HYBRID NEURAL NETWORKS OVER TIME SERIES FOR TREND FORECASTING

Tao Lin[*], **Tian Guo**[*] **& Karl Aberer**
School of Computer and Communication Sciences
Ecole polytechnique federale de Lausanne
Lausanne, Switzerland
{tao.lin, tian.guo, karl.aberer}@epfl.ch

ABSTRACT

The trend of time series characterize the intermediate upward and downward patterns of time series. Learning and forecasting the trend in time series data play an important role in many real applications, ranging from resource allocation in data centers and load schedule in smart grid. Inspired by the recent successes of neural networks, in this paper we propose TreNet, a novel hybrid neural network based learning approach over time series and the associated trend sequence. TreNet leverages convolutional neural networks (CNNs) to extract salient features from local raw data of time series and uses a long-short term memory recurrent neural network (LSTM) to capture the sequential dependency in historical trend evolution. Some preliminary experimental results demonstrate the advantage of TreNet over CNN, LSTM, the cascade of CNN and LSTM, Hidden Markov Model method and various kernel based baselines on real datasets.

1 INTRODUCTION

Time series, as a sequential data points ordered by time, is being generated in a wide spectrum of domains. However, in many applications, users are only interested in understanding and forecasting the evolving trend in time series, i.e., upward or downward pattern of time series, since the conventional prediction on specific data points could deliver very little information about the semantics and dynamics of the underlying process generating the time series. For instance, time series in Figure 1 are from the power consumption dataset. Figure 1(a) shows some raw data points of time series. Though point A and B have approximately the same value, the underlying system is likely to be in two different states when it outputs A and B, because A is in an upward trend while B is in a downward trend (Wang et al., 2011; Matsubara et al., 2014). On the other hand, even when two points with the similar value are both in the upward trend, e.g., point A and C, the different slopes and durations of the trends where point A and C locate, could also indicate different states of the underlying process.

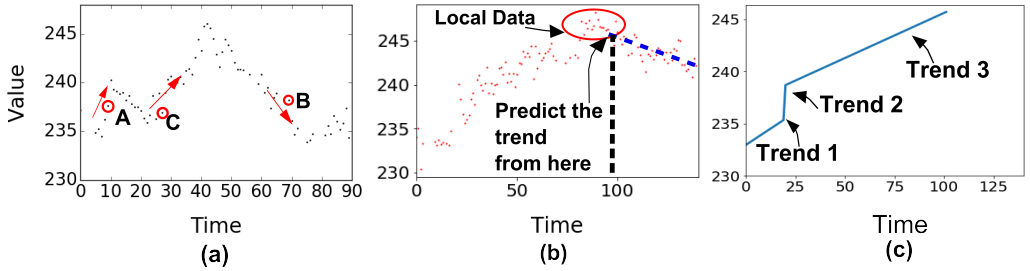

Figure 1: (a) Time series of power consumption. (b) Trend prediction on time series. (c) Sequence of historical trends associated with the time series.

Learning and forecasting trends are quite useful in a wide range of applications, e.g., in the smart energy domain, knowing the predictive trend of power consumption time series enables energy

[*]These two authors contributed equally.

providers to schedule power supply and maximize energy utilization (Zhao & Magoulès, 2012). In this paper, we are particularly interested in the trend of time series, i.e., upward or downward pattern of time series that characterized by the slope and duration (Wang et al., 2011).

Specifically, given a time series and the associated historical trend evolution, we aim to predict the duration and slope of the subsequent trend. It involves the learning of different aspects of the data. On one hand, the trend variation of the time series is a sequence of historical trends describing the long-term contextual information of time series and thus naturally affect the evolution of the following trend. On the other hand, the recent raw data points of time series (Wang et al., 2011; Batal et al., 2012), which represent the local variation and behaviour of time series, affect the evolving of the following trend as well and have particular predictive power for abruptly changing trends (Wang et al., 2011). Therefore, it is highly desired to develop a systematic way to model such various hidden and complementary dependencies in time series.

To this end, we propose an end-to-end hybrid neural network, referred to as TreNet. It consists of a LSTM recurrent neural network to capture the sequential dependency in historical trends, a convolutional neural network to extract local features from local raw data of time series and a feature fusion layer to learn joint representation to take advantage of both features drawn from CNN and LSTM. Such joint representation is then used for the trend forecasting. Some preliminary experimental analysis on real datasets demonstrates that TreNet outperforms RNN, CNN, the cascade of RNN and CNN, and a variety of baselines in term of trend prediction accuracy.

2 HYBRID NEURAL NETWORKS FOR LEARNING THE TREND

In this section, we first provide the formal definition of the trend learning and forecasting problem in this paper, and then present the proposed TreNet.

Problem Formulation.
We define time series as a sequence of data points $\mathcal{X} = \{x_1, \ldots, x_T\}$, where each data point x_t is real-valued and subscript t represents the time instant. The historical *trend sequence* of $\mathcal{X}$ is a series of historical trends over $\mathcal{X}$, denoted by $\mathcal{T} = \{\langle \ell_k, s_k \rangle\}$. Each element of $\mathcal{T}$, i.e., $\langle \ell_k, s_k \rangle$, describes a linear function over a certain subsequence (or segment) of $\mathcal{X}$ and corresponds to a trend in $\mathcal{X}$. ℓ_k and s_k respectively represent the duration and slope of trend k. The *local data w.r.t.* each historical trend in $\mathcal{T}$ is defined as a set of data points of size w, denoted by $\mathcal{L} = \{\langle x_{t_k - w}, \ldots, x_{t_k} \rangle\}$ and t_k is the ending time of trend k in $\mathcal{T}$.

Our ultimate goal is to propose a neural network based approach to learn a function $f(\mathcal{T}, \mathcal{L})$ to predict the subsequent trend $\langle \hat{l}, \hat{s} \rangle$. In this paper, we focus on univariate time series, and the method proposed below can be naturally generalized to multivariate time series.

TreNet.
The idea of our TreNet is to combine CNN with LSTM to utilize their representation abilities on different aspects of training data (i.e., $\mathcal{X}$ and $\mathcal{T}$) and then to learn a joint feature for trend prediction.

Technically, TreNet is designed to learn a predictive function $\langle \hat{l}, \hat{s} \rangle = f(R(\mathcal{T}), C(\mathcal{L}))$. $R(\mathcal{T})$ is derived by training the LSTM over sequence $\mathcal{T}$ to capture the dependency in the trend evolving, while $C(\mathcal{L})$ corresponds to local features extracted by CNN from sets of local data in $\mathcal{L}$. The long-term and local features captured by LSTM and CNN, i.e., $R(\mathcal{T})$ and $C(\mathcal{L})$, convey complementary information pertaining to the trend varying. Therefore, the feature fusion layer is supposed to take advantages of both features to produce a fused features used for forecasting the subsequent trend. Finally, the trend prediction is realized by the function $f(\cdot, \cdot)$, which corresponds to the feature fusion and output layers as is shown in Figure 2.

During the training phase, the duration ℓ_k and slope s_k of each trend k in sequence $\mathcal{T}$ are fed into the LSTM layer of TreNet. Please refer to (Hochreiter & Schmidhuber, 1997) for more details of LSTM. When the k-th trend in $\mathcal{T}$ is fed to LSTM, the corresponding local raw time series data points $\langle x_{t_k - w}, \ldots, x_{t_k} \rangle$ in $\mathcal{L}$ is input to the CNN part of TreNet. CNN consists of H stacked layers of 1-d convolutional, activation and pooling operations. Each layer has a specified number of filters of a specified filter size. The output of CNN in TreNet is the concatenation of max-pooling output on the last layer H (Donahue et al., 2015).

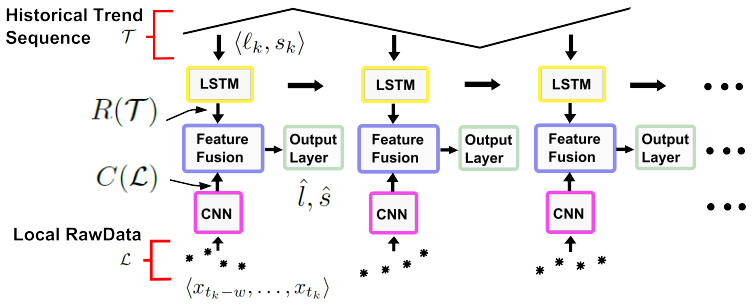

Figure 2: Illustration of the hybrid architecture of TreNet. (best viewed in colour)

The feature fusion layer combines the representations $R(\mathcal{T})$ and $C(\mathcal{L})$, to form a joint feature. Particularly, we first map $R(\mathcal{T})$ and $C(\mathcal{L})$ to the same feature space and add them together to obtain the activation of the feature fusion layer (Mao et al., 2014). The output layer is a fully-connect layer following the feature fusion layer. Mathematically, the prediction of TreNet is expressed as:

$$\langle \hat{l}, \hat{s} \rangle = f(R(\mathcal{T}),\ C(\mathcal{L})) = \boldsymbol{W^o} \cdot \underbrace{\boldsymbol{\phi}(\boldsymbol{W^r} \cdot R(\mathcal{T}) + \boldsymbol{W^c} \cdot C(\mathcal{L}))}_{feature\ fusion} + \boldsymbol{b^o} \tag{1}$$

where $\boldsymbol{\phi}(\cdot)$ is element-wise leaky ReLU activation function and $+$ denotes the element-wise addition. $\boldsymbol{W^o}$ and $\boldsymbol{b^o}$ are the weights and bias of the output layer. TreNet is trained to minimize the squared loss function of durations and slopes.

3 Experimental Analysis

In this section, we report some preliminary experiments to demonstrate the prospect of TreNet. For evaluation, we compare the performance of TreNet with six baselines: CNN, LSTM, the cascade of CNN and LSTM (CLSTM)(Bashivan et al., 2015), Support Vector Regression, Pattern-based Hidden Markov Model (Wang et al., 2011), and Naive. For space limitation, we only report the comparison on Power Consumption (PC) dataset. Regarding the details of experimental setup, training procedure and a complete comparison, please refer to Section 5.

Dataset	Model	RMSE @ Duration	RMSE @ Slope
PC	CNN	27.51	13.56
	LSTM	27.27	13.27
	CLSTM	25.97	13.77
	SVRBF	31.81	12.94
	SVPOLY	31.81	12.93
	SVSIG	31.80	12.93
	pHMM	34.06	26.00
	Naive	39.68	21.17
	TreNet	25.89	12.89

Table 1: RMSE of the prediction of trend duration and slope on each dataset.

Table 1 studies the prediction performances of TreNet and baselines on PC data, and more comparison can be found in Section 5. We can observe that TreNet consistently outperforms baselines on the duration and slope prediction by achieving around 30% less errors at maximum. It verifies that the hybrid architecture of TreNet can improve the performance by utilizing the information captured by both CNN and LSTM. Specifically, pHMM method performs worse due to the limited representation capability of HMM. On the slope prediction, SVR based approaches can get comparable results as TreNet.

4 Conclusion

In this paper, we propose TreNet, a novel hybrid neural network to learn and predict the trend behaviour of time series The preliminary experimental results demonstrate that such a hybrid framework can enhance the prediction performance.

REFERENCES

Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015.

Yukun Bao, Tao Xiong, and Zhongyi Hu. Multi-step-ahead time series prediction using multiple-output support vector regression. *Neurocomputing*, 129:482–493, 2014.

Pouya Bashivan, Irina Rish, Mohammed Yeasin, and Noel Codella. Learning representations from eeg with deep recurrent-convolutional neural networks. *arXiv preprint arXiv:1511.06448*, 2015.

Iyad Batal, Dmitriy Fradkin, James Harrison, Fabian Moerchen, and Milos Hauskrecht. Mining recent temporal patterns for event detection in multivariate time series data. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 280–288. ACM, 2012.

Li-Chiu Chang, Pin-An Chen, and Fi-John Chang. Reinforced two-step-ahead weight adjustment technique for online training of recurrent neural networks. *IEEE transactions on neural networks and learning systems*, 23(8):1269–1278, 2012.

Sucheta Chauhan and Lovekesh Vig. Anomaly detection in ecg time signals via deep long short-term memory networks. In *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*, pp. 1–7. IEEE, 2015.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2625–2634, 2015.

Nils Y Hammerla, Shane Halloran, and Thomas Ploetz. Deep, convolutional, and recurrent models for human activity recognition using wearables. *arXiv preprint arXiv:1604.08880*, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Colin Lea, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks: A unified approach to action segmentation. *arXiv preprint arXiv:1608.08242*, 2016.

Zhen Li and Yizhou Yu. Protein secondary structure prediction using cascaded convolutional and recurrent neural networks. *arXiv preprint arXiv:1604.07176*, 2016.

Zachary C Lipton, David C Kale, Charles Elkan, and Randall Wetzell. Learning to diagnose with lstm recurrent neural networks. *arXiv preprint arXiv:1511.03677*, 2015.

Jiajun Liu, Kun Zhao, Brano Kusy, Ji-rong Wen, and Raja Jurdak. Temporal embedding in convolutional neural networks for robust learning of abstract snippets. *arXiv preprint arXiv:1502.05113*, 2015.

Qi Lyu and Jun Zhu. Revisit long short-term memory: An optimization perspective. In *Advances in neural information processing systems workshop on deep Learning and representation Learning*, 2014.

Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *European Symposium on Artificial Neural Networks*, volume 23, 2015.

Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632*, 2014.

Yasuko Matsubara, Yasushi Sakurai, and Christos Faloutsos. Autoplait: Automatic mining of co-evolving time sequences. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 193–204. ACM, 2014.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Souhaib Ben Taieb and Amir F Atiya. A bias and variance analysis for multistep-ahead time series forecasting. *IEEE transactions on neural networks and learning systems*, 27(1):62–76, 2016.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3156–3164, 2015.

Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, and Wei Xu. Cnn-rnn: A unified framework for multi-label image classification. *arXiv preprint arXiv:1604.04573*, 2016.

Peng Wang, Haixun Wang, and Wei Wang. Finding semantics in time series. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pp. 385–396. ACM, 2011.

Jian Bo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali 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*, pp. 25–31, 2015.

Hai-xiang Zhao and Frédéric Magoulès. A review on the prediction of building energy consumption. *Renewable and Sustainable Energy Reviews*, 16(6):3586–3592, 2012.

5 APPENDIX

In this section, we first introduce some related work and then report all omitted experimental details, including the information of three datasets, the description of baselines, training procedure and the complete performance evaluation on three datasets.

5.1 RELATED WORK

Traditional learning approaches over trends of time series mainly make use of Hidden Markov Models (HMMs) (Wang et al., 2011; Matsubara et al., 2014). HMMs maintain short-term state dependences, i.e., the memoryless Markov property and predefined number of states, which requires significant task specific knowledge. RNNs instead use high dimensional, distributed hidden states that could take into account long-term dependencies in sequence data. Multi-step ahead prediction is another way to realize trend prediction by fitting the predicted values to estimate the trend. However, multi-step ahead prediction is a non-trivial problem itself (Chang et al., 2012) suffers from the accumulative prediction errors (Taieb & Atiya, 2016; Bao et al., 2014). In this paper, we concentrate on training neural networks over time series to learn the trend.

RNNs have recently shown promising results in a variety of applications, especially when there exist sequential dependencies in data (Lyu & Zhu, 2014; Chung et al., 2014; Sutskever et al., 2014). Long short-term memory (LSTM) (Hochreiter & Schmidhuber, 1997; Lyu & Zