Very Short Term Active And Reactive Load Forecasting Based On Complex-Valued LSTM Neural Network

Clemente Thompson

Abstract—**This paper proposes a new recurrent complex-valued LSTM-based neural network model that performs ultra/very short-term complex load forecasting of active and reactive power consumption at a distribution level 15 minutes ahead. This model was trained, validated, and tested using the active and reactive profiles of ten distinct rural loads from the SimBench benchmark based on German grid data. These profiles contain year-round active and reactive power consumption data for multiple categories of loads with 15-minute resolution.**

In parallel, a real-valued analogous model that is only able to perform uni-variate forecasts was subjected to the same input data and learning scheme (including the same hyper-parameters) with the objective of measuring the incremental benefits of having a complex-valued forecasting model rather than a real-valued one. The results obtained strongly favor the complex-valued model performance with a significance of at most 2.15% and are explained by the ability of the model of leveraging mutual correlations between active and reactive profiles to execute a complex load forecast. The bulk of accuracy improvements were in reactive load forecasts. These results hint that the complex domain could be a promising framework for performing time-series-based multi-task load forecasting.

Index Terms—**Active Load Forecasting, Complex-Valued Neural Networks, Deep Learning, Load Forecasting, LSTM Networks, Reactive Load Forecasting.**

I. Introduction

LOAD forecasting involves predicting future electricity demand for a specific time frame and level of load aggregation based on historical data. This data can be previous consumption, weather patterns, economic indicators, seasonality, and other factors that may affect electric energy demand [1]. These forecasts are crucial for planning and operation in the energy industry, as it helps utilities, power system operators, and other electricity market agents to ensure that the power supply is capable of meeting anticipated demand in a secure, stable, and uninterrupted way.

In operations management, various types of demand forecasts are used for specific organizational purposes. In the electricity sector, short-term forecasts are used for daily and weekly operations, medium-term forecasts for tactical planning, and long-term forecasts for strategic designs. Examples include scheduling electricity generation and transmission for the short-term, fuel purchasing and maintenance for the medium term, and setting sustainability goals that involve infrastructure expansion and technology adoption for the long term [2].

As the energy industry shifts towards renewable energy sources and smart grid technology, the need for accurate ultra short-term load forecasts (few seconds up to few hours [1], [2]) has become essential for ensuring the real-time control and management of power grids. This includes implementing strategies such as preventive control, emergency management, and demand control to improve the efficiency and reliability of grid operations, as well as to increase socio-economic benefits [3], [4]. Because the focus of this research is to propose an ultra-short-term electric load forecasting model, from now on, when addressing the load forecasting concept throughout this paper, it is implied to have an ultra-short lead time unless explicitly stated otherwise. Now, the model will also perform reactive load forecasts. The rationale for including reactive power forecasts in a multi-variable/multi-task approach, as outlined in the following three paragraphs, is based on the research of A. Bracale, et al [5].

Forecasters in power systems have typically centered around predicting active loads, as active power is the rate of energy that can do average net positive work in time. The latter goes to the extent that load forecasting is used interchangeably with active load forecasting in both academia and industry. However, advances in computing power and the widespread adoption of smart grid technology, which enable the management of both active and reactive power flows, have made reactive load forecasting not only possible but necessary within the operational realm of power systems i.e. to manage and control instant and short-term activities. As a result, it is now important to consider both active and reactive loads in forecasting efforts.

Some examples of grid operations that require nodal estimations of reactive power are Volt/VAr optimization [6], harmonic compensation [7], and optimal energy dispatch [8], [9]. The numerous power converters distributed throughout the grid can be controlled to provide local compensation for reactive power [10], thereby increasing the grid's overall power transmission capacity and reducing losses [11]. It is worth noting that reactive power support is a valuable ancillary service that is compensated in several countries [12] and that there have been proposals for the creation of reactive power markets in the literature as [13], [14] put forward.

Practitioners are increasingly interested in managing reactive power, but there has been slow progress in the literature on reactive power forecasting. Reactive power forecasts are often

Clemente Thompson is a student opting for a Master's Degree in Engineering Sciences within Departamento de Ingeniería Eléctrica, Pontificia Universidad Católica de Chile. e-mail:cmthompson@uc.cl

created by adjusting active power forecasts using experience-based knowledge or average power factor corrections, rather than directly forecasting reactive power. Few papers have directly addressed reactive power forecasting in contrast with the extensive real load forecasting bibliography. The majority of literature on reactive forecasting is based on the application of artificial neural networks (ANNs), either at domestic nodes [15], disaggregated levels [16] or at bulk supply nodes [17]. In addition to the aforementioned ANNs, other techniques for predicting reactive power include fuzzy approaches, support vector regression (SVR), and statistical methods like autoregressive models applied to time series, and other types of regression analysis [5].

Complex valued neural networks (CVNNs) are neural networks that handle complex inputs, produce complex outputs, have complex weights, and use complex activation functions. According to the studies [18]–[20], CVNNs are more efficient and have greater generalization ability than their real-valued counterparts in certain problems, such as the insightful XOR problem [21]. Some disciplines that have seen successful deployments of CVNNs are computer vision [22], radar [23], speech recognition [24], telecommunications [25] and wind forecasting [26]. CVNNs are well-suited for solving wave signal processing problems as they can directly extract wave information, such as amplitude and phase.

The intuition behind pursuing CVNNs solutions can be found in the complex algebra itself, specifically in complex multiplication. A CVNN is not equivalent to 2-dimensional real-valued ANN. As J. Bassey et al. explain in their survey of complex-valued neural networks [19], “in fact, it has been shown that this is not the case [18], [27], because the operation of complex multiplication limits the degree of freedom of the CVNNs at the synaptic weighting. This suggests that the phase-rotational dynamics strongly underpin the process of learning”.

Complex power is a parameterization that arises from a physical electromagnetic wave phenomenon in alternating current power systems; that is, the product of instant node reference voltage waveform by instant leading/lagging current delivery/consumption waveform respectively. Moreover, the electrical circuit theory behind grid analysis is a simplification of electromagnetic field theory under the assumption of low-frequency electromagnetic waves encountered in power systems.

The correlation between active and reactive power can be captured as shown in [28]. Furthermore, the works [5], [29] conclude that correctly modeling the mutual correlation between active and reactive power allows for enhancing the performance of a forecasting system. The goal of this paper is to introduce a new recurrent CVNN-based supervised learning scheme of time series for performing multi-task load forecasting in a very short-term lead time (15 minutes ahead) by utilizing both active and reactive aspects of load as inputs to a CVNN, under the hypothesis that CVNNs are effective at learning and representing complex power temporal patterns in a synergistic manner. To measure the impact of the new complex model results, they will be compared against the results obtained from a real-valued equivalent network with an analogous architecture and hyper-parameters. This equivalent network will perform a single task or uni-variate forecasts either on active load profiles series or reactive load profiles series. Hopefully, success in achieving the stated paper objective will serve as motivation for further research on this topic, as well as exploring CVNNs applications in other power system control and operations beyond load forecasting.

II. Methods

A. Data collection

The load profiles used in this study were sourced from SimBench, a collection of open-source and comprehensive grid datasets for use in the comparison and analysis of grid planning and operation methods. SimBench is based on grid data and topology from Germany and includes datasets for various voltage levels. Additionally, this benchmark provides time series data for active and reactive power load, generation, and storage, covering an entire year with 15-minute resolution data points (i.e. 35,136 discrete values for each profile) [30].

The procedure for compiling the load profile data for the SimBench benchmark is outlined in the conference paper *SimBench: Open Source Time Series of Power Load, Storage, and Generation for the Simulation of Electrical Distribution Grids* [31]. The relevant information for this investigation is that the load profiles were taken from a medium-voltage distribution level in a rural grid (10/20 kV) with ID `1-MV-rural-0-sw` in the SimBench system. The model was trained, validated, and tested using the active and reactive profiles, represented as uni-dimensional complex-valued time series, from ten unique loads[1] independently: four commercial loads (G0-A, G0-M, G3-A, G3-M), two agriculture-related loads (L0-A, L2-M), three household rural loads (Rural-1, Rural-2, Rural-3), and one household semi-urban load (SemiUrban-4).

The statistical biannual characterization of these loads is displayed in Table I. In particular, the statistical descriptors are the sample mean, sample standard deviation, and load factor for each profile (real-valued time series). Finally, the correlation factor between the active and reactive profiles is presented for each load to provide further insight into temporal behavior and to assert the high correlation existing in active/reactive profile pairs [28].

The twenty real-valued profiles of the ten aforementioned loads of interest are obtained through the SimBench API. Then, the active and reactive profiles are stored as a unidimensional complex-valued discrete sequence for each load as shown in Eq. (1). Due to a lack of computational resources, using the full-year profile data of a load is not possible. So, each yearly load profile will be divided equally into two different load profiles, one for each semester.

In total, the model will be trained, validated, and tested two times for each load, for each of the ten loads displayed on Tab. I resulting in twenty different load profiles to try the model on. To sum up, the proposed model is structured to handle semi-annual data for a single load making it irrelevant to index equations and procedures according to load name

[1] It is not clear if some of these loads are a true consumption endpoint or they represent a cluster of smaller loads.

TABLE I: `1-MV-rural-0-sw` grid available profiles yearly characterization.

Load	Semester	Active Profile			Reactive Profile			Correlation
		Mean [kW]	Standard deviation [kW]	Load factor [%]	Mean [kVAr]	Standard deviation [kVAr]	Load factor [%]	[%]
G0-A	I	332.01	177.29	36.75	381.36	88.86	51.20	41.22
	II	361.68	183.47	36.17	438.79	108.43	48.81	52.45
G0-M	I	373.96	176.81	37.40	307.65	161.01	26.79	48.18
	II	407.18	199.09	43.04	3344.35	208.00	32.27	67.1
G3-A	I	419.31	108.43	41.93	363.16	215.68	27.77	88.27
	II	427.07	102.46	48.64	391.01	196.34	30.80	90.44
G3-M	I	463.98	109.10	46.40	250.79	105.75	26.20	67.2
	II	496.91	101.83	51.95	256.70	109.80	32.72	61.03
L0-A	I	329.31	142.96	33.34	567.74	261.34	38.98	82.36
	II	325.61	137.61	32.56	625.82	274.08	37.06	83.63
L2-M	I	319.06	145.69	32.34	329.13	149.87	28.50	60.77
	II	325.67	154.80	32.57	384.39	209.11	32.79	75.37
Rural-1	I	293.46	110.80	31.05	333.92	73.20	55.43	80.89
	II	288.61	106.29	40.57	319.58	77.52	52.89	74.27
Rural-2	I	157.26	63.76	38.00	97.52	50.84	24.31	70.1
	II	146.43	61.12	35.02	96.00	50.76	24.17	73.63
Rural-3	I	132.30	67.18	30.92	61.74	44.99	14.38	64.03
	II	116.14	62.86	26.26	53.63	40.98	12.99	65.25
Semiurb-4	I	193.36	84.65	39.86	107.66	44.77	30.52	84.75
	II	182.38	81.43	37.71	99.11	43.27	27.62	78.29

and semester. In any case the input data size conflicts with the lead time of the forecast. Consequently, the complex time series representing the complex load profile takes the vector form of Eq. (2) where $L + 1 = 17,568$ datapoints.

$$s_t = p_t + jq_t \tag{1}$$

$$\boldsymbol{s} = \begin{bmatrix} x_0 & x_1 & \dots & x_{L-1} & x_L \end{bmatrix}^\top \in \mathbb{C}^{L+1} \tag{2}$$

The model proposed falls within the supervised learning paradigm. This implies the restructuring of available data to form labeled examples and subsequently, split those example-label pairs onto a training set, validation set, and testing set. In order to get the example vectors from sequence $\boldsymbol{s}$ a sliding window of n samples with unitary stride is applied forming an ordered set $\mathbb{X} = \left\{\boldsymbol{x}^{(0)}, \dots, \boldsymbol{x}^{(m-1)}\right\}$ of m sequence vectors $\boldsymbol{x}^{(i)} \in \mathbb{C}^n$ displayed in its expanded form in Eq. (3). The labels for each window are derived from the data of the time step immediately following the last entry of each example vector $\boldsymbol{x}^{(i)}$. Therefore, the ground truth vector $\boldsymbol{y} \in \mathbb{C}^m$ is structured as shown in Eq. (4) reflecting a positional correspondence between the i-th example of set $\mathbb{X}$ and the i-th entry of target vector $\boldsymbol{y}$.

$$\mathbb{X} = \left\{ \begin{bmatrix} x_0 \\ \vdots \\ x_{n-1} \end{bmatrix}, \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \dots, \begin{bmatrix} x_{L-n+1} \\ \vdots \\ x_L \end{bmatrix} \right\} \tag{3}$$

$$\boldsymbol{y} = \begin{bmatrix} x_n & x_{n+1} & \dots & x_{L+1} \end{bmatrix}^\top \tag{4}$$

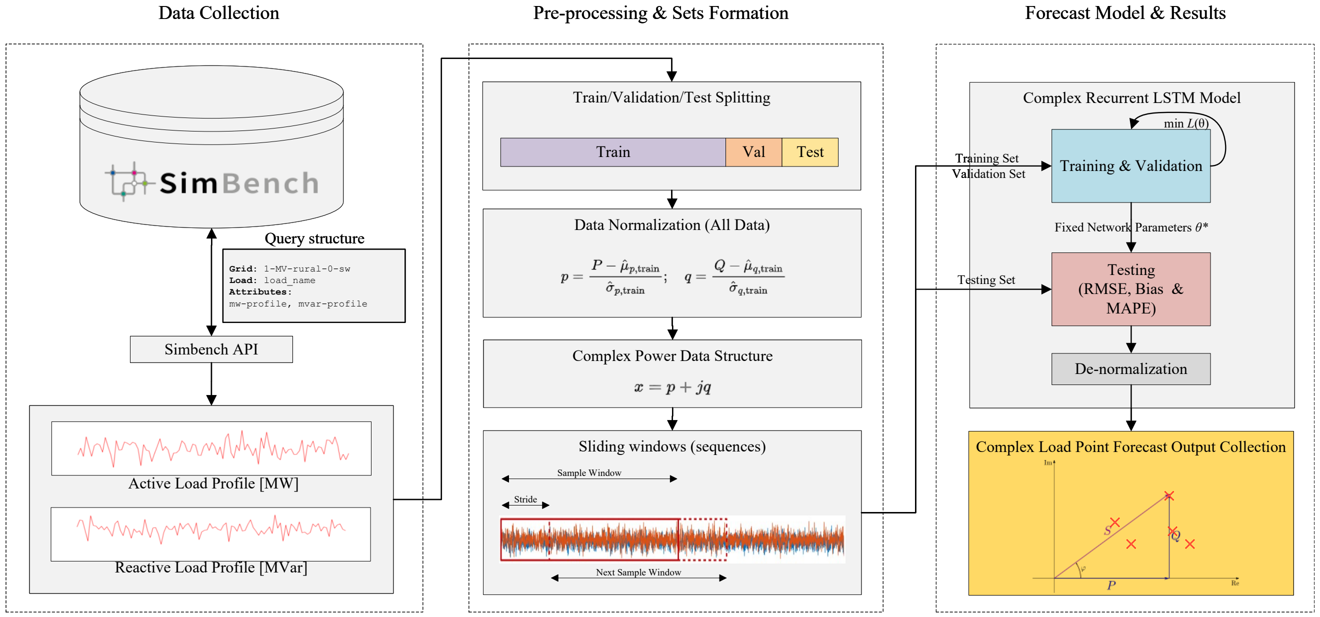

Data Collection
Pre-processing & Sets Formation
Forecast Model & Results
SimBench
Query structure
Grid: 1-MV-rural-0-sw
Load: load_name
Attributes:
mw-profile, mvar-profile
Simbench API
Active Load Profile [MW]
Reactive Load Profile [MVar]
Train/Validation/Test Splitting
Train
Val
Test
Data Normalization (All Data)
$p = \frac{P - \hat{\mu}_{p,\text{train}}}{\hat{\sigma}_{p,\text{train}}}; \quad q = \frac{Q - \hat{\mu}_{q,\text{train}}}{\hat{\sigma}_{q,\text{train}}}$
Complex Power Data Structure
$\boldsymbol{x} = \boldsymbol{p} + j\boldsymbol{q}$
Sliding windows (sequences)
Sample Window
Stride
Next Sample Window
Complex Recurrent LSTM Model
min $L(\theta)$
Training Set
Validation Set
Training & Validation
Fixed Network Parameters θ^*
Testing Set
Testing
(RMSE, Bias &
MAPE)
De-normalization
Complex Load Point Forecast Output Collection
Im
Re
S
Q
P
φ

Fig. 1: Ultra short-term active and reactive load forecasting system overview.

The procedure for splitting $\mathbb{X}$ and $\boldsymbol{y}$ is carried out to obtain the training, validation, and test subsets in a proportion $u : v : w$ respectively. The splitting is ordered i.e. first u elements of $\mathbb{X}$ and $\boldsymbol{y}$ are assigned for training purposes, the next v elements are assigned to validation, and the remaining w to testing. It follows that $u + v + w = m$.

It is worth noting that to make it easier for the model to fit input data a normalization process is applied before explicitly forming the training, validation, and testing sets. This normalization consists of the subtraction of the sample mean and subsequent division by the sample standard deviation of the training input time series (say $\hat{\mu}_{\text{train}}, \hat{\sigma}_{\text{train}}$) (i.e. parameters computed using only u data points) for the entire time-series of length $L + 1$. So, once the model performs a forecast, that number must be stretched back by $\hat{\sigma}_{\text{train}}$ and up-shifted by $\hat{\mu}_{\text{train}}$.

B. Neural network model

The main architecture of the proposed network model consists of a stack of five identical complex recurrent layers (CRLs) as can be seen in Fig. 2a. The inner composition of each layer can be seen in Fig. 2b. So, to understand the entire CRL as a block it is necessary to understand its foundations; that is, review real-valued long short term memory (LSTM) layers, comprehend complex matrix multiplication as a composition of parallel real operations and finally, revise the chosen rectified linear unit (ReLU) type in complex space.

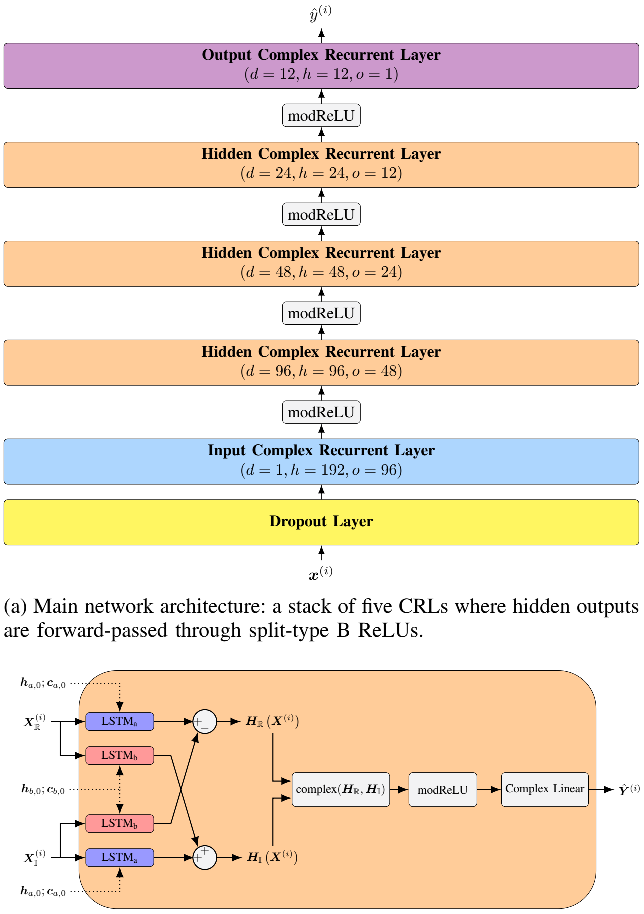
The basic building block employed to construct the proposed CRL is a real-valued LSTM network which is a type of recurrent neural network (RNN) that has been proven to consistently generate high accuracy load forecasts [32]. The LSTM cell was introduced in [33]. An LSTM network consists of a series of memory cells and gates that control the flow of information in and out of these cells in a selective manner. The purpose of a long short memory cell is to store representational information in the form of latent vectors at every step of a sequence. Specifically, information is encoded in hidden $\boldsymbol{h}_t$ and cell states $\boldsymbol{c}_t$ vectors of the sequence across time, making them suitable to recognize long-term dependencies as well as short-lived ones in time series data. This cell is illustrated in Fig. 3. The forget gate, input gate, and output gate equations are given in Eqs. (7, 8, 9) respectively. Then, the candidate cell state calculation and cell and hidden states updates are given in Eqs. (10, 11, 12).

These mechanisms heavily rely on sigmoid and hyperbolic tangent activations defined in Eqs. (5, 6) as well as the learnable weights $\boldsymbol{W}$ and $\boldsymbol{U}$ matrices and bias vectors $\boldsymbol{b}$ participating in the affine combinations of the new input entry $\boldsymbol{x}_t$ and previous hidden state $\boldsymbol{h}_{t-1}$ as can be seen in the argument of gate equations. These parameters remain constant over time once the learning process is finished.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{5}$$

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{6}$$

$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{b}_f\right) \tag{7}$$



$\hat{y}^{(i)}$
Output Complex Recurrent Layer
$(d = 12, h = 12, o = 1)$
modReLU
Hidden Complex Recurrent Layer
$(d = 24, h = 24, o = 12)$
modReLU
Hidden Complex Recurrent Layer
$(d = 48, h = 48, o = 24)$
modReLU
Hidden Complex Recurrent Layer
$(d = 96, h = 96, o = 48)$
modReLU
Input Complex Recurrent Layer
$(d = 1, h = 192, o = 96)$
Dropout Layer
$\boldsymbol{x}^{(i)}$
$\boldsymbol{h}_{a,0}; \boldsymbol{c}_{a,0}$
$\boldsymbol{X}_{\mathbb{R}}^{(i)}$
LSTM$_\text{a}$
LSTM$_\text{b}$
$\boldsymbol{h}_{b,0}; \boldsymbol{c}_{b,0}$
LSTM$_\text{b}$
$\boldsymbol{X}_{\mathbb{I}}^{(i)}$
LSTM$_\text{a}$
$\boldsymbol{h}_{a,0}; \boldsymbol{c}_{a,0}$
+
−
+
+
$\boldsymbol{H}_{\mathbb{R}}\left(\boldsymbol{X}^{(i)}\right)$
$\boldsymbol{H}_{\mathbb{I}}\left(\boldsymbol{X}^{(i)}\right)$
complex$(\boldsymbol{H}_{\mathbb{R}}, \boldsymbol{H}_{\mathbb{I}})$
modReLU
Complex Linear
$\hat{\boldsymbol{Y}}^{(i)}$

(a) Main network architecture: a stack of five CRLs where hidden outputs are forward-passed through split-type B ReLUs.

(b) CRL inner architecture.

Fig. 2: Depictions of both main network architecture (a) and sub-layer structures (b).

TABLE II: LSTM involved variables and parameters at time step t.

Name	Variable	Domain
Input vector	$\boldsymbol{x}_t$	$\mathbb{R}^d$
Forget vector	$\boldsymbol{f}_t$	$[0, 1]^h$
Input gate vector	$\boldsymbol{i}_t$	$[0, 1]^h$
Output gate vector	$\boldsymbol{o}_t$	$[0, 1]^h$
Candidate cell state vector	$\tilde{\boldsymbol{c}}_t$	$[-1, 1]^h$
Cell state vector	$\boldsymbol{c}_t$	$[-1, 1]^h$
Hidden state vector	$\boldsymbol{h}_t$	$[-1, 1]^h$
Input associated weights	$\boldsymbol{W}$	$\mathbb{R}^{d \times h}$
Hidden associated weights	$\boldsymbol{U}$	$\mathbb{R}^{h \times h}$
Input-hidden bias combination	$\boldsymbol{b}$	$\mathbb{R}^h$

$$\boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{b}_i\right) \tag{8}$$

$$\boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{b}_o\right) \tag{9}$$

$$\tilde{\boldsymbol{c}}_t = \tanh\left(\boldsymbol{W}_c \boldsymbol{x}_t + \boldsymbol{U}_c \boldsymbol{h}_{t-1} + \boldsymbol{b}_o\right) \tag{10}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \otimes \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \otimes \tilde{\boldsymbol{c}}_t \tag{11}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \otimes \tanh\left(\boldsymbol{c}_t\right) \tag{12}$$

The superscripts d and h correspond to the number of features in the input vector (not to be confused with the length of the series) and the number of hidden units or the dimension of the vector $\boldsymbol{h}_t$, respectively.

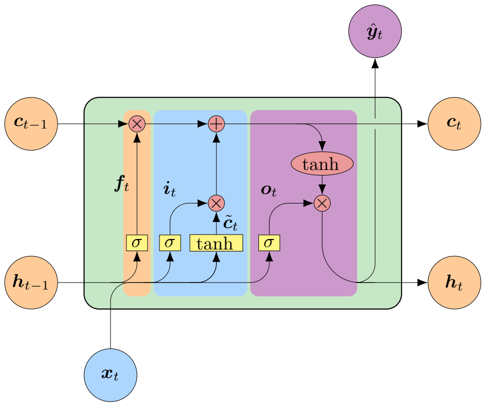


Fig. 3: LSTM cell architecture. Converging arrows imply concatenation and diverging arrows mean replication. So, the desired output $\hat{\boldsymbol{y}}_t$ is the hidden state $\boldsymbol{h}_t$. Usually, some intermediate layer is added in between in order to produce a decoded output from the hidden state.

The generalized function LSTM: $\mathbb{R}^{n\times d} \mapsto \mathbb{R}^{n\times h}$, is defined as the causal sequence of transformations described in Eqs. (7, 8,9, 10, 11, 12). This function maps an input real-valued sequence $\boldsymbol{X}^{(i)}$ of n time steps with d features[2] for each time step, to a latent real-valued sequence matrix $\boldsymbol{H}^{(i)}$ of $n \times h$, where there is a different representational entry of h dimensions for each of the n time steps. It's worth noting that the last row of $\boldsymbol{H}^{(i)}$ has the best encoding representation of the input sequence matrix $\boldsymbol{X}^{(i)}$, following the rules of recurrent networks. When the LSTM non-linear transformation is applied to an entire real-valued sequence matrix, it produces a real-valued latent matrix $\boldsymbol{H}$, as shown in Eq. (13).

$$\boldsymbol{H} = \text{LSTM}(\boldsymbol{X}) \tag{13}$$

So far, all operations have been computed in the real domain. The proposed model utilizes two independent LSTM networks, LSTM$_\text{a}$ and LSTM$_\text{b}$, to achieve holistic learning of complex load temporal dependencies. To do so, both networks examine both the real and imaginary parts of the input sequence independently and in tandem. This means that in one forward pass of complex input, each network revises two real-valued inputs (the real and imaginary parts) with tied network parameters, operating in a siamese manner [34]. By performing algebraic operations on the outputs of these networks, a complex output is structured. Additionally, by selecting a specific set of operations on the resulting hidden matrices of each network associated with real inputs, a complex multiplication behavior pattern emerges.

The product of two complex numbers x and w in rectangular form can be computed using Eq. (14).

$$xw = x_\mathbb{R} w_\mathbb{R} - x_\mathbb{I} w_\mathbb{I} + j(x_\mathbb{I} w_\mathbb{R} + x_\mathbb{R} w_\mathbb{I}) \tag{14}$$

This behavior of complex multiplication can be extended to complex matrices in $\mathbb{C}^{n\times m}$ space by following the distributive properties dictated by matrix algebra, with the distinction that matrix multiplication is not commutative. Thus, a complex linear transformation $\text{T}(\boldsymbol{X}) = \boldsymbol{Z}\boldsymbol{X}$ with kernel $\boldsymbol{Z} = \boldsymbol{A} + j\boldsymbol{B}$ can be written as Eq. (15).

$$\text{T}(\boldsymbol{X}) = \boldsymbol{A}\boldsymbol{X}_\mathbb{R} - \boldsymbol{B}\boldsymbol{X}_\mathbb{I} + j(\boldsymbol{B}\boldsymbol{X}_\mathbb{R} + \boldsymbol{A}\boldsymbol{X}_\mathbb{I}) \tag{15}$$

Each term of Eq. (15) can be interpreted as a separate real-valued linear transform. Therefore, $\text{T}(\boldsymbol{X})$ is a composition of real-valued linear transformations that behaves like complex multiplication in complex matrix space. If a non-linearity Φ is applied to each term of Eq. (15), then Eq. (16) can be viewed as a composition of individual nonlinear units, giving rise to a neural network pattern.

$$\Phi(\boldsymbol{X}) = \Phi(\boldsymbol{A}\boldsymbol{X}_\mathbb{R}) - \Phi(\boldsymbol{B}\boldsymbol{X}_\mathbb{I}) + j(\Phi(\boldsymbol{B}\boldsymbol{X}_\mathbb{R}) + \Phi(\boldsymbol{A}\boldsymbol{X}_\mathbb{I})) \tag{16}$$

We can use the shorthand notation in Eq. (16) and rewrite it as Eq. (17) for brevity.

$$\Phi_T(\boldsymbol{X}) = \Phi_\text{a}(\boldsymbol{X}_\mathbb{R}) - \Phi_\text{b}(\boldsymbol{X}_\mathbb{I}) + j(\Phi_\text{b}(\boldsymbol{X}_\mathbb{R}) + \Phi_\text{a}(\boldsymbol{X}_\mathbb{I})) \tag{17}$$

Instead of Φ_a and Φ_b being individual non-linear operators, we can use them as whole forward passes through independent LSTM networks, LSTM$_\text{a}$ and LSTM$_\text{b}$, as hinted in Eq. (13). These networks are a series of compositions of individual linear transformations passed through activations in of themselves. By doing so, we can construct the output in Eq. (18, 19) to mimic a recurrent complex multiplication. This way, a neural network pattern that results in a complex output can be generated. This set of operations to imitate complex multiplication was already successfully implemented in creating a phase-aware speech enhancement model [35]. The authors coined the term "Realisation of Complex LSTM (RC-LSTM)" to encapsulate Eqs. (18, 19).

$$\boldsymbol{H}_\mathbb{R}(\boldsymbol{X}) = \text{LSTM}_\text{a}(\boldsymbol{X}_\mathbb{R}) - \text{LSTM}_\text{b}(\boldsymbol{X}_\mathbb{I}) \tag{18}$$
$$\boldsymbol{H}_\mathbb{I}(\boldsymbol{X}) = \text{LSTM}_\text{a}(\boldsymbol{X}_\mathbb{I}) + \text{LSTM}_\text{b}(\boldsymbol{X}_\mathbb{R}) \tag{19}$$

The modReLU activation is employed in the proposed model and acts as a split-type B activation; meaning that it only acts in the magnitude of the input preserving its original phase. This activation was proposed in [24]. It applies the real-valued ReLU shown in Eq. (20) to the magnitude of the input and is formalized as Eq. (21). This function is designed such that if the magnitude of the complex input falls within a circle of radius of a learnable parameter b centered around the complex plane origin, the activation yields zero. Otherwise, the input magnitude is shifted by b, and the input angle is preserved. The modReLU activation acts element-wise, implying that for each entry of an input tensor, there will be a distinct trainable parameter b.

$$\text{ReLU}(x) = \max(0, x) \tag{20}$$

$$\text{modReLU}(z) = \text{ReLU}(\|z\| + b)\frac{z}{\|z\|} \tag{21}$$

In Fig. 2b can be appreciated that complex output matrix $\boldsymbol{H} \in \mathbb{C}^{n\times h}$ is composed and then passed through a modReLU activation that acts element-wise forming the $\boldsymbol{M} \in \mathbb{C}^{n\times h}$ tensor. Finally, a complex affine mapping denoted Complex Linear: $\mathbb{C}^{n\times h} \mapsto \mathbb{C}^{n\times o}$ is applied to the output of the modReLU module to obtain output matrix $\hat{\boldsymbol{Y}} \in \mathbb{C}^{n\times o}$ with

[2] For unidimensional time series input, $d = 1$.

$\boldsymbol{W}$ kernel matrix $\in \mathbb{C}^{o \times h}$ and $\boldsymbol{B}$ offset matrix $\in \mathbb{C}^{n \times o}$. This transform corresponds to a fully connected layer and is described in Eq. (22).

$$\hat{\boldsymbol{Y}} = \boldsymbol{M}\boldsymbol{W}^{\top} + \boldsymbol{B} \tag{22}$$

The superscript o denotes the output vector dimension of each timestep up to n. In black box terms, the complex recurrent layer transforms a d-dimensional n-length sequence into an o-dimensional n-length sequence. The entire layer, CRL: $\mathbb{C}^{n \times d} \mapsto \mathbb{C}^{n \times o}$, is parameterized by learnable $\boldsymbol{\theta}$ and can be summarized in Eq. (23).

$$\hat{\boldsymbol{Y}}(\boldsymbol{X}) = \mathrm{CRL}(\boldsymbol{X}; \boldsymbol{\theta}) \tag{23}$$

To encode the necessary information for complex power output forecasts, two types of encoding occur in the process. The first type is temporal encoding, which is generated by the hidden matrices and is explained by the recurrence of the model. The second type is forward bottleneck encoding, which originates from the stacking of layers in the general architecture shown in Fig. 2a. This encoding action adds non-linearities between layers and aims to achieve a deep representation of recurrent latent vectors that is condensed enough. As the layer depth increases, the hidden dimensions get smaller, making it easier to retrieve the complex power output forecast from the condensed representation through a complex linear transformation. It is worth remembering that in recurrent models the point forecast of interest corresponds to the n-th (last) entry of the output vector recollected from the output layer. Additionally, a dropout regularization layer was added prior to the input CRL to prevent overfitting of training data [36].

C. Loss function

When the training loop is executed, the selected optimizer corresponds to adaptive moment estimation (ADAM), which is a specific variation of the stochastic gradient descent (SGD) algorithm [37]. This means that the model is trained using random batches (or samples from the training set) stored in the subset $\mathbb{B} \subset \mathbb{X}_{\text{train}}$ of cardinality $\mathrm{card}(\mathbb{B})$ to improve memory usage efficiency. Therefore, the back-propagated gradients of errors that result in network parameter modifications are computed only using a subset of training examples. It is said that when the network has seen the entirety of the training examples, a training epoch has passed.

The complex error or difference between the network estimation $\hat{y}$ and the ground truth value y is arbitrarily chosen as seen in Eq. (24).

$$\mathrm{err}(\hat{y}, y) = \hat{y} - y \tag{24}$$

The proposed loss function $L : \mathbb{C} \mapsto \mathbb{R}$ is given in Eq. (25), was successfully used in [35], and represents the assumed independent total sum of active forecast mean squared error (MSE) and reactive forecast MSE for each sample contained in the batch. MSE is the cross-entropy (or negative log-likelihood) probabilistic distance under Gaussian assumptions of theoretical data generating distribution [38].

$$L = \frac{1}{\mathrm{card}(\mathbb{B})} \sum_{i=0}^{\mathrm{card}(\mathbb{B})-1} \mathrm{err}(\hat{y}^{(i)}, y^{(i)})\, \mathrm{err}(\hat{y}^{(i)}, y^{(i)})^{*} \tag{25}$$

TABLE III: Hyper-parameters found during the validation stage.

Batch size	Epochs	Hidden size	Sequence length	Learning rate	Dropout probability
7	10	192	144	10^{-3}	0.25

L is conveniently expressed as a conjugate product instead of the extended MSE form to ease the burden of the complex automatic differentiation algorithm. According to the Wirtinger calculus theory for complex optimization, ${}^{\partial L}/_{\partial z}$ and ${}^{\partial L}/_{\partial z^{*}}$ must be computed assuming $z = \mathrm{err}(\hat{y}, y)$ [39][3].

This objective function does not account for the relative magnitudes of active and reactive forecast errors. As a result, the SGD algorithm will prioritize minimizing the sum of these errors. However, this approach has two limitations. Firstly, the learning process and solutions discovered will be influenced by the magnitude of the input data, even after applying normalization procedures to the individual series. Secondly, the function does not assign penalty or cost values for forecast inaccuracies in the active load series or the reactive load series respectively.

D. Validation process

The model was instantiated using the PyTorch library [40] for the objective of generating point-wise complex forecasts. As the paper objective states, the goal is to achieve a CVNN architecture that can provide a forecast that outperforms a real-valued equivalent architecture with a similar capacity per task. That does not equate to finding the best possible forecast that can be achieved. So, through several iterative trials and by looking at learning curves, a suitable set of hyper-parameters that produce acceptable forecasts was found and are summarized in Tab. III. By studying the learning curves, these parameters result in a balanced bias-variance trade-off with validation data.

Hidden size in Tab. III references to the hidden size h of the input layer, as the hidden sizes of the upper layers are half of the previous one. Additionally, the sequence length parameter refers to the dimensionality of the input vector, denoted as n. Essentially, it represents the number of preceding values provided to the network in order to generate a single-point forecast. In this case, $n = 144$, means that the network utilizes the preceding 36-hour history to forecast the load for the next 15 minutes.

III. Results and Discussion

Following the validation of the model, the validation data is merged with the training set, and the model is retrained for a fixed number of ten epochs to evaluate its performance on the test data. The purpose of combining the training and validation sets is to ensure that the model remains as up-to-date as possible, thus preventing the loss of auto-correlation between the training and test data.

Furthermore, an independent forecast is conducted for the time series of active and reactive loads using a real-valued analog model for each load and semester. For this baseline

[3] Valuable intuition is contained in the following PyTorch documentation.

analogous model formation, the five CRL layers are replaced with the standard real-valued LSTM layers and are instantiated with the same hyper-parameters, such as hidden dimensions, sequence length, input and output sizes shown in Fig. 2a. The in-between layers' modReLU (21) activations are replaced with classical ReLUs (20). The linear or fully connected layers remain the same but become affine transformations of real matrix spaces instead of complex ones. The dropout layer does not undergo any changes.

The instantiation of the CRL model with the aforementioned hyper-parameters results in 1,195,297 trainable parameters, while the baseline model instantiation consists of 570,241 trainable parameters. This observation is logical as the baseline model, which performs half of the task, roughly has half the capacity of the CRL model[4].

Forecast errors are measured using the following metrics for the entirety of the testing set: root mean squared error (RMSE) expressed in Eq. (26), the bias exhibited in Eq. (27), and the mean absolute percentage error (MAPE) manifested in Eq. (28).

$$\mathrm{RMSE} = \sqrt{\frac{1}{\mathrm{card}(y_{\mathrm{test}})} \sum_{i=0}^{\mathrm{card}(y_{\mathrm{test}})-1} (y^{(i)} - \hat{y}^{(i)})^2} \tag{26}$$

$$\mathrm{bias}(\hat{y}) = \frac{1}{\mathrm{card}(y_{\mathrm{test}})} \sum_{i=0}^{\mathrm{card}(y_{\mathrm{test}})-1} \hat{y}^{(i)} \tag{27}$$

$$\mathrm{MAPE} = \frac{100\%}{\mathrm{card}(y_{\mathrm{test}})} \sum_{i=0}^{\mathrm{card}(y_{\mathrm{test}})-1} \frac{|y^{(i)} - \hat{y}^{(i)}|}{y^{(i)}} \tag{28}$$

Eqs. (26, 27, 28) represent real-valued metrics and are either applied to the real components or the imaginary components of the target and forecast vectors. These metrics should be interpreted collectively as they shed light on different aspects of forecast error. The RMSE measures the errors in the unit of the data (kW, kVAr, or kVA), giving a sense of magnitude. The bias indicates the offset or systematic error in the means of the forecast, and the MAPE metric speaks in percentage terms, decoupling from magnitudes and facilitating comparisons with other loads. However, it should be noted that the MAPE metric can be distorted when the target data is close to zero.

The results of the proposed CRL model are presented in Tab. IV, while the results of the baseline model can be found in Table V. Additionally, Figs. 4a, 4b, 4c, and 4d illustrate the different forecast models employed. Each figure presents the target time series for the real-active components of the series in kW in the upper subplot and the imaginary-reactive part of the series in kVAr in the lower subplot.

TABLE IV: Results of complex forecasting using the proposed CRL model.

Load	Term	RMSE$_p$ [kW]	Bias$_p$ [kW]	MAPE$_p$ [%]	RMSE$_q$ [kVAr]	Bias$_q$ [kVAr]	MAPE$_q$ [%]
G0-A	I	46.72	7.84	8.05	44.59	12.63	6.93
	II	35.79	8.65	7.28	30.83	-6.47	7.12
G0-M	I	36.48	15.20	6.40	64.34	15.47	13.84
	II	25.42	-4.32	4.76	31.83	-3.48	15.62
G3-A	I	62.11	-4.10	10.15	125.01	2.06	19.13
	II	77.67	24.19	10.84	147.03	31.69	22.88
G3-M	I	38.06	4.93	5.30	52.11	1.90	14.94
	II	41.56	11.32	4.42	61.23	10.62	13.76
L0-A	I	54.96	-0.97	8.83	84.29	28.28	7.92
	II	43.84	14.83	8.28	62.43	-6.31	8.78
L2-M	I	47.95	15.77	7.50	67.44	10.83	9.51
	II	27.34	-11.12	12.06	25.27	-5.29	11.36
Rural-1	I	38.66	-6.40	9.94	26.30	-2.92	6.63
	II	41.89	4.96	10.31	23.50	-5.10	7.65
Rural-2	I	16.23	-0.01	9.25	22.77	4.36	17.18
	II	25.39	5.51	9.52	37.06	0.41	27.23
Rural-3	I	17.67	-4.89	16.13	22.71	-3.34	53.29
	II	33.00	-1.12	13.62	50.26	1.96	73.21
Semiurb-4	I	17.01	-4.49	7.90	18.49	3.67	12.72
	II	25.96	9.26	8.21	25.68	-6.98	26.13
Total average	**All**	**37.69**	**4.25**	**8.94**	**51.16**	**4.20**	**18.79**

TABLE V: Results of individual forecasting using a real-valued equivalent LSTM model.

Load	Term	RMSE$_p$ [kW]	Bias$_p$ [kW]	MAPE$_p$ [%]	RMSE$_q$ [kVAr]	Bias$_q$ [kVAr]	MAPE$_q$ [%]
G0-A	I	62.15	3.15	13.20	155.60	126.05	24.86
	II	39.57	-2.07	8.98	106.17	-95.66	29.53
G0-M	I	41.62	12.47	6.28	280.74	72.62	96.78
	II	29.48	-14.73	7.00	38.06	-20.32	22.20
G3-A	I	64.14	-3.82	10.58	199.69	69.38	29.23
	II	134.41	16.67	21.78	242.35	17.51	48.72
G3-M	I	39.29	13.39	5.20	52.69	-2.56	15.53
	II	42.20	10.90	4.48	125.38	13.62	43.42
L0-A	I	57.76	13.19	9.22	127.22	79.90	11.33
	II	48.06	7.11	8.89	261.34	-86.70	62.38
L2-M	I	49.06	4.46	7.76	82.14	-14.13	11.92
	II	25.58	-6.35	10.71	27.72	9.39	11.64
Rural-1	I	39.70	-4.52	10.55	27.39	4.20	6.79
	II	42.73	0.89	11.86	24.86	3.82	7.75
Rural-2	I	18.75	-4.71	10.53	44.47	-2.30	47.28
	II	26.15	0.22	10.19	38.33	4.90	25.54
Rural-3	I	17.75	-3.11	16.19	23.25	-4.61	58.34
	II	32.90	4.14	13.23	51.80	0.06	79.14
Semiurb-4	I	18.55	1.94	8.20	18.48	2.04	12.83
	II	23.41	6.79	7.30	25.37	-2.29	22.40
Total average	**All**	**42.66**	**2.80**	**10.11**	**97.65**	**8.75**	**33.38**

A series of paired sample statistical tests will be conducted to compare the RMSE results achieved with the CRL model, as shown in Table IV, with the RMSE results obtained from the baseline real-valued equivalent model, as seen in Table V. In this comparison, the paired sign test will be employed, which belongs to a family of non-parametric tests and works with little assumptions of underlying data. The only conditions required for the validity of this test are that the paired differences must be continuous and independent from each other [41], [42]. These conditions are satisfied in our case, only if the results obtained per semester are treated as a different population i.e. to analyze paired differences of samples for semester I and then for semester II. The main limitation of this test is, as the name suggests, that does not factor in paired differences magnitudes but only signs.

It is important to note that the paired sign test will be applied six times: twice for the paired active differences (once for each semester), twice for the paired reactive differences (once for each semester), and twice for the Pythagorean addition of active and reactive RMSEs[5] resulting in apparent power errors, as defined in Eq. (29), for the respective paired differences as well (once for each semester).

$$\mathrm{RMSE}_s = \sqrt{\mathrm{RMSE}_p^2 + \mathrm{RMSE}_q^2} \tag{29}$$

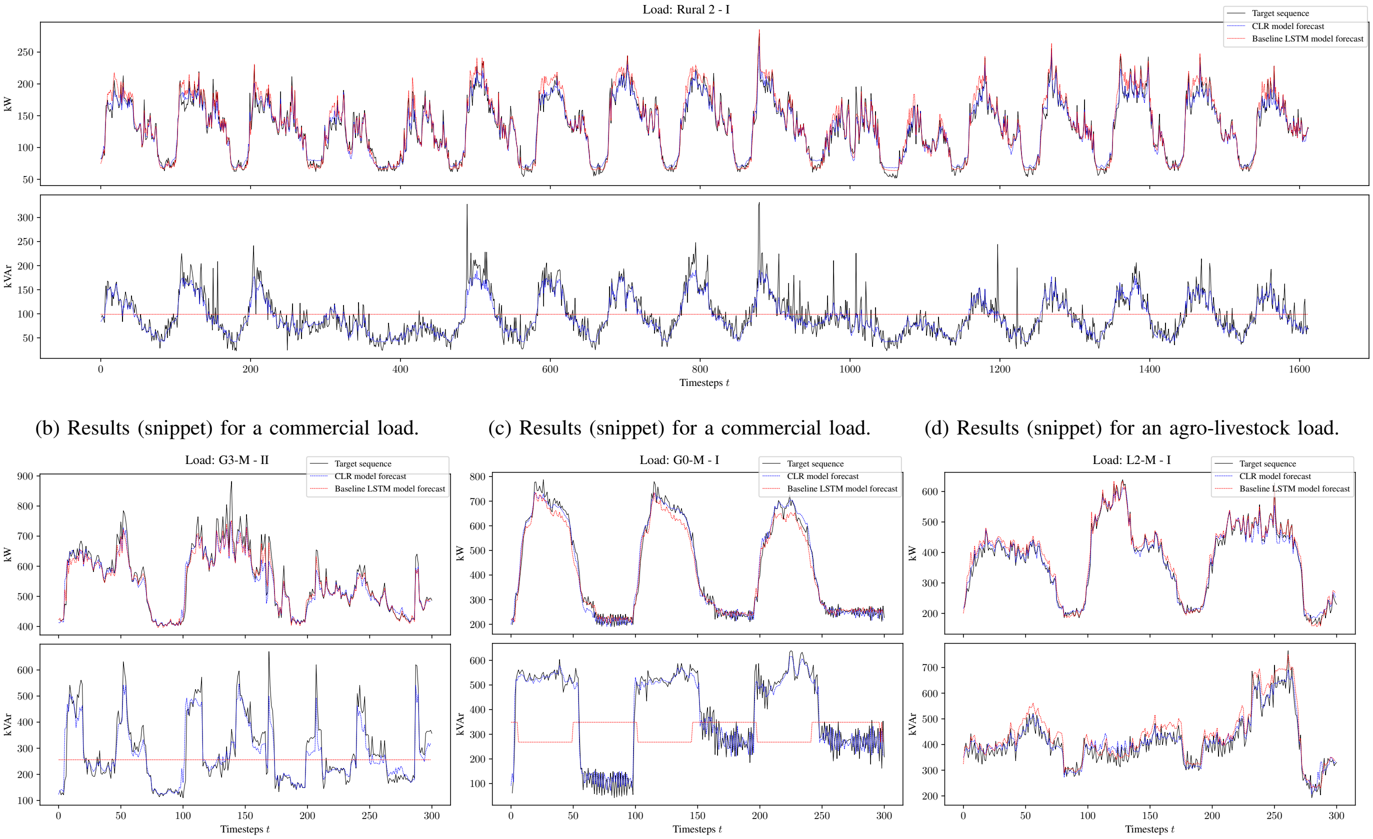
The null hypothesis (H_0) for this test states that the median of the paired differences is zero, while the alternative hypothesis (H_a) states that the median is not zero [41]. If the results

[4] The parameter ratio of an ideal 2:1 between models is breached since there is not an exact real equivalent of the modReLU activation.

[5] Not to confuse the plural form of RMSE with the apparent power RMSE denoted as RMSE_s defined in Eq. (29).

Fig. 4: Visualizations of some results over slices of the test sets for different load categories.

(a) Results over an entire test set for a residential rural load.



(b) Results (snippet) for a commercial load.

(c) Results (snippet) for a commercial load.

(d) Results (snippet) for an agro-livestock load.

TABLE VI: Medians and p-values obtained from the sign test of paired differences (Δ).

Semester	Descriptor	ΔRMSE$_p$	ΔRMSE$_q$	ΔRMSE$_s$
I	Median	-1.79	-18.20	-16.61
	p-value	0.20%	2.15%	0.20%
II	Median	-0.80	-4.34	-4.44
	p-value	34.44%	2.15%	2.15%

of both models are similar, H$_0$ cannot be rejected in favor of H$_a$. However, if H$_0$ is rejected in favor of H$_a$, it provides evidence that the proposed CRL model behaves in a different way than the baseline model depending on the sign of the median statistic. The paired differences were set up as CRL RMSEs minus the baseline RMSEs. Consequently, a negative median suggests the favorable performance of the CRL model.

The obtained two-tailed p-values as well as the median statistic from the tests are gathered in Tab. VI. Five out of six tests reach the same conclusion: strongly reject H$_0$ in favor of H$_a$ with a significance far below the 5% threshold. The remaining result fails to reject H$_0$ in the case of the second semester's active forecast results for the given 5% threshold. All things considered, from a birds-eye point of view, the CRL model significantly outperforms the baseline model, especially for the accuracy of reactive forecasts.

Overall, the obtained results can be explained by the ability of the CRL network to recognize patterns in both active and reactive profiles and leverage their correlations to generate more accurate complex or bi-variate forecasts in both dimensions compared to the uni-variate non-complex equivalent. This synergistic behavior is attributed to the working principles of CVNNs, which can be summed up as productively extracting mutual correlation patterns between real and imaginary components of the input and productively being better function approximators for some problems as [18]–[21], [27] research points out.

The benefits reaped from utilizing the CRL network, in terms of its two tasks, are asymmetrical: the improvement in active load forecast accuracy is relatively modest compared to the significant gains achieved in accuracy for reactive load forecasts. This discrepancy can be primarily attributed to the presence of stronger noise in reactive profiles and considering the already discussed limitations of the loss function formulation. By employing basic statistical tools such as auto-correlation and partial auto-correlation plots, it becomes apparent that the reactive load profiles exhibit a lower signal-to-noise ratio. Consequently, the spectral energy density of the target signal is more evenly distributed across its bandwidth, in contrast to its active load counterpart. As a result, effectively capturing low-frequency dynamics (where usually the bulk of the series relevant information lies) coupled with intricate high-frequency dynamics in a uni-variate forecast demands a higher model capacity that the baseline LSTM model fails to provide.

Several results indicate that the baseline model under-fits the reactive load data due to the aforementioned reason. This situation is captured by the second subplot of Fig. 4a, where the reactive load forecast closely approximates the mean of the training set. Hence, when processing both the active and reactive parts of the input, which are correlated and considered prior knowledge (refer to Tab. I), the CRL sub-networks can effectively extrapolate the noise and signal dynamics from the simpler sequence (i.e., the active profile in this case) and infer some dynamics of the difficult sequence (i.e., the reactive profile).

IV. Conclusions

In this paper, a new recurrent complex-valued LSTM-based load ultra short-term forecasting model was proposed that can perform multi-task forecasts of active and reactive power consumption at a distribution level 15 minutes ahead. This model was trained, validated, and tested using 10 SimBench rural profiles benchmarks from Germany that contain year-round active and reactive consumption data for multiple categories of loads with 15-minute resolution. Suitable hyper-parameters were found in the validation stage.

In parallel, an analogous real-valued baseline model that performs single-task forecasts was instantiated with the same hyper-parameters to for competition. The reason for this was to compare the incremental benefits of using a complex-valued architecture instead of a real-valued one. The preferred metric to measure forecast error results and subsequently perform statistical inference on was the RMSE. All results measured from different experiments can be found on Tab. IV and Tab. V.

The results retrieved by both models were compared by a series of paired sample hypothesis sign tests that provided strong statistical evidence that the CRL model performed better in at least one of the tasks for each semester. The maximum p-value obtained when considering the apparent power RMSE metric was 2.15%. The most notorious improvements were in the accuracy of reactive forecasts. This in turn supports the hypothesis stated in the introduction section: CVNNs are effective at learning and representing complex load temporal patterns in a synergistic manner. This conclusion hints that the complex domain could be a promising framework for performing time-series-based multi-task load forecasting. Of course, more research is needed.

Some follow-up research topics are suggested in the following listing:

1) Formalize the validation process using systematic approaches, such as grid search of hyper-parameters and cross-validation techniques, as demonstrated in [32].
2) Investigate the formulation of a loss function that incorporates error costs for over and underestimation in active and reactive forecasts respectively.
3) Enhance the model by incorporating the latest techniques in sequence learning, such as attention mechanisms described in [43], specifically tailored to time series analysis.
4) Modify the complex-valued architecture to support real-valued exogenous inputs, expanding its forecasting capabilities.
5) Explore the potential of fully complex recurrent networks proposed in [44] as an alternative to the "Realisation of Complex LSTM (RC-LSTM)" approach [35].
6) Conduct real-world deployments of the model to evaluate its performance and consider the influence of physical and computational limitations.
7) Investigate the model's behavior across different timescales and node aggregation levels, testing all profiles available in the SimBench benchmark.
8) Explore the extension of the model to multi-dimensional complex load forecasting, enabling forecasts for multiple loads and their active/reactive components simultaneously.
9) Extend the application of the model beyond load forecasting, such as incorporating black-box or grey-box load modeling.

Acknowledgments

The hypothesis testing for this paper was generated using the Real Statistics Resource Pack software (Release 7.6) for Microsoft Excel. Copyright (2013 – 2023) Charles Zaiontz. www.real-statistics.com. I also acknowledge the use of ChatGPT large language model as a grammar and spell checker, but it was never used as a source of text generation for this article.

References

[1] A. A. Mamun, M. Sohel, N. Mohammad, M. S. Haque Sunny, D. R. Dipta, and E. Hossain, "A comprehensive review of the load forecasting techniques using single and hybrid predictive models," *IEEE Access*, vol. 8, pp. 134911–134939, 2020.

[2] I. K. Nti, M. Teimeh, O. Nyarko-Boateng, and A. F. Adekoya, "Electricity load forecasting: a systematic review," *Journal of Electrical Systems and Information Technology*, vol. 7, p. 13, Sep 2020.

[3] M. Tan, S. Yuan, S. Li, Y. Su, H. Li, and F. He, "Ultra-short-term industrial power demand forecasting using lstm based hybrid ensemble learning," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 2937–2948, 2020.

[4] M. Q. Raza and A. Khosravi, "A review on artificial intelligence based load demand forecasting techniques for smart grid and buildings," *Renewable and Sustainable Energy Reviews*, vol. 50, pp. 1352–1372, 2015.

[5] A. Bracale, P. Caramia, P. De Falco, and T. Hong, "A multivariate approach to probabilistic industrial load forecasting," *Electric Power Systems Research*, vol. 187, p. 106430, 2020.

[6] S. Pirouzi, J. Aghaei, V. Vahidinasab, T. Niknam, and A. Khodaei, "Robust linear architecture for active/reactive power scheduling of ev integrated smart distribution networks," *Electric Power Systems Research*, vol. 155, pp. 8–20, 2018.

[7] M. Ghaljehei, Z. Soltani, J. Lin, G. Gharehpetian, and M. Golkar, "Stochastic multi-objective optimal energy and reactive power dispatch considering cost, loading margin and coordinated reactive power reserve management," *Electric Power Systems Research*, vol. 166, pp. 163–177, 2019.

[8] N. S. Coleman and K. N. Miu, "Distribution load capability with nodal power factor constraints," *IEEE Transactions on Power Systems*, vol. 32, no. 4, p. 3120 – 3126, 2017. Cited by: 6; All Open Access, Bronze Open Access.

[9] R. H. Zubo, G. Mokryani, and R. Abd-Alhameed, "Optimal operation of distribution networks with high penetration of wind and solar power within a joint active and reactive distribution market environment," *Applied Energy*, vol. 220, pp. 713–722, 2018.

[10] H. Nezamabadi and M. Setayesh Nazar, "Arbitrage strategy of virtual power plants in energy, spinning reserve and reactive power markets," *IET Generation, Transmission & Distribution*, vol. 10, no. 3, pp. 750–763, 2016.

[11] A. Samimi, M. Nikzad, and P. Siano, "Scenario-based stochastic framework for coupled active and reactive power market in smart distribution systems with demand response programs," *Renewable Energy*, vol. 109, pp. 22–40, 2017.

[12] A. Kargarian, M. Raoofat, and M. Mohammadi, "Probabilistic reactive power procurement in hybrid electricity markets with uncertain loads," *Electric Power Systems Research*, vol. 82, no. 1, pp. 68–80, 2012.

[13] A. Potter, R. Haider, G. Ferro, M. Robba, and A. M. Annaswamy, "A reactive power market for the future grid," *Advances in Applied Energy*, vol. 9, p. 100114, 2023.

[14] J. Zhong and K. Bhattacharya, "Reactive power market design and its impact on market power," in *2008 IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century*, pp. 1–4, 2008.

[15] Y. Xu and J. V. Milanović, "Accuracy of ann based methodology for load composition forecasting at bulk supply buses," in *2014 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pp. 1–6, 2014.

[16] X. Han, "Ultra-short-term multi-node load forecasting – a composite approach," *IET Generation, Transmission & Distribution*, vol. 6, pp. 436–444(8), May 2012.

[17] J.-M. Cho, J.-H. Kim, W.-H. Park, Y. ho Lee, and J.-O. Kim, "Short-term reactive power load forecasting using multiple time-series model," *IFAC Proceedings Volumes*, vol. 36, no. 20, pp. 985–990, 2003. 5th IFAC Symposium on Power Plants and Power Systems Control 2003, Seoul, South Korea, 15-19 September 2003.

[18] A. Hirose, *Complex-Valued Neural Networks: Distinctive Features*, pp. 17–56. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[19] J. Bassey, L. Qian, and X. Li, "A survey of complex-valued neural networks," 2021.

[20] C. Lee, H. Hasegawa, and S. Gao, "Complex-valued neural networks: A comprehensive survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 8, pp. 1406–1426, 2022.

[21] T. Nitta, "Orthogonality of decision boundaries in complex-valued neural networks," *Neural Computation*, vol. 16, no. 1, pp. 73–97, 2004.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[23] X. Yao, X. Shi, and F. Zhou, "Complex-value convolutional neural network for classification of human activities," in *2019 6th Asia-Pacific Conference on Synthetic Aperture Radar (APSAR)*, pp. 1–6, 2019.

[24] C. Trabelsi, O. Bilaniuk, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, "Deep complex networks," *CoRR*, vol. abs/1705.09792, 2017.

[25] A. Marseet and F. Sahin, "Application of complex-valued convolutional neural network for next generation wireless networks," in *2017 IEEE Western New York Image and Signal Processing Workshop (WNYISPW)*, pp. 1–5, 2017.

[26] H. H. Çevik, Y. E. Acar, and M. Çunkaş, "Day ahead wind power forecasting using complex valued neural network," in *2018 International Conference on Smart Energy Systems and Technologies (SEST)*, pp. 1–6, 2018.

[27] A. Hirose, "Continuous complex-valued back-propagation learning," *Electronics Letters*, vol. 28, no. 20, pp. 1854 – 1855, 1992.

[28] G. M. Jónsdóttir and F. Milano, "Modeling correlation of active and reactive power of loads for short-term analysis of power systems," in *2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe)*, pp. 1–6, 2020.

[29] K. Berk, A. Hoffmann, and A. Müller, "Probabilistic forecasting of industrial electricity load with regime switching behavior," *International Journal of Forecasting*, vol. 34, no. 2, pp. 147–162, 2018.

[30] S. Meinecke, D. Sarajlić, S. R. Drauz, A. Klettke, L.-P. Lauven, C. Rehtanz, A. Moser, and M. Braun, "Simbench—a benchmark dataset of electric power systems to compare innovative solutions based on power flow analysis," *Energies*, vol. 13, no. 12, 2020.

[31] C. Spalthoff, D. Sarajlic, C. Kittl, S. Drauz, T. Kneiske, C. Rehtanz, and M. Braun, "Simbench: Open source time series of power load, storage and generation for the simulation of electrical distribution grids," in *International ETG-Congress 2019; ETG Symposium*, pp. 1–6, 2019.

[32] J. Qin, Y. Zhang, S. Fan, X. Hu, Y. Huang, Z. Lu, and Y. Liu, "Multitask short-term reactive and active load forecasting method based on attention-lstm model," *International Journal of Electrical Power & Energy Systems*, vol. 135, p. 107517, 2022.

[33] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, pp. 1735–1780, 11 1997.

[34] D. Chicco, *Siamese Neural Networks: An Overview*, pp. 73–94. New York, NY: Springer US, 2021.

[35] R. G. Goswami, S. Andhavarapu, and K. S. R. Murty, "Phase aware speech enhancement using realisation of complex-valued lstm," 2020.

[36] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, vol. abs/1207.0580, 2012.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.

[38] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, ch. 5. MIT Press, 2016. http://www.deeplearningbook.org.

[39] K. Kreutz-Delgado, "The complex gradient operator and the cr-calculus," 2009.

[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," 2019.

[41] K. M. Ramachandran and C. P. Tsokos, "Chapter 12 - non-parametric statistics," in *Mathematical Statistics with Applications in R (Third Edition)* (K. M. Ramachandran and C. P. Tsokos, eds.), pp. 491–530, Academic Press, third edition ed., 2021.

[42] E. L. Lehmann and J. P. Romano, *Testing statistical hypotheses*. Springer texts in statistics, Cham, Switzerland: Springer Nature, 4 ed., June 2022.

[43] S. Shih, F. Sun, and H. Lee, "Temporal pattern attention for multivariate time series forecasting," *CoRR*, vol. abs/1809.04206, 2018.

[44] M. Wolter and A. Yao, "Complex gated recurrent neural networks," in *Advances in Neural Information Processing Systems* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), vol. 31, Curran Associates, Inc., 2018.

Acronyms

ADAM	Adaptive Moment Estimation
SGD	Stochastic Gradient Descent
ANN	Artificial Neural Network
CRL	Complex Recurrent Layer
CVNN	Complex Valued Neural Network
LSTM	Long Short Term Memory
MAPE	Mean Absolute Percentage Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SVR	Support Vector Regression

[redacted]