parameters, namely, adjustable weights and biases. Each neuron receives inputs, computes the dot product, and applies a nonlinear or linear transformation of the dot product. A CNN typically comprises an input layer, convolutional and pooling layers, and some fully connected layers. CNNs offer several advantages, including parameter sharing, sparse interactions, and equivariant representations, leading to fewer parameters compared to fully-connected MLP networks. Typically, a CNN has a number of convolutional layers followed by a pooling operation in the hidden layers, and a fully connected output layer. Each convolutional layer contains several filters (kernels) that produce feature maps after convolution with the input. Filters enable CNNs to handle inputs of variable size and are useful for processing data with grid-like or one-dimensional sequential structures. CNNs are usually trained using regularized back-propagation. A detailed explanation of convolutional networks, convolution and learning algorithms, and applications is given in [24]. Likewise LSTM, an open-source implementation of CNNs in Python, which is utilized in this paper, can be found in [22].

The construction of a CNN requires the specification of several hyperparameters, including the number of filters, the size of the convolution window (kernel size), the stride length of the convolution filters, padding, and the dilation rate to use for dilated convolution [22]. When modeling dynamic systems, it is common to choose unity stride and assume that the data should not violate the temporal order, which means that padding must be causal, that is, the output at time step k should not depend on input at time step $k+1$. The dilation rate is often set to unity as well, to keep the model complexity parsimonious. In this scenario, the number of parameters C in a CNN is given by $C = C_c + C_d$, where $C_c = n_f(f_s + 1)$, $C_d = (n_f + 1)$; and n_f is the number of filters, and f_s denotes the kernel size.

III. Computational Experiments

This section presents computational experiments conducted using: (i) synthetic data generated by a highly nonlinear two-dimensional dynamic system, as suggested in the modeling and prediction literature; and (ii) load data obtained from a major electrical utility in the Southeast region of Brazil, used for forecasting electricity load. These experiments aim to evaluate the performance and effectiveness of the methods described in Section II in handling synthetic and real-world data scenarios.

A. Nonlinear System Modeling and Prediction

The synthetic training data are generated using the model

$$y^k = \cos\left(\frac{y^{k-1}}{y^{k-2}+2}\right) + 0.8\sin\left((y^{k-2})^2 + y^{k-1}\right) + \eta \quad (12)$$

where $k = 1, \ldots, K$, and $\eta = N(\mu, \sigma)$ is a normally distributed noise with mean μ and standard deviation σ [14].

In [14], a series of experiments were conducted involving different sizes of training and testing data, utilizing TensorFlow [22]. One particular scenario involved 195 training instances subjected to normally distributed noise $N(0, 0.1)$. This dataset was used to train various models, including long-short-term memories (LSTM), convolutional neural networks with long short-term memory (CNN-LSTM), convolutional neural networks with bidirectional long short-term memory (CNN-BLSTM), and Gaussian process nonlinear autoregressive exogenous models (GP-NARX). The results in [14] indicate that, on average and considering 104 testing instances, LSTM, CNN-LSTM, and CNN-BLSTM achieved similar mean square errors, while GP-NARX achieved the worst performance. Therefore, this paper focuses on using LSTM, CNN, and

the Data-Driven Level Set Fuzzy Modeling (DLSM) and its recursive form (RLSM) to model and predict the output of the system described in (12). The prediction models take the form $\hat{y}^k = f(y^{k-1}, y^{k-2})$. Thus, the input variables for DLSM, RLSM, LSTM, and CNN are y^{k-1} and y^{k-2}, with the output variable being $\hat{y}^k$.

The experimental setup for evaluating the LSTM, CNN, DLSM, and RLSM is as follows. Following the data generation process and experiments outlined in the literature [14], a dataset comprising 302 instances is produced using (12). Of these, 200 instances are affected by noise with $N(0, 0.1)$ and are designated for training, while the remaining 102 noise-free instances are reserved for testing. The level set fuzzy models are configured with six rules using Gaussian membership functions. Likewise, the LSTM model consists of two memory blocks and one dense layer, trained over 50 epochs (L050) and 100 epochs (L100). The CNN employs nine filters, a unity kernel size, and a dense sigmoidal layer, trained over 50 epochs (C050) and 100 epochs (C100). The complexity of the models is characterized by their number of parameters (PAR), while their accuracy is assessed by the mean square error (MSE). Table I summarizes the results.

