Multivariate Time Series Forecasting

Suyash Ghuge
16IT114
Department of Information Technology
National Institute of Technology Karnataka
Mangalore, India
suyashghuge@gmail.com

Shreyas Shankar
16IT138
Department of Information Technology
National Institute of Technology Karnataka
Mangalore, India
shreyas.shankar920@gmail.com

***Abstract*—Multivariate time series forecasting is a vital machine learning problem across several domains, as well as predictions of solar plant energy output, electricity consumption, and traffic congestion state of affairs. Temporal data arises in these real-world applications typically involves a combination of long and short patterns, for which traditional approaches like Autoregressive models and Gaussian Process might fail. In the implemented paper [1], a unique deep learning framework is proposed, namely Long- and short term Time-series network (LSTNet), to deal with this open challenge. LSTNet uses the Convolution Neural Network (CNN) and then the Recurrent Neural Network (RNN) to extract short native dependency patterns among variables and to find long patterns for time series trends. Moreover, autoregressive model is included in the architecture to tackle the scale insensitive drawback of the neural network model. In the evaluation on real-world knowledge with complicated mixtures of repetitive patterns, LSTNet achieves important performance enhancements over that of many progressive baseline ways. LSTNet without the skip or autoregressive components yields quite low accuracy which is compared in the Results and Analysis Section. The correlation obtained by us for traffic(0.8659) and solar(0.9087) datasets are better than the author's which were 0.8429 and 0.8995 respectively.**

***Index Terms*—Multivariate Time Series, Neural Network, LSTNet, CNN, RNN, autoregressive model.**

I. Introduction

Multivariate time series data are present in our everyday life starting from the prices available in markets, the traffic flows on highways, the outputs of solar energy plants, the temperatures across different cities. In such applications, users are usually fascinated by the forecasting of the new trends or potential venturous events supported by historical observations on time series signals. For example, a far better plan can be devised supported by the anticipated traffic jam patterns some hours ahead, and a bigger profit can be created with the forecasting of the near-future stock market. Multivariate time series forecasting usually faces a significant analysis challenge, that is, the way to capture and leverage the dynamics dependencies among multiple variables. Specifically, real-world applications usually entail a combination of short-run and long term patterns. Let us say we are considering the hourly occupancy rate of a freeway. Apparently, there can be 2 continuation patterns, daily and weekly. An efficient time series forecasting model ought to be capture each type of pattern for correct predictions. As another example, contemplate the task of predicting the output of a solar energy farm supported by the measured radiation by large sensors over totally different locations. The long term patterns mirror the distinction between days vs. nights, summer vs. winter, etc., and therefore the short term patterns mirror the consequences of cloud movements, wind direction changes, etc. Again, while not taking each types of recurrent patterns under consideration, correct time series forecasting isn't potential. However, ancient approaches like autoregressive strategies [2–4] let down in this aspect, as most of them don't distinguish the two types of patterns nor model their interactions explicitly and dynamically. Addressing such limitations of existing strategies in time series foreacsting is the main focus of this project, that we have a tendency to implement a unique framework that takes benefits of recent developments in deep learning analysis. Deep neural networks are intensively studied in connected domains, and created extraordinary impacts on the solutions of a broad vary of issues. The recurrent neural networks (RNN) models [5], for instance, became preferred in recent natural language processing (NLP). Two variants of RNN particularly, specifically the Long Short Term Memory (LSTM) [17] and the Gated recurrent Unit (GRU) [6], have considerably improved the progressive performance in machine translation, speech recognition and alternative NLP tasks as they will effectively capture the meanings of words supporting the long and short-term dependencies among them in input documents [7]. In the field of computer vision, as another precedent, convolution neural network (CNN) models [8, 9] have demonstrated extraordinary execution by extracting local and shift-invariant options (called "shapelets" in some cases) at differed coarseness levels from input pictures.

II. Literature Survey

One of the foremost distinguished univariate time series models is the autoregressive integrated moving average (ARIMA) model. The recognition of the ARIMA model is because of its statistical properties, also further because of the well-known Box-Jenkins methodology [2] within the model selection procedure. ARIMA models don't seem to be solely adaptational to numerous exponential smoothing techniques [10] however additionally versatile enough to subsume different forms of time series models together with autoregression (AR), moving average (MA) and Autoregressive Moving Average (ARMA). However, ARIMA models, together with

their variants for modeling long-term temporal dependencies [2], are seldom utilized in multivariate time series forecasting because of their high computational value. On the opposite hand, vector autoregression (VAR) is arguably the widely used models in multivariate time series [2] thanks to its simplicity. VAR models naturally extend AR models to the multiple variable setting, that ignores the dependencies between output variables. Vital progress has been created in recent years in various VAR models, together with the elliptical VAR model for heavy-tail time series and structured VAR model [11] for higher interpretations of the dependencies between high dimensional variables, and more. Still, the model capability of VAR grows linearly over the temporal window size and quadratically over the amount of variables. This means, once addressing long-term temporal patterns, the large model is susceptible to overfitting. To alleviate this issue, [12] projected to reduce the first high dimensional signals into lower dimensional hidden representations, then applied VAR for forecasting with an alternative of regularization. Time series forecasting issues may also be treated as normal regression problems with time-varying parameters. It is not shocking that varied regression models with completely different loss functions and regularization terms are applied to time series forecasting tasks. For instance, linear support vector regression (SVR) [13, 14] learns a maximum margin hyperplane supporting the regression loss with a hyper-parameter dominating the threshold of prediction errors. Ridge regression is yet one more example which may be recovered from SVR models by setting to zeros. Lastly, [4] applied LASSO models to encourage sparsity within the model parameters so interesting patterns among completely different input signals might be manifested. These linear strategies are much a lot of economical for multivariate time series forecasting thanks to high-quality off-the-shelf solvers within the machine learning community. However, like VARs, those linear models might fail to capture complicated non-linear relationships of variable signals, leading to an inferior performance at the price of its efficiency. Gaussian Processes (GP) may be a non-parametric methodology for modeling distributions over a continuous domain of functions. This contrasts with models outlined by a parameterized category of functions like VARs and SVRs. GP can be applied to multiple variable time series forecasting task as instructed in [15], and may be used as a prior over the function space in Bayesian inference. For instance, [16] gives a fully Bayesian approach with the GP prior for nonlinear state-space models, that is capable of capturing complicated phenomena. However, the importance of Gaussian method comes with the worth of high computation complexness. An easy implementation of Gaussian method for variable time-series forecasting has cubic complexness over the amount of observations, thanks to the matrix operation of the kernel matrix.

III. Outcome of Literature Survey

We encountered the Autoregressive Integrated Moving Average (ARIMA) model for forecasting and control of Time Series in the literature. Advantages of ARIMA model are adaptiveness to various exponential smoothing techniques and also flexible enough to subsume other types of time series models including autoregression (AR), moving average (MA) and Moving Average (ARMA). The limitation here is that they have a very high computational cost. Next was Vector Autoregression (VAR) model for Time Series Analysis. They are easy to estimate. They have good forecasting capabilities. The researcher does not need to specify which variables are endogenous or exogeneous. All are endogenous. However The model capacity of VAR grows linearly over the temporal window size and quadratically over the number of variables. This infers, when managing long-term temporal patterns, the acquired extensive model is inclined to overfitting. Linear Support Vector Regression (SVR) are linear models. They are more efficient for multivariate time series forecasting due to high-quality off-the-shelf solvers in the machine learning community. The limitation is that they fail to capture complex non-linear relationships of multivariate signals, resulting in an inferior performance at the cost of its efficiency. LASSO models have sparsity in the model parameters so that interesting patterns among different input signals could be manifested. They fail to capture complex non-linear relationships of multivariate signals, resulting in an inferior performance at the cost of its efficiency. Gaussian processes are 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.

IV. Motivation

Following are the motivations behind implementing this project:

1) Various models like ARIMA, VAR, GP, etc. have been used for multivariate time series forecasting. Time series forecasting is a critical region of machine learning.
2) It is important because there are so many prediction problems that involve a time component, like predicting stock values for the next few seconds.
3) The perceptions inside every series are not free of one another, the likelihood of finding a high connection between the two series might be higher than is proposed by standard equations.
4) New methodologies using Deep Neural Networks have been applied to make the predictions more efficient.

Thus with the above motivation we have chosen to implement the research paper.

V. Problem Statement

Given a progression of completely watched time series signals, we aim to design and develop a system that predicts a series of future signals in a rolling forecasting fashion. We assume that the time series values for a particular range are given and then do the prediction for a required horizon over the time series.

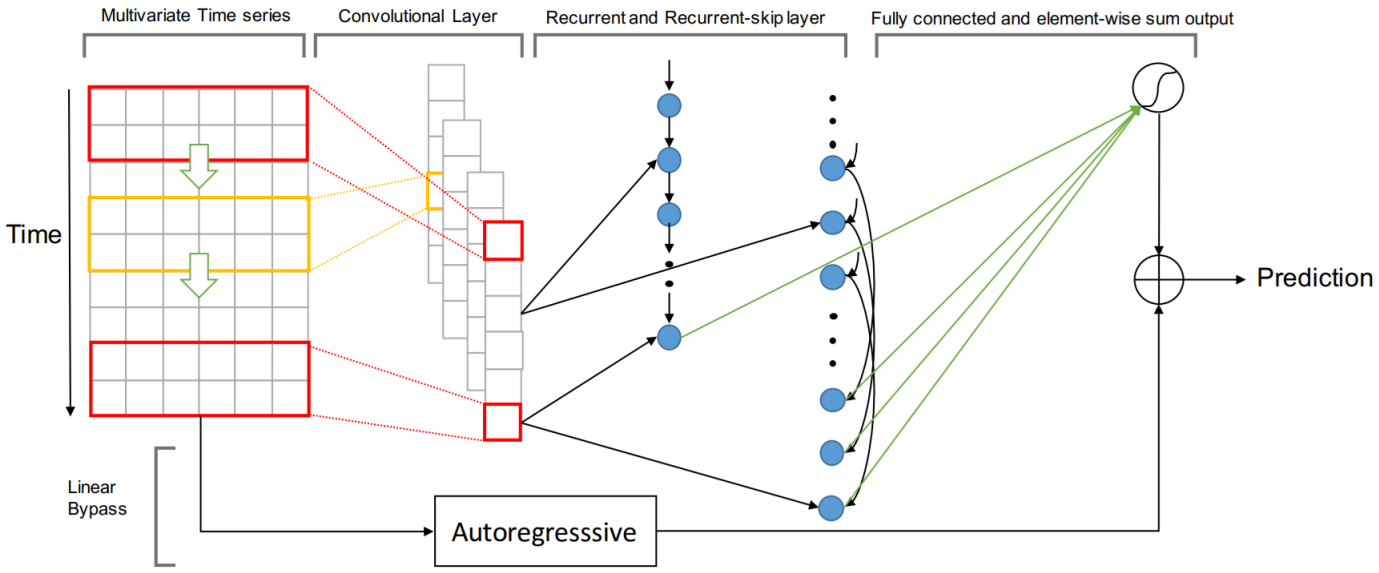

Fig. 1. Model Architecture.

VI. Objectives

To demonstrate the efficiency of our framework design, a careful ablation study will be conducted. Specifically, we will remove each component one at a time in our LSTNet framework.

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

We will compare the results obtained from these and the original framework to prove its efficiency.

VII. Proposed Model

Above is the diagram (Fig. 1) illustrating the proposed model. This model is named LSTNet.

A. Convolutional Component

The first layer of LSTNet is a convolutional network without pooling. This layer aims to extract short-term patterns in the time dimension. Also, it tries to exact local dependencies between variables. The convolutional layer makes use of multiple filters of width and height n (the height is set to be the same as the number of variables). The k-th filter clears through the info matrix X and produces hk = RELU (Wk X + bk) where denotes the convolution operation and the output hk would be a vector, and the RELU function is RELU (x) = max(0, x).

B. Recurrent Component

The output of the convolutional layer is fed into the Recurrent component and Recurrent-skip component at the same time. The Recurrent component uses the Gated Recurrent Unit (GRU) [6]. It makes use of the RELU function as the hidden update activation function.

C. Recurrent-skip Component

The Recurrent layers with GRU [6] and LSTM [17] unit are scrupulously designed to memorise the historical data and thus to bear in mind the long term dependencies. Because of gradient vanishing, however, GRU and LSTM typically fail to capture long term dependencies. The paper tends to propose to alleviate this issue via a unique recurrent-skip element that leverages the periodic pattern in real-world sets. A recurrent structure with temporal skip connections to increase the temporal span of the data flow and thus to ease the optimisation method is developed. Specifically, skip-links are added between the present hidden cell and the hidden cells within the same phase in adjacent periods.

D. Temporal Attention Layer

However, the Recurrent-skip layer needs a predefined hyper-parameter p, that is unfavorable within the non seasonal time series forecasting, or whose amount length is dynamic over time. To alleviate such issue, another approach is proposed, attention mechanism, that learns the weighted combination of hidden representations at every window position of the input matrix.

E. Autoregressive Component

Due to the non-linear nature of the Convolutional and recurrent elements, one major downside of the neural network model is that the size of outputs is not sensitive to the scale of inputs. Unfortunately, in specific real datasets, the size of input signals frequently changes during a non-periodic manner, that considerably lowers the prediction accuracy of the neural

network model. To deal with this deficiency, similar in spirit to the highway network, we tend to decompose the ultimate prediction of LSTNet into a linear part, that primarily focuses on the local scaling issue, and a non-linear part containing recurrent patterns. Within the LSTNet design, we adopt the classical Autoregressive (AR) model as the linear element.

F. Optimization Strategy

Our optimization strategy is the same as that in the traditional time series forecasting model. Supposing the input time series is Yt = y1,y2, . . . ,yt , we define a tunable window size q, and reformulate the input at time stamp t as Xt = yt-q+1, yt-q+2, . . . , yt . The problem then becomes a regression task with a set of feature-value pairs Xt ,Yt+h , and can be solved by Stochastic Gradient Decent (SGD) or its variants such as Adam.

VIII. Work Done

Initial step is build data iterators. Suppose we need to construct a model which can foresee the electricity utilization of any house h estimations into the future, given the past q electricity utilization readings for all houses.

The first layer added to the LSTNet is a convolutional network of 6 layers without pooling, which aims to extract short-term patterns in the time dimension as well as local dependencies between variables. These convolutional features are utilized in two segments ofthe network, which are recurrent components.

The output of the convolutional layer is fed into the Recurrent component. The Recurrent component is a recurrent layer consisting of 100 neurons with the Gated Recurrent Unit (GRU) and uses the RELU function as the hidden update activation function.

Because of gradient vanishing be that as it may, GRU and LSTM more often than not neglect to catch long-term correlation in practice. The output of the convolutional layer is also fed into the Recurrent skip component of 120 neurons. Specifically, skip-links of length 5 are added between the current hidden cell and the hidden cells in the same phase in adjacent periods.

The output from the RNN and skip-RNN layers are fed to a temporal attention layer of 220 neurons.

We have adopted the classical Autoregressive (AR) model as the linear component of 24 neurons which takes in the original input. The sum of the output from the autogressive, temporal attention components goes through a sigmoid activation function. It is used to predict the future value for every time series

We have incorporated Adam optimizer to complete the LSTNet model.

Made the code runnable in google colabs cuda10 runtime

Trained and tested with four datasets including exchange rate, electricity, solar and traffic

IX. Results and Analysis

Datasets		**LSTw/oSkip**	**LSTw/oAR**	**LSTNet**
Stock CORR	Author	-	-	0.9339
	Us	0.9243	0.7328	0.9362
Electric CORR	Author	-	-	0.9025
	Us	0.8874	0.8504	0.8983
Traffic CORR	Author	-	-	0.8429
	Us	0.8628	0.8501	0.8659
Solar CORR	Author	-	-	0.8995
	Us	0.9102	0.9070	0.9087

Comparison of our work with author's

The performance of 3 models has been compared in the above table, for all the 4 datasets. Our work has been compared with the author's work and this comparison is on the basis of Empirical Correlation Coefficient (CORR). LSTw/oSkip model is the LSTNet model without the Recurrentskip component and attention component. LSTw/oAR is the LSTNet-skip model without the AR component. These two models have not been implemented in the author's paper. We can see that for Stock-exchange dataset, LSTNet implementation of Author gives a CORR value of 0.9339 whereas our implementation gives a value of 0.9362. For the same dataset, LSTw/oSkip gives CORR 0.9243 and LSTw/oAR gives CORR 0.7328. For the Electricity dataset, Author's implementation gives CORR 0.9025 whereas our implementation gives CORR 0.8983. LSTw/oSkip and LSTw/oAR gives CORR 0.8874 and 0.8504 respectively. For the Traffic dataset, Author's implementation gives CORR 0.8429 whereas our implementation gives CORR 0.8659. LSTw/oSkip and LSTw/oAR gives CORR 0.8628 and 0.8501 respectively. For the Solar-energy dataset, Author's implementation gives CORR 0.8995 whereas our implementation gives CORR 0.9087. LSTw/oSkip and LSTw/oAR gives CORR 0.9102 and 0.9070 respectively.

We can thus infer that our implementation outperforms Author's implementation for 3 out of the 4 used datasets. These are Stock-exchange, Traffic and Solar-energy datasets. However, these values are bound to change every time the model is run. Thus we calculate average values over certain number of iterations. We also observe that the LSTNet model outperforms the LSTw/oSkip and LSTw/oAR models. There is only 1 case where the LSTw/oSkip model performs better than LSTNet model, this is in the case of Solar-energy dataset. However, LSTNet can be thus considered to be more efficient than LSTw/oSkip and LSTw/oAR models.

Dataset	**CORR**	**RSE**
Stock Exchange	93.62	0.531
Electricity	89.83	0.100
Solar Energy	90.87	0.4282
Traffic	86.59	0.4969

Final Results

Above table gives the consolidated results for all the results, with respect to our implementation of the LSTNet model. Here RSE is the Root Relative Squared Error. RSE is 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.

For the Stock-exchange dataset CORR is 93.62% and RSE is 0.531. For Electricty dataset, CORR and RSE are 89.83% and 0.1 respectively. For Solar-energy dataset, CORR and RSE are 90.87% and 0.4282 respectively. For Traffic dataset, CORR and RSE are 86.59% and 0.4969 respectively. Thus we can see that the model performs best for Stock-exchange dataset in terms of CORR whereas it works best for Electricity dataset in terms of RSE.

X. CONCLUSION

We have implemented a unique 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 part, the proposed approach considerably improved the currently existing results in time series forecasting on multiple benchmark datasets. With in-depth analysis and empirical proof, we infer that the efficiency of the LSTNet model is higher, and that it successfully captures each short-term and long-term repeating patterns in data, and combines each linear and non-linear models for strong prediction. Also, we observe that LSTNet works best for Stock-exchange dataset with CORR value of 93.39%, while performance for Electricity, Traffic, Solar-energy datasets being 90.25%, 84.29% and 89.95% respectively. The LSTw/oSkip and LSTw/oAR models have also been implemented. However, the LSTNet model outperforms these except for once where LSTw/oSkip performs better than LSTNet for the Solar-energy dataset with a CORR of 91.02%. For future research, there are many promising directions in extending the work. Firstly, the skip length p of the skip-recurrent layer could be a crucial hyper-parameter. Currently, we have a tendency to manually tune it supporting the the validation dataset. The way to mechanically select p according to the dataset is a stimulating drawback. Secondly, within the convolution layer we have a tendency to treat every variable dimension equally, however in the real world dataset, we have a attribute information. Clubbing them into the LSTNet model is a challenging problem which can be researched upon.

XI. TIMELINE OF PROJECT

Milestones	***Feb 2018***	***Mar 2018***
Data Preprocessing	21st Feb	
CNN model	28th Feb	
CNN + RNN/skipRNN model		7th Mar
LSTNet model		7th Mar
Adam Optimization		14th Mar
Google Colab Cuda Runtime		21st Mar

XII. INDIVIDUAL CONTRIBUTION

Shreyas Shankar	**Suyash Ghuge**
Data Preprocessing	
LSTNet model	LSTNet model
	Adam Optimizer
Colab Cuda	

REFERENCES

[1] Guokun Lai, Wei-Cheng Chang, Yiming Yang, Hanxiao Liu. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. 21 Mar 2017.

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

[4] 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):9961015, 2014.

[5] J. L. Elman. Finding structure in time. Cognitive science, 14(2):179211, 1990. 15. S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural computation, 9(8):17351780, 1997.

[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] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 10971105, 2012.

[9] 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.

[10] E. McKenzie. General exponential smoothing and the equivalent arma process. Journal of Forecasting, 3(3):333344, 1984.

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

[12] 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 847855, 2016.

[13] 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.

[14] K.-j. Kim. Financial time series forecasting using support vector machines. Neurocomputing, 55(1):307319, 2003.

[15] 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.

[16] R. Frigola, F. Lindsten, T. B. Schn, and C. E. Rasmussen. Bayesian inference and learning in gaussian process state-space models with particle mcmc. In Advances in Neural Information Processing Systems, pages 31563164, 2013.

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural computation, 9(8):17351780, 1997.