Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks

Guokun Lai
Carnegie Mellon University
guokun@cs.cmu.edu

Wei-Cheng Chang
Carnegie Mellon University
wchang2@andrew.cmu.edu

Yiming Yang
Carnegie Mellon University
yiming@cs.cmu.edu

Hanxiao Liu
Carnegie Mellon University
hanxiaol@cs.cmu.edu

ABSTRACT

Multivariate time series forecasting is an important machine learning problem across many domains, including predictions of solar plant energy output, electricity consumption, and traffic jam situation. Temporal data arise in these real-world applications often involves a mixture of long-term and short-term patterns, for which traditional approaches such as Autoregressive models and Gaussian Process may fail. In this paper, we proposed a novel deep learning framework, namely Long- and Short-term Time-series network (LSTNet), to address this open challenge. LSTNet uses the Convolution Neural Network (CNN) and the Recurrent Neural Network (RNN) to extract short-term local dependency patterns among variables and to discover long-term patterns for time series trends. Furthermore, we leverage traditional autoregressive model to tackle the scale insensitive problem of the neural network model. In our evaluation on real-world data with complex mixtures of repetitive patterns, LSTNet achieved significant performance improvements over that of several state-of-the-art baseline methods. All the data and experiment codes are available online.

KEYWORDS

Multivariate Time Series, Neural Network, Autoregressive models

ACM Reference Format:
Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. In *Proceedings of ACM Conference (SIGIR'18).* ACM, New York, NY, USA, 11 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

Multivariate time series data are ubiquitous in our everyday life ranging from the prices in stock markets, the traffic flows on highways, the outputs of solar power plants, the temperatures across different cities, just to name a few. In such applications, users are often interested in the forecasting of the new trends or potential hazardous events based on historical observations on time series signals. For instance, a better route plan could be devised based on the predicted traffic jam patterns a few hours ahead, and a larger profit could be made with the forecasting of the near-future stock market.

Multivariate time series forecasting often faces a major research challenge, that is, how to capture and leverage the dynamics dependencies among multiple variables. Specifically, real-world applications often entail a mixture of short-term and long-term repeating patterns, as shown in Figure 1 which plots the hourly occupancy rate of a freeway. Apparently, there are two repeating patterns, daily and weekly. The former portraits the morning peaks vs. evening peaks, while the latter reflects the workday and weekend patterns. A successful time series forecasting model should be capture both kinds of recurring patterns for accurate predictions. As another example, consider the task of predicting the output of a solar energy farm based on the measured solar radiation by massive sensors over different locations. The long-term patterns reflect the difference between days vs. nights, summer vs. winter, etc., and the short-term patterns reflect the effects of cloud movements, wind direction changes, etc. Again, without taking both kinds of recurrent patterns into account, accurate time series forecasting is not possible. However, traditional approaches such as the large body of work in autoregressive methods [2, 12, 22, 32, 35] fall short in this aspect, as most of them do not distinguish the two kinds of patterns nor model their interactions explicitly and dynamically. Addressing such limitations of existing methods in time series forecasting is the main focus of this paper, for which we propose a novel framework that takes advantages of recent developments in deep learning research.

Deep neural networks have been intensively studied in related domains, and made extraordinary impacts on the solutions of a broad range of problems. The recurrent neural networks (RNN) models [9], for example, have become most popular in recent natural language processing (NLP) research. Two variants of RNN in particular, namely the Long Short Term Memory (LSTM) [15] and the Gated Recurrent Unit (GRU) [6], have significantly improved the state-of-the-art performance in machine translation, speech recognition and other NLP tasks as they can effectively capture the meanings of words based on the long-term and short-term dependencies among them in input documents [1, 14, 19].In the field of computer vision, as another example, convolution neural network (CNN) models [19, 21] have shown outstanding performance by successfully extracting local and shift-invariant features (called "shapelets" sometimes) at various granularity levels from input images.

SIGIR'18, July 2018, Ann Arbor, MI, USA

ACM ISBN 123-4567-24-567/08/06...$15.00
https://doi.org/10.475/123_4

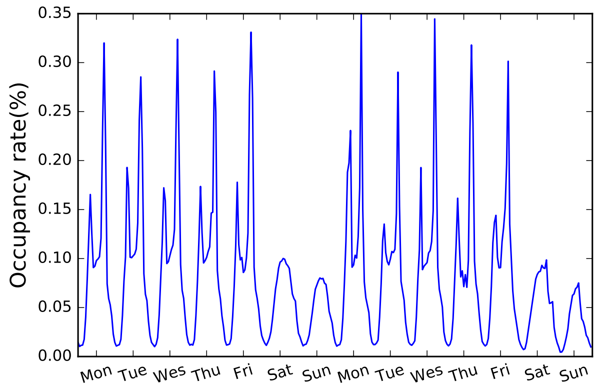


Figure 1: The hourly occupancy rate of a road in the bay area for 2 weeks

Deep neural networks have also received an increasing amount of attention in time series analysis. A substantial portion of the previous work has been focusing on *time series classification*, i.e., the task of automated assignment of class labels to time series input. For instance, RNN architectures have been studied for extracting informative patterns from health-care sequential data [5, 23] and classifying the data with respect diagnostic categories. RNN has also been applied to mobile data, for classifying the input sequences with respect to actions or activities [13]. CNN models have also been used in action/activity recognition [13, 20, 31], for the extraction of shift-invariant local patterns from input sequences as the features of classification models.

Deep neural networks have also been studied for *time series forecasting* [8, 33], i.e., the task of using observed time series in the past to predict the unknown time series in a look-ahead horizon – the larger the horizon, the harder the problem. Efforts in this direction range from the early work using naive RNN models [7] and the hybrid models [16, 34, 35] combining the use of ARIMA [3] and Multilayer Perceptron (MLP), to the recent combination of vanilla RNN and Dynamic Boltzmann Machines in time series forecasting [8].

In this paper, we propose a deep learning framework designed for the multivariate time series forecasting, namely Long- and Short-term Time-series Network (LSTNet), as illustrated in Figure 2. It leverages the strengths of both the convolutional layer to discover the local dependency patterns among multi-dimensional input variables and the recurrent layer to captures complex long-term dependencies. A novel recurrent structure, namely Recurrent-skip, is designed for capturing very long-term dependence patterns and making the optimization easier as it utilizes the periodic property of the input time series signals. Finally, the LSTNet incorporates a traditional autoregressive linear model in parallel to the non-linear neural network part, which makes the non-linear deep learning model more robust for the time series with violate scale changing. In the experiment on the real world seasonal time series datasets, our model consistently outperforms the traditional linear models and GRU recurrent neural network.

The rest of this paper is organized as follows. Section 2 outlines the related background, including representative auto-regressive methods and Gaussian Process models. Section 3 describe our proposed LSTNet. Section 4 reports the evaluation results of our model in comparison with strong baselines on real-world datasets. Finally, we conclude our findings in Section 5.

2 RELATED BACKGROUND

One of the most prominent univariate time series models is the autoregressive integrated moving average (ARIMA) model. The popularity of the ARIMA model is due to its statistical properties as well as the well-known Box-Jenkins methodology [2] in the model selection procedure. ARIMA models are not only adaptive to various exponential smoothing techniques [25] but also flexible enough to subsume other types of time series models including autoregression (AR), moving average (MA) and Autoregressive Moving Average (ARMA). However, ARIMA models, including their variants for modeling long-term temporal dependencies [2], are rarely used in high dimensional multivariate time series forecasting due to their high computational cost.

On the other hand, vector autoregression (VAR) is arguably the most widely used models in multivariate time series [2, 12, 24] due to its simplicity. VAR models naturally extend AR models to the multivariate setting, which ignores the dependencies between output variables. Significant progress has been made in recent years in a variety of VAR models, including the elliptical VAR model [27] for heavy-tail time series and structured VAR model [26] for better interpretations of the dependencies between high dimensional variables, and more. Nevertheless, the model capacity of VAR grows linearly over the temporal window size and quadratically over the number of variables. This implies, when dealing with long-term temporal patterns, the inherited large model is prone to overfitting. To alleviate this issue, [32] proposed to reduce the original high dimensional signals into lower dimensional hidden representations, then applied VAR for forecasting with a variety choice of regularization.

Time series forecasting problems can also be treated as standard regression problems with time-varying parameters. It is therefore not surprising that various regression models with different loss functions and regularization terms are applied to time series forecasting tasks. For example, linear support vector regression (SVR) [4, 17] learns a max margin hyperplane based on the regression loss with a hyper-parameter ϵ controlling the threshold of prediction errors. Ridge regression is yet another example which can be recovered from SVR models by setting ϵ to zeros. Lastly, [22] applied LASSO models to encourage sparsity in the model parameters so that interesting patterns among different input signals could be manifest. These linear methods are practically more efficient for multivariate time series forecasting due to high-quality off-the-shelf solvers in the machine learning community. Nonetheless, like VARs, those linear models may fail to capture complex non-linear relationships of multivariate signals, resulting in an inferior performance at the cost of its efficiency.

Gaussian Processes (GP) is a non-parametric method for modeling distributions over a continuous domain of functions. This contrasts with models defined by a parameterized class of functions

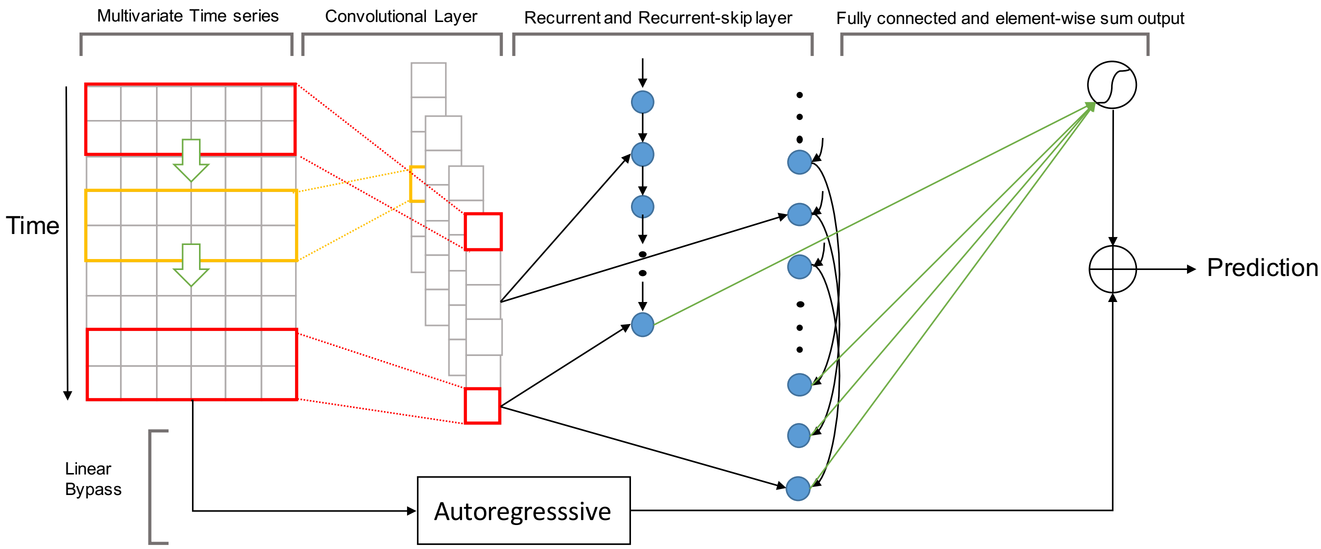


Figure 2: An overview of the Long- and Short-term Time-series network (LSTNet)

such as VARs and SVRs. GP can be applied to multivariate time series forecasting task as suggested in [28], and can be used as a prior over the function space in Bayesian inference. For example, [10] presented a fully Bayesian approach with the GP prior for nonlinear state-space models, which is capable of capturing complex dynamical phenomena. However, the power of Gaussian Process comes with the price of high computation complexity. A straightforward implementation of Gaussian Process for multivariate time-series forecasting has cubic complexity over the number of observations, due to the matrix inversion of the kernel matrix.

3 FRAMEWORK

In this section, we first formulate the time series forecasting problem, and then discuss the details of the proposed LSTNet architecture (Figure 2) in the following part. Finally, we introduce the objective function and the optimization strategy.

3.1 Problem Formulation

In this paper, we are interested in the task of multivariate time series forecasting. More formally, given a series of fully observed time series signals $Y = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T\}$ where $\boldsymbol{y}_t \in \mathbb{R}^n$, and n is the variable dimension, we aim at predicting a series of future signals in a rolling forecasting fashion. That being said, to predict $\boldsymbol{y}_{T+h}$ where h is the desirable horizon ahead of the current time stamp, we assume $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T\}$ are available. Likewise, to predict the value of the next time stamp $\boldsymbol{y}_{T+h+1}$, we assume $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T, \boldsymbol{y}_{T+1}\}$ are available. We hence formulate the input matrix at time stamp T as $X_T = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T\} \in \mathbb{R}^{n \times T}$.

In the most of cases, the horizon of the forecasting task is chosen according to the demands of the environmental settings, e.g. for the traffic usage, the horizon of interest ranges from hours to a day; for the stock market data, even seconds/minutes-ahead forecast can be meaningful for generating returns.

Figure 2 presents an overview of the proposed LSTnet architecture. The LSTNet is a deep learning framework specifically designed for multivariate time series forecasting tasks with a mixture of long- and short-term patterns. In following sections, we introduce the building blocks for the LSTNet in detail.

3.2 Convolutional Component

The first layer of LSTNet is a convolutional network without pooling, which aims to extract short-term patterns in the time dimension as well as local dependencies between variables. The convolutional layer consists of multiple filters of width ω and height n (the height is set to be the same as the number of variables). The k-th filter sweeps through the input matrix X and produces

$$h_k = RELU(W_k * X + b_k) \tag{1}$$

where $*$ denotes the convolution operation and the output h_k would be a vector, and the $RELU$ function is $RELU(x) = \max(0, x)$. We make each vector h_k of length T by zero-padding on the left of input matrix X. The output matrix of the convolutional layer is of size $d_c \times T$ where d_c denotes the number of filters.

3.3 Recurrent Component

The output of the convolutional layer is simultaneously fed into the Recurrent component and Recurrent-skip component (to be described in subsection 3.4). The Recurrent component is a recurrent layer with the Gated Recurrent Unit (GRU) [6] and uses the $RELU$ functionentrrentfunction filters.

where $\odot$ is the element-wise product, σ is the sigmoid function and x_t is the input of this layer at time t. The output of this layer is the hidden state at each time stamp. While researchers are accustomed to using tanh function as hidden update activation function, we empirically found $RELU$ leads to more reliable performance, through which the gradient is easier to back propagate.

3.4 Recurrent-skip Component

The Recurrent layers with GRU [6] and LSTM [15] unit are carefully designed to memorize the historical information and hence to be aware of relatively long-term dependencies. Due to gradient vanishing, however, GRU and LSTM usually fail to capture very long-term correlation in practice. We propose to alleviate this issue via a novel recurrent-skip component which leverages the periodic pattern in real-world sets. For instance, both the electricity consumption and traffic usage exhibit clear pattern on a daily basis. If we want to predict the electricity consumption at t o'clock for today, a classical trick in the seasonal forecasting model is to leverage the records at t o'clock in historical days, besides the most recent records. This type of dependencies can hardly be captured by off-the-shelf recurrent units due to the extremely long length of one period (24 hours) and the subsequent optimization issues. Inspired by the effectiveness of this trick, we develop a recurrent structure with temporal skip-connections to extend the temporal span of the information flow and hence to ease the optimization process. Specifically, skip-links are added between the current hidden cell and the hidden cells in the same phase in adjacent periods. The updating process can be formulated as,

$$\begin{aligned} r_t &= \sigma(x_t W_{xr} + h_{t-p} W_{hr} + b_r) \\ u_t &= \sigma(x_t W_{xu} + h_{t-p} W_{hu} + b_u) \\ c_t &= RELU(x_t W_{xc} + r_t \odot (h_{t-p} W_{hc}) + b_c) \\ h_t &= (1 - u_t) \odot h_{t-p} + u_t \odot c_t \end{aligned} \tag{3}$$

where the input of this layer is the output of the convolutional layer, and p is the number of hidden cells skipped through. The value of p can be easily determined for datasets with clear periodic patterns (e.g. $p = 24$ for the hourly electricity consumption and traffic usage datasets), and has to be tuned otherwise. In our experiments, we empirically found that a well-tuned p can considerably boost the model performance even for the latter case. Furthermore, the LSTNet could be easily extended to contain variants of the skip length p.

We use a dense layer to combine the outputs of the Recurrent and Recurrent-skip components. The inputs to the dense layer include the hidden state of Recurrent component at time stamp t, denoted by h_t^R, and p hidden states of Recurrent-skip component from time stamp $t-p+1$ to t denoted by $h_{t-p+1}^S, h_{t-p+2}^S \dots, h_t^S$. The output of the dense layer is computed as,

$$h_t^D = W^R h_t^R + \sum_{i=0}^{p-1} W_i^S h_{t-i}^S + b \tag{4}$$

where h_t^D is the prediction result of the neural network (upper) part in the Fig.2 at time stamp t.

3.5 Temporal Attention Layer

However, the Recurrent-skip layer requires a predefined hyper-parameter p, which is unfavorable in the nonseasonal time series prediction, or whose period length is dynamic over time. To alleviate such issue, we consider an alternative approach, attention mechanism [1], which learns the weighted combination of hidden representations at each window position of the input matrix. Specifically, the attention weights $\boldsymbol{\alpha}_t \in \mathbb{R}^q$ at current time stamp t are calculated as

$$\boldsymbol{\alpha}_t = AttnScore(H_t^R, h_{t-1}^R)$$

where $H_t^R = [h_{t-q}^R, \dots, h_{t-1}^R]$ is a matrix stacking the hidden representation of RNN column-wisely and $AttnScore$ is some similarity functions such as dot product, cosine, or parameterized by a simple multi-layer perceptron.

The final output of temporal attention layer is the concatenation of the weighted context vector $\boldsymbol{c}_t = H_t \boldsymbol{\alpha}_t$ and last window hidden representation h_{t-1}^R, along with a linear projection operation

$$h_t^D = W[\boldsymbol{c}_t; h_{t-1}^R] + b.$$

3.6 Autoregressive Component

Due to the non-linear nature of the Convolutional and Recurrent components, one major drawback of the neural network model is that the scale of outputs is not sensitive to the scale of inputs. Unfortunately, in specific real datasets, the scale of input signals constantly changes in a non-periodic manner, which significantly lowers the forecasting accuracy of the neural network model. A concrete example of this failure is given in Section 4.6. To address this deficiency, similar in spirit to the highway network [29], we decompose the final prediction of LSTNet into a linear part, which primarily focuses on the local scaling issue, plus a non-linear part containing recurring patterns. In the LSTNet architecture, we adopt the classical Autoregressive (AR) model as the linear component. Denote the forecasting result of the AR component as $h_t^L \in \mathbb{R}^n$, and the coefficients of the AR model as $W^{ar} \in \mathbb{R}^{q^{ar}}$ and $b^{ar} \in \mathbb{R}$, where q^{ar} is the size of input window over the input matrix. Note that in our model, all dimensions share the same set of linear parameters. The AR model is formulated as follows,

$$h_{t,i}^L = \sum_{k=0}^{q^{ar}-1} W_k^{ar} \boldsymbol{y}_{t-k,i} + b^{ar} \tag{5}$$

The final prediction of LSTNet is then obtained by by integrating the outputs of the neural network part and the AR component:

$$\hat{Y}_t = h_t^D + h_t^L \tag{6}$$

where $\hat{Y}_t$ denotes the model's final prediction at time stamp t.

3.7 Objective function

The squared error is the default loss function for many forecasting tasks, the corresponding optimization objective is formulated as,

$$\underset{\Theta}{\text{minimize}} \sum_{t \in \Omega_{Train}} ||Y_t - \hat{Y}_{t-h}||_F^2 \tag{7}$$

where Θ denotes the parameter set of our model, Ω_{Train} is the set of time stamps used for training, $|| \cdot ||_F$ is the Frobenius norm, and h is the horizon as mentioned in Section 3.1. The traditional linear regression model with the square loss function is named as Linear Ridge, which is equivalent to the vector autoregressive model with ridge regularization. However, experiments show that the Linear Support Vector Regression (Linear SVR) [30] dominates the Linear Ridge model in certain datasets. The only difference between Linear SVR and Linear Ridge is the objective function. The objective function for Linear SVR is,

$$\begin{aligned} &\underset{\Theta}{\text{minimize}}\ \frac{1}{2}||\Theta||_F^2 + C\sum_{t\in\Omega_{Train}}\sum_{i=0}^{n-1}\xi_{t,i} \\ &\text{subject to } |\hat{Y}_{t-h,i} - Y_{t,i}| \le \xi_{t,i} + \epsilon, t \in \Omega_{Train} \\ &\qquad \xi_{t,i} \ge 0 \end{aligned} \tag{8}$$

where C and ϵ are hyper-parameters. Motivated by the remarkable performance of the Linear SVR model, we incorporate its objective function in the LSTNet model as an alternative of the squared loss. For simplicity, we assume $\epsilon = 0$[1], and the objective function above reduces to absolute loss (L1-loss) function as follows:

$$\underset{\Theta}{\text{minimize}} \sum_{t\in\Omega_{Train}}\sum_{i=0}^{n-1}|Y_{t,i} - \hat{Y}_{t-h,i}| \tag{9}$$

The advantage of the absolute loss function is that it is more robust to the anomaly in the real time series data. In the experiment section, we use the validation set to decide to use which objective function, square loss Eq.7 or absolute one Eq.9.

3.8 Optimization Strategy

In this paper, our optimization strategy is the same as that in the traditional time series forecasting model. Supposing the input time series is $Y_t = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_t\}$, we define a tunable window size q, and reformulate the input at time stamp t as $X_t = \{\boldsymbol{y}_{t-q+1}, \boldsymbol{y}_{t-q+2}, \ldots, \boldsymbol{y}_t\}$. The problem then becomes a regression task with a set of feature-value pairs $\{X_t, Y_{t+h}\}$, and can be solved by Stochastic Gradient Decent (SGD) or its variants such as Adam [18].

4 EVALUATION

We conducted extensive experiments with 9 methods (including our new methods) on 4 benchmark datasets for time series forecasting tasks. All the data and experiment codes are available online [2].

4.1 Methods for Comparison

The methods in our comparative evaluation are the follows.

- AR stands for the autoregressive model, which is equivalent to the one dimensional VAR model.
- LRidge is the vector autoregression (VAR) model with L2-regularization, which has been most popular for multivariate time series forecasting.
- LSVR is the vector autoregression (VAR) model with Support Vector Regression objective function [30] .
- TRMF is the autoregressive model using temporal regularized matrix factorization by [32].
- GP is the Gaussian Process for time series modeling. [11, 28]
- VAR-MLP is the model proposed in [35] that combines Multilayer Perception (MLP) and autoregressive model.
- RNN-GRU is the Recurrent Neural Network model using GRU cell.
- LSTNet-skip is our proposed LSTNet model with skip-RNN layer.
- LSTNet-Attn is our proposed LSTNet model with temporal attention layer.

For the single output methods above such as AR, LRidge, LSVR and GP, we just trained n models independently, i.e., one model for each of the n output variables.

4.2 Metrics

We used three conventional evaluation metrics defined as:

- Root Relative Squared Error (RSE):

$$RSE = \frac{\sqrt{\sum_{(i,t)\in\Omega_{Test}}(Y_{it} - \hat{Y}_{it})^2}}{\sqrt{\sum_{(i,t)\in\Omega_{Test}}(Y_{it} - mean(Y))^2}} \tag{10}$$

- Empirical Correlation Coefficient (CORR)

$$CORR = \frac{1}{n}\sum_{i=1}^{n}\frac{\sum_t\left(Y_{it} - mean(Y_i)\right)\left(\hat{Y}_{it} - mean(\hat{Y}_i)\right)}{\sqrt{\sum_t\left(Y_{it} - mean(Y_i)\right)^2\left(\hat{Y}_{it} - mean(\hat{Y}_i)\right)^2}} \tag{11}$$

where $Y, \hat{Y} \in \mathbb{R}^{n\times T}$ are ground true signals and system prediction signals, respectively. The RSE are the scaled version of the widely used Root Mean Square Error(RMSE), which is design to make more readable evaluation, regardless the data scale. For RSE lower value is better, while for CORR higher value is better.

4.3 Data

We used four benchmark datasets which are publicly available. Table 1 summarizes the corpus statistics.

- Traffic[3]: A collection of 48 months (2015-2016) hourly data from the California Department of Transportation. The data describes the road occupancy rates (between 0 and 1) measured by different sensors on San Francisco Bay area freeways.
- Solar-Energy[4] : the solar power production records in the year of 2006, which is sampled every 10 minutes from 137 PV plants in Alabama State.
- Electricity[5]: The electricity consumption in kWh was recorded every 15 minutes from 2012 to 2014, for n = 321 clients. We converted the data to reflect hourly consumption;
- Exchange-Rate: the collection of the daily exchange rates of eight foreign countries including Australia, British, Canada, Switzerland, China, Japan, New Zealand and Singapore ranging from 1990 to 2016.

All datasets have been split into training set (60%), validation set (20%) and test set (20%) in chronological order. To facilitate future research in multivariate time series forecasting, we publicize all raw datasets and the one after preprocessing in the website.

[1] One could keep ϵ to make the objective function more faithful to the Linear SVR model without modifying the optimization strategy. We leave this for future study.
[2] the link is anonymous due to the double blind policy

[3] http://pems.dot.ca.gov
[4] http://www.nrel.gov/grid/solar-power-data.html
[5] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

Datasets	T	D	L
Traffic	17,544	862	1 hour
Solar-Energy	52,560	137	10 minutes
Electricity	26,304	321	1 hour
Exchange-Rate	7,588	8	1 day

Table 1: Dataset Statistics, where T is length of time series, D is number of variables, L is the sample rate.

In order to examine the existence of long-term and/or short-term repetitive patterns in time series data, we plot autocorrelation graph for some randomly selected variables from the four datasets in Figure 3. Autocorrelation, also known as serial correlation, is the correlation of a signal with a delayed copy of itself as a function of delay deﬁned below

$$R(\tau) = \frac{\mathbb{E}[(X_t - \mu)(X_{t+\tau} - \mu)]}{\sigma^2}$$

where X_t is the time series signals, μ is mean and σ^2

Dataset		Solar-Energy				Traffic				Electricity				Exchange-Rate			
		Horizon				Horizon				Horizon				Horizon			
Methods	Metrics	3	6	12	24	3	6	12	24	3	6	12	24	3	6	12	24
AR	RSE	0.2435	0.3790	0.5911	0.8699	0.5991	0.6218	0.6252	0.6293	0.0995	0.1035	0.1050	0.1054	0.0228	0.0279	**0.0353**	**0.0445**
(3)	CORR	0.9710	0.9263	0.8107	0.5314	0.7752	0.7568	0.7544	0.7519	0.8845	0.8632	0.8591	0.8595	0.9734	0.9656	0.9526	**0.9357**
LRidge	RSE	0.2019	0.2954	0.4832	0.7287	0.5833	0.5920	0.6148	0.6025	0.1467	0.1419	0.2129	0.1280	**0.0184**	0.0274	0.0419	0.0675
(3)	CORR	0.9807	0.9568	0.8765	0.6803	0.8038	0.8051	0.7879	0.7862	0.8890	0.8594	0.8003	0.8806	**0.9788**	**0.9722**	0.9543	0.9305
LSVR	RSE	0.2021	0.2999	0.4846	0.7300	0.5740	0.6580	0.7714	0.5909	0.1523	0.1372	0.1333	0.1180	0.0189	0.0284	0.0425	0.0662
(1)	CORR	0.9807	0.9562	0.8764	0.6789	0.7993	0.7267	0.6711	0.7850	0.8888	0.8861	0.8961	0.8891	0.9782	0.9697	**0.9546**	0.9370
TRMF	RSE	0.2473	0.3470	0.5597	0.9005	0.6708	0.6261	0.5956	0.6442	0.1802	0.2039	0.2186	0.3656	0.0351	0.0875	0.0494	0.0563
(0)	CORR	0.9703	0.9418	0.8475	0.5598	0.6964	0.7430	0.7748	0.7278	0.8538	0.8424	0.8304	0.7471	0.9142	0.8123	0.8993	0.8678
GP	RSE	0.2259	0.3286	0.5200	0.7973	0.6082	0.6772	0.6406	0.5995	0.1500	0.1907	0.1621	0.1273	0.0239	**0.0272**	0.0394	0.0580
(1)	CORR	0.9751	0.9448	0.8518	0.5971	0.7831	0.7406	0.7671	0.7909	0.8670	0.8334	0.8394	0.8818	0.8713	0.8193	0.8484	0.8278
VARMLP	RSE	0.1922	0.2679	0.4244	0.6841	0.5582	0.6579	0.6023	0.6146	0.1393	0.1620	0.1557	0.1274	0.0265	0.0304	0.0407	0.0578
(0)	CORR	0.9829	0.9655	0.9058	0.7149	0.8245	0.7695	0.7929	0.7891	0.8708	0.8389	0.8192	0.8679	0.8609	0.8725	0.8280	0.7675
RNN-GRU	RSE	0.1932	0.2628	0.4163	0.4852	0.5358	0.5522	0.5562	0.5633	0.1102	0.1144	0.1183	0.1295	0.0192	0.0264	0.0408	0.0626
(0)	CORR	0.9823	0.9675	0.9150	0.8823	0.8511	0.8405	0.8345	0.8300	0.8597	0.8623	0.8472	0.8651	0.9786	0.9712	0.9531	0.9223
LST-Skip	RSE	0.1843	0.2559	**0.3254**	0.4643	**0.4777**	**0.4893**	**0.4950**	**0.4973**	**0.0864**	**0.0931**	0.1007	**0.1007**	0.0226	0.0280	0.0356	0.0449
(17)	CORR	0.9843	0.9690	**0.9467**	0.8870	**0.8721**	**0.8690**	**0.8614**	**0.8588**	**0.9283**	**0.9135**	**0.9077**	**0.9119**	0.9735	0.9658	0.9511	0.9354
LST-Attn	RSE	**0.1816**	**0.2538**	0.3466	**0.4403**	0.4897	0.4973	0.5173	0.5300	0.0868	0.0953	**0.0984**	0.1059	0.0276	0.0321	0.0448	0.0590
(7)	CORR	**0.9848**	**0.9696**	0.9397	**0.8995**	0.8704	0.8669	0.8540	0.8429	0.9243	0.9095	0.9030	0.9025	0.9717	0.9656	0.9499	0.9339

Table 2: Results summary (in RSE and CORR) of all methods on four datasets: 1) each row has the results of a specific method in a particular metric; 2) each column compares the results of all methods on a particular dataset with a specific horizon value; 3) bold face indicates the best result of each column in a particular metric; and 4) the total number of bold-faced results of each method is listed under the method name within parentheses.

datasets to show the existence of repetitive patterns in the Solar-Energy, Traffic and Electricity datasets but not in Exchange-Rate. The current results provide empirical evidence for the success of LSTNet models in modeling long-term and short-term dependency patterns when they do occur in data. Otherwise, LSTNet performed comparably with the better ones (AR and LRidge) among the representative baselines.

Compared the results of univariate AR with that of the multivariate baseline methods (LRidge, LSVR and RNN), we see that in some datasets, i.e. Solar-Energy and Traffic, the multivariate approaches is stronger, but weaker otherwise, which means that the richer input information would causes overfitting in the traditional multivariate approaches. In contrast, the LSTNet has robust performance in different situations, partly due to its autoregressive component, which we will discuss further in Section 4.6.

4.6 Ablation Study

To demonstrate the efficiency of our framework design, a careful ablation study is conducted. Specifically, we remove each component one at a time in our LSTNet framework. First, we name the LSTNet without different components as follows.

- **LSTw/oskip**: The LSTNet models without the Recurrent-skip component and attention component.
- **LSTw/oCNN**: The LSTNet-skip models without the Convolutional component.
- **LSTw/oAR**: The LSTNet-skip models without the AR component.

For different baselines, we tune the hidden dimension of models such that they have similar numbers of model parameters to the completed LSTNet model, removing the performance gain induced by model complexity.

The test results measured using RSE and CORR are shown in Figure 5[6]. Several observations from these results are worth highlighting:

- The best result on each dataset is obtained with either LST-Skip or LST-Attn.
- Removing the AR component (in LSTw/oAR) from the full model caused the most significant performance drops on most of the datasets, showing the crucial role of the AR component in general.
- Removing the Skip and CNN components in (LSTw/oCNN or LSTw/oskip) caused big performance drops on some datasets but not all. All the components of LSTNet together leads to the robust performance of our approach on all the datasets.

The conclusion is that our architecture design is most robust across all experiment settings, especially with the large horizons.

As for why the AR component would have such an important role, our interpretation is that AR is generally robust to the scale changing in data. To empirically validate this intuition we plot one dimension (one variable) of the time series signals in the electricity consumption dataset for the duration from 1 to 5000 hours in Figure 6, where the blue curve is the true data and the red curve is the system-forecasted signals. We can see that the true consumption suddenly increases around the 1000th hour, and that LSTNet-Skip successfully captures this sudden change but LSTw/oAR fails to react properly.

[6]We omit the results in RAE as it shows similar comparison with respect to the relative performance among the methods.

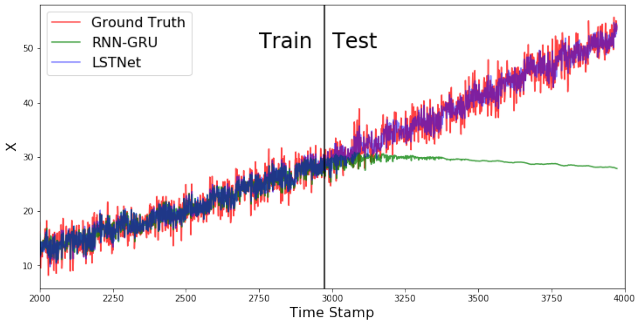


Figure 4: Simulation Test: Left side is the training set and right side is test set.

In order to better verify this assumption, we conduct a simulation experiment. First, we randomly generate an autoregressive process with the scale changing by the following steps. Firstly, we randomly sample a vector, $w \sim N(0, I), w \in \mathbb{R}^p$, where p is a given window size. Then the generated autoregressive process x_t can be described as

$$x_t = \sum_{i=1}^{p} w_i x_{t-i} + \epsilon \tag{12}$$

where $\epsilon \sim N(\mu, 1)$. To inject the scale changing, we increase the mean of Gaussian noise by μ_0 every T timestamp. Then the Gaussian noise of time series x_t can be written as

$$\epsilon \sim N(\lfloor t/T \rfloor \mu_0, 1) \tag{13}$$

where the $\lfloor \cdot \rfloor$ denotes the floor function. We split the time series as the training set and test in chronological order, and test the RNN-GRU and the LSTNet models. The result is illustrated in Figure 4. Both RNN and LSTNet can memorize the pattern in training set (left side). But, the RNN-GRU model cannot follow the scale changing pattern in the test set (right side). Oppositely, the LSTNet model fits the test set much better. In other words, the normal RNN module, or says the neural-network component in LSTNet, may not be sufficiently sensitive to violated scale fluctuations in data (which is typical in Electricity data possibly due to random events for public holidays or temperature turbulence, etc.), while the simple linear AR model can make a proper adjustment in the forecasting.

In summary, this ablation study clearly justifies the efficiency of our architecture design. All components have contributed to the excellent and robust performance of LSTNet.

4.7 Mixture of long- and short-term patterns

To illustrate the success of LSTNet in modeling the mixture of short-term and long-term recurring patterns in time series data, Figure 7 compares the performance of LSTNet and VAR on an specific time series (one of the output variables) in the Traffic dataset. As discussed in Section 4.3, the Traffic data exhibit two kinds of repeating patterns, i.e. the daily ones and the weekly ones. We can see in Figure 7 that the true patterns (in blue) of traffic occupancy are very different on Fridays and Saturdays, and another on Sunday and Monday. The Figure 7 is the prediction result of the VAR model (part (a)) and LSTNet (part (b)) of a traffic flow monitor sensor, where their hyper-parameters are chosen according to the RMSE result on the validation set. The figure shows that the VAR model is only capable to deal with the short-term patterns. The pattern of prediction results of the VAR model only depend on the day before the predictions. We can clearly see that the results of it in Saturday (2rd and 9th peaks) and Monday (4th and 11th peaks) is different from the ground truth, where the ground truth of Monday (weekday) has two peaks, one peak for Saturday (weekend). In the contrary, our proposed LSTNet model performs two patterns for weekdays and weekends respectfully. This example proves the ability of LSTNet model to memorize short-term and long-term recurring patterns simultaneously, which the traditional forecasting model is not equipped, and it is crucial in the prediction task of the real world time series signals.

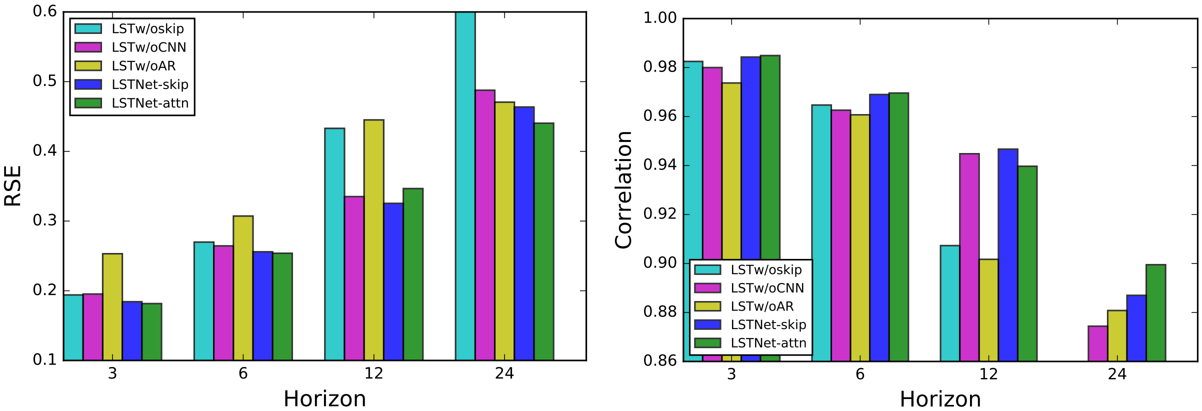
5 CONCLUSION

In this paper, we presented a novel deep learning framework (LSTNet) for the task of multivariate time series forecasting. By combining the strengths of convolutional and recurrent neural networks and an autoregressive component, the proposed approach significantly improved the state-of-the-art results in time series forecasting on multiple benchmark datasets. With in-depth analysis and empirical evidence, we show the efficiency of the architecture of LSTNet model, and that it indeed successfully captures both short-term and long-term repeating patterns in data, and combines both linear and non-linear models for robust prediction.

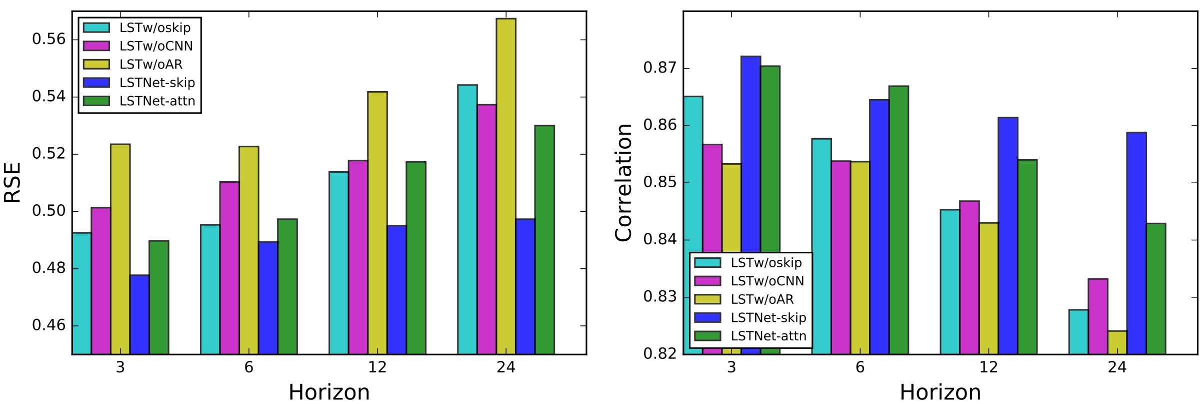
For future research, there are several promising directions in extending the work. Firstly, the skip length p of the skip-recurrent layer is a crucial hyper-parameter. Currently, we manually tune it based on the validation dataset. How to automatically choose p according to data is an interesting problem. Secondly, in the convolution layer we treat each variable dimension equally, but in the real world dataset, we usually have rich attribute information. Integrating them into the LSTNet model is another challenging problem.

REFERENCES

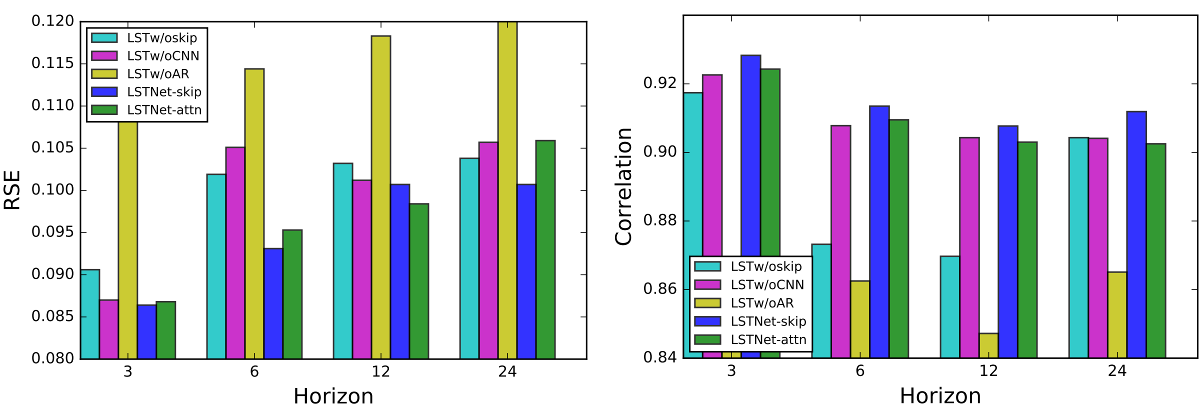
[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
[2] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.
[3] G. E. Box and D. A. Pierce. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American statistical Association*, 65(332):1509–1526, 1970.
[4] L.-J. Cao and F. E. H. Tay. Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on neural networks*, 14(6):1506–1518, 2003.
[5] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu. Recurrent neural networks for multivariate time series with missing values. *arXiv preprint arXiv:1606.01865*, 2016.
[6] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.
[7] J. Connor, L. E. Atlas, and D. R. Martin. Recurrent networks and narma modeling. In *NIPS*, pages 301–308, 1991.
[8] S. Dasgupta and T. Osogami. Nonlinear dynamic boltzmann machines for time-series prediction. *AAAI-17. Extended research report available at goo. gl/Vd0wna*, 2016.
[9] J. L. Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.
[10] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen. Bayesian inference and learning in gaussian process state-space models with particle mcmc. In *Advances*



(a) Solar-Energy **dataset**



(b) Traffic **dataset**



(c) Electricity **dataset**

Figure 5: Results of LSTNet in the ablation tests on the Solar-Energy**,** Traffic **and** Electricity **dataset**

in Neural Information Processing Systems, pages 3156–3164, 2013.

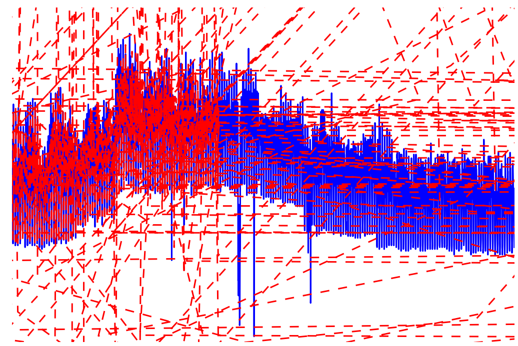
[11] R. Frigola-Alcade. *Bayesian Time Series Learning with Gaussian Processes*. PhD thesis, PhD thesis, University of Cambridge, 2015.

[12] J. D. Hamilton. *Time series analysis*, volume 2. Princeton university press Princeton, 1994.

[13] N. Y. Hammerla, S. Halloran, and T. Ploetz. Deep, convolutional, and recurrent models for human activity recognition using wearables. *arXiv preprint arXiv:1604.08880*, 2016.

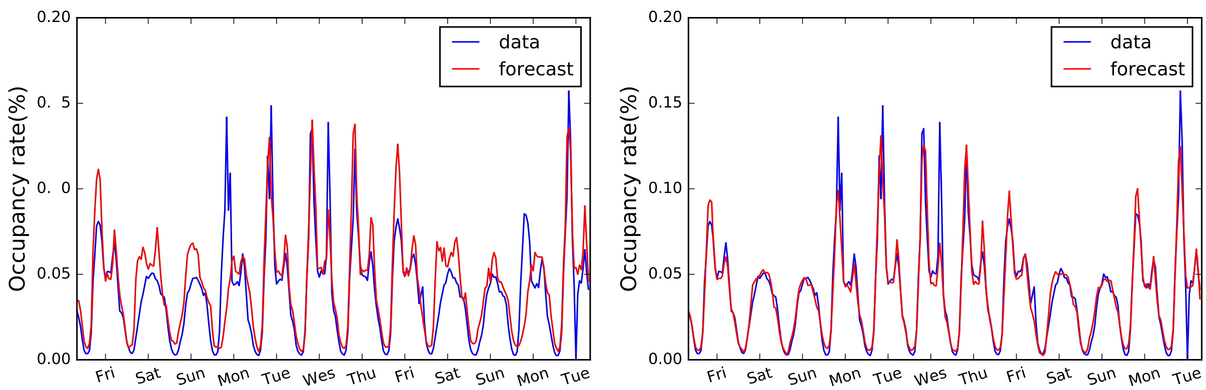
[14] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

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



(a) (b)

Figure 6: The predicted time series (red) by LSTw/oAR (a) and by LST-Skip (b) vs. the true data (blue) on Electricity **dataset with** $horizon = 24$



(a) (b)

Figure 7: The true time series (blue) and the predicted ones (red) by VAR (a) and by LSTNet (b) for one variable in the Traffic **occupation dataset. The X axis indicates the week days and the forecasting** $horizon = 24$**. VAR inadequately predicts similar patterns for Fridays and Saturdays, and ones for Sundays and Mondays, while LSTNet successfully captures both the daily and weekly repeating patterns.**

[16] A. Jain and A. M. Kumar. Hybrid neural network models for hydrologic time series forecasting. *Applied Soft Computing*, 7(2):585–592, 2007.

[17] K.-j. Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1):307–319, 2003.

[18] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[20] C. Lea, R. Vidal, A. Reiter, and G. D. Hager. Temporal convolutional networks: A unified approach to action segmentation. In *Computer Vision–ECCV 2016 Workshops*, pages 47–54. Springer, 2016.

[21] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[22] J. Li and W. Chen. Forecasting macroeconomic time series: Lasso-based approaches and their forecast combinations with dynamic factor models. *International Journal of Forecasting*, 30(4):996–1015, 2014.

[23] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzell. Learning to diagnose with lstm recurrent neural networks. *arXiv preprint arXiv:1511.03677*, 2015.

[24] H. Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

[25] E. McKenzie. General exponential smoothing and the equivalent arma process. *Journal of Forecasting*, 3(3):333–344, 1984.

[26] I. Melnyk and A. Banerjee. Estimating structured vector autoregressive model. *arXiv preprint arXiv:1602.06606*, 2016.

[27] H. Qiu, S. Xu, F. Han, H. Liu, and B. Caffo. Robust estimation of transition matrices in high dimensional heavy-tailed vector autoregressive processes. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1843–1851, 2015.

[28] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain. Gaussian processes for time-series modelling. *Phil. Trans. R. Soc. A*, 371(1984):20110550, 2013.

[29] R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[30] V. Vapnik, S. E. Golowich, A. Smola, et al. Support vector method for function approximation, regression estimation, and signal processing. *Advances in neural information processing systems*, pages 281–287, 1997.

[31] J. B. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy. Deep convolutional neural networks on multichannel time series for human activity recognition. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI), Buenos Aires, Argentina*, pages 25–31, 2015.

[32] H.-F. Yu, N. Rao, and I. S. Dhillon. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in Neural Information Processing Systems*, pages 847–855, 2016.
[33] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu. Deep learning: A generic approach for extreme condition tra c forecasting. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 777–785. SIAM, 2017.
[34] G. Zhang, B. E. Patuwo, and M. Y. Hu. Forecasting with arti cial neural networks:: The state of the art. *International journal of forecasting*, 14(1):35–62, 1998.
[35] G. P. Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175, 2003.