TABLE I
DYNAMIC SYSTEM MODELING ACCURACY AND PARAMETRIC COMPLEXITY

	L050	L100	C050	C100	DLSM	RLSM
MSE	0.0384	0.0137	0.0247	0.0194	0.0264	0.0038
PAR	43	43	37	37	36	36

Remarkably, the data-driven fuzzy model DLSM outperforms the LSTM model (L050) trained over 50 epochs. However, when training continues for an additional 50 epochs, the LSTM (L100) surpasses DLSM. Nevertheless, LSTM requires more parameters, 43 compared to DLSM's 36. The CNN exhibits a similar trend, albeit using fewer parameters than LSTM. The recursive form of the level set fuzzy model (RLSM) surpasses L100, C100, and DLSM. It is noteworthy that the results in Table I closely align with the average outcomes reported in [14]. Additionally, the fuzzy models, DLSM and RLSM, achieve comparable or superior accuracy with simpler models and faster learning procedures.

Further insights can be gained by closely examining the predictions generated by the top-performing level set fuzzy models DLSM and RLSM, along with the LSTM model, using the 100 test instances. Notice that, with the exception of a few instances, the DLSM predictions shown in Fig. 1, and the LSTM predictions shown in Fig. 2, are fundamentally similar. However, the predictions made by RLSM, as shown in Fig. 3, are notably more accurate than those of DLSM and LSTM. This observation highlights the superior performance of RLSM in capturing the underlying patterns and nonlinear dynamics of the data.

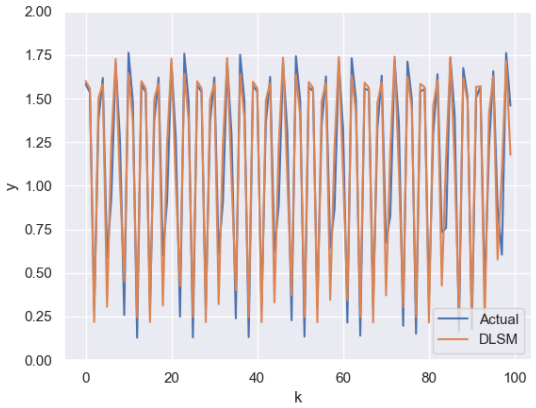

Fig. 1. Predictions of DSLM for the nonlinear dynamic system (12) (test data).

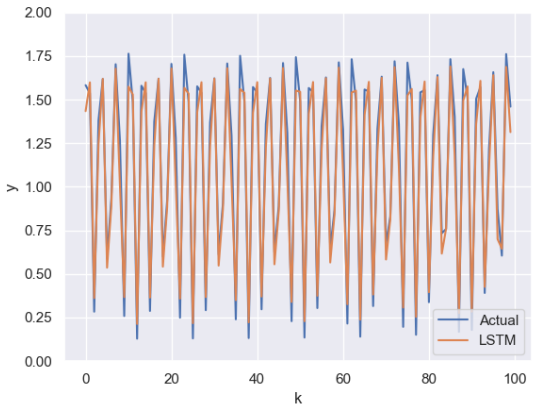

Fig. 2. Predictions of LSTM for the nonlinear dynamic system (12) (test data).

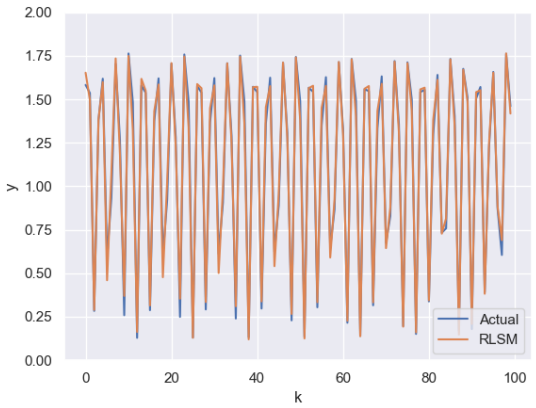

Fig. 3. Predictions of RLSM for the nonlinear dynamic system (12) (test data).

B. Electricity Load Forecasting

Short-term load forecasting plays a key role in ensuring efficient and reliable power system operations. The primary objective is to provide accurate predictions for the 24 hourly loads for the upcoming operational day, enabling better resource planning, cost optimization, and system stability. This section evaluates the performance of DLSM and RLSM models in forecasting the electricity loads for a major utility located in the Southeast region of Brazil. The dataset consists of hourly load measurements recorded over a month, with load values expressed in kilowatts per hour (kW/h). Fig. 4 shows the normalized load values (shifted and rescaled to the range [0, 1]) for the first three days of the month. The models operate as one-step-ahead forecasters, whose purpose is to predict the next load value based on lagged values from the series. The forecasting model takes the functional form:

$$\hat{y}^k = f(y^{k-1}, y^{k-2}) \tag{13}$$

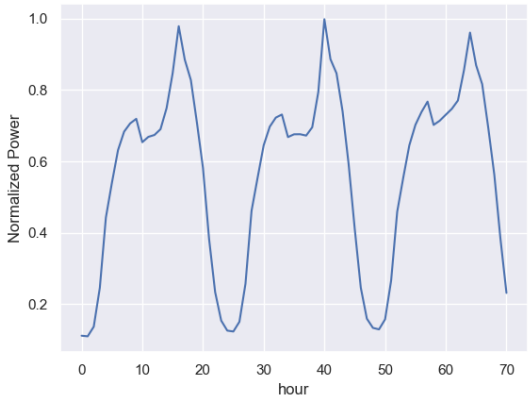

Fig. 4. Normalized electricity load dataset for the first three days of the month – showing only three cycles to highlight nuances on an expanded x-axis scale.

In this paper, the performance of the DLSM and RLSM models is compared with that of the (i) uninorm-based fuzzy neural network (UNN) introduced in [25], as well as with other models reported in the same study. These include: (ii) a single-hidden-layer feedforward neural network with logistic hidden neurons and a linear neuron in the output layer, trained with the resilient backpropagation algorithm (MLP); (iii) the adaptive neural fuzzy inference system (ANFIS), with the topology defined using the fuzzy c-means algorithm; and (iv) an extreme learning machine (ELM) feed-forward neural network with a single hidden layer and linear output layer. Additionally, (v) a long-short-term memory (LSTM) model with two memory cells and one dense layer; as well as (vi) a convolutional neural network (CNN) with six filters and a kernel size of 2×2, are included. The LSTM and CNN are trained for 100 epochs. For the MLP, ELM, and ANFIS models, the number of hidden neurons and fuzzy rules, respectively, were set to eight. Forecasting accuracy is evaluated using the root mean square error (RMSE). Table II and Table III summarize the results.

TABLE II
ELECTRICITY LOAD FORECASTING ACCURACY AND PARAMETRIC COMPLEXITY

	MLP	ANFIS	ELM	UNN
MSE	0.0766	0.0603	0.0579	0.0546
PAR	33	40	33	40

TABLE III
ELECTRICITY LOAD FORECASTING ACCURACY AND PARAMETRIC COMPLEXITY

	LSTM	CNN	DLSM	RLSM
MSE	0.0597	0.0729	0.0511	0.0286
PAR	43	25	24	24

Tables II and III show that both DLSM and RLSM outperform the other models, with RLSM clearly achieving the best overall performance. Notably, DLSM slightly outperforms UNN, ELM, LSTM, ANFIS, and MLP. The latter performs similarly to the CNN. Interestingly, previous studies have demonstrated that in electricity load forecasting, MLP can yield comparable results to some CNN models [26]. Fig. 5 and 6 illustrate the forecasts generated by the LSTM and RLSM models discussed in this paper. We observe that the RLSM model produces slightly smoother and earlier variations, with an RMSE of 0.0286, compared to the LSTM model, which presents an RMSE of 0.0597.

The results in this section highlight the effectiveness of the recursive fuzzy level-set RLSM method in capturing the dynamics in time series or data streams. The superior performance of RLSM in this application suggests that it can be a valuable tool for decision-making in power systems and other signal processing domains, offering both accuracy and interpretability. While the recursive LSTM network demonstrates strong predictive performance, especially when highly-parametric deep topologies are explored, the RLSM method distinguishes by modeling subtle changes in streaming data using fewer parameters. This highlights the potential advantages of fuzzy data-driven approaches in certain domains. Further investigation into the practical implications of incremental fuzzy models (either standalone or hybridized with neural networks) in real-time forecasting and adaptive control [9] could lead to interpretable and scalable solutions for dynamic environments.

IV. CONCLUSION

This paper has analyzed the use of state-of-the-art fuzzy and neural methods for autonomous modeling and prediction of nonlinear dynamic systems based solely on data, without relying on first-principles knowledge. In particular, data-driven level-set fuzzy RLSM modeling, along with long-short-term memory and convolutional neural network models, have demonstrated high performance in predicting unknown

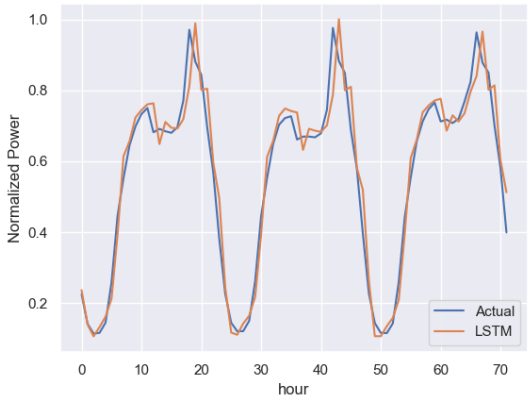

Fig. 5. LSTM electricity load forecasting for 3 consecutive days (test data).

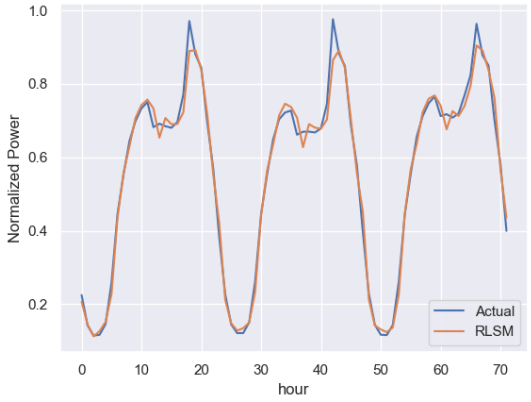

Fig. 6. RLSM electricity load forecasting for 3 consecutive days (test data).

nonlinear and potentially time-varying systems. Computational experiments conducted using a benchmark problem reported in the literature reveal that the data-driven level-set fuzzy model either competes closely with or outperforms other advanced data-driven and neural modeling approaches. Additionally, an application in electricity load forecasting further highlights that the recursive version of the level set modeling approach is particularly accurate and robust. Further research will focus on evaluating the scalability of the data-driven level set fuzzy method for longer prediction horizons and in higher-dimensional data spaces.

Acknowledgment

The authors thank the reviewers for the comments that helped improve the paper. The first author acknowledges the Ministry of Culture and Science of the State of North Rhyne-Westphalia for grant no. NW21-059D, SAIL project. The last author is grateful to the Brazilian National Council for Scientific and Technological Development for grant 302467/2019-0.

References

[1] K. Astrom, and B. Wittenmark, Computer Controlled System: Theory and Desing, 3rd ed., Mineola, Dover, 2011.
[2] C. Bishop, Pattern Recognition and Machine Learning, New York, Springer, 2006.
[3] T. Mitchell, Machine Learning, New York, MacGraw-Hill, 1997.
[4] N. Narendra, and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," IEEE Trans. on Neural Networks, vol. 1, pp. 4–27, March 1990.
[5] N. Narendra, and K. Parthasarathy, "Neural networks and dynamical systems," J. of Approximate Reasoning, vol. 6, pp. 109–131, February 1992.
[6] R. Jang, C. Sun, and E. Mizutani, Neuro-fuzzy and Soft Computing, Upper Saddle River, Prentice hall, 1997.
[7] W. Pedrycz, and F. Gomide, Fuzzy Systems Engineering: Toward Human-Centric Computing, Hoboken, John Wiley, 1997.
[8] D. Solomatine, and A. Ostfeld, "Data-driven modelling: Some past experiences and new approaches," J. of Hidroinformatics, vol. 10, pp. 3–22, 2008.
[9] I. Skrjanc, J. Iglesias, A. Sanchis, D. Leite, E. Lughofer, and F. Gomide, "Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A survey," Information Sciences, vol. 490, pp. 344–368, July 2019.
[10] K. Murphy, Machine Learning: A Probabilistic Approach, Cambridge, MIT Press, 2012.
[11] A. Deshmuk, and J. Alisson, "Design of dynamic systems using surrogate models of derivative functions," J. of Mechanical Design, vol. 139, pp. 3–22, October 2017.
[12] D. Gedon, N. Wahlstron, T. Shon, and L. Jung, "Deep state models for nonlinear system identification," IFAC PaperOnline, vol. 54, pp. 481–486, September 2021.
[13] K. Kowaja, M. Shcherbatyy and W. Hardle, "Surrogate models for optimization of dynamical systems," in Foundations of Modern Statistics, vol. 425, D. Belomestny, C. Butucea, Enno Mammen, E. Moulines, and M. Reiss, Eds. Springer, 2023.
[14] Y. Zhao, C. Jiang, M. Viega, M. Todd, and Z. Hu, "Surrogate modeling of nonlinear dynamic systems: A comparative study," J. of Computing and Information Science in Engineering, vol. 23, May 2022.
[15] R. Yager, "Alternative mechanisms for fuzzy logic control," Proc. of the International Congress of Biomedical Fuzzy Systems, Tokio, Japan, 91–94, 1991.
[16] D. Leite, F. Gomide, and R. Yager, "Data driven fuzzy modeling using level sets," Proc. FUZZ-IEEE, Padova, July 2022.
[17] S. Serre, Matrices: Theory and Applications, New York, Springer, 2007.
[18] V. Stanojevic, L. Kazakovtsev, P. Stanimirovic, N. Rezovz, and G. Shkaberina, "Calculating the Moore-Penrose generalized inverse on massively parallel systems," Algorithms, vol. 15, September 2022.
[19] L. Jung, System Identification: Theory for the User, London, Pearson, 1999.
[20] L. Maciel, and F. Gomide, "Recursive level set fuzzy modeling," Proc. 2024 IEEE EAIS, Madrid, May 2024.
[21] K. Greff, and R. Srivatana, and J. Koutinik, and B. Steunebrink, and J. Schidhuber "LSTM: A search space odissey," IEEE Trans. on Neural Networks and Learning Systems, vol. 28, pp. 2222–2232, October 2017.
[22] Tensorflow, https://www.tensorflow.org, 2024.
[23] H. Sak, A. Senior, and P. Basufaya, "Long short-term mempory based recurrent neural network architectures for large vocabulary speech recognition," https://arxiv.org/pdf/1402.1128.pdf, 2014.
[24] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, Cambridge, MIT Press, 2016.
[25] A. Lemos, W. Caminhas, F. Gomide, "A fast learning algorithm for uninorm-based fuzzy neural networks," Proc. of the NAFIPS, Berkeley, 2012.
[26] K. Amarasinghe, D. Marino, and M. Manic, "Deep neural networks for energy load forecasting," Proc. of the 26th IEEE Int. Symposium on Industrial Electronics, Edinburgh, 1483–1488, 2017.
[27] G. Van Houdt, C. Mosquer, and G. Napoles "A review on the long short-term memory model," Aritificial Intelligence Review, vol. 53, pp. 5929–5955, May 2020.
[28] M. Zhu, B. Chang, and C. Fu, "Convolutional neural networks combined with Runge-Kutta methods," Neural Computing and Applications, vol. 35, pp. 1629–1643, October 2022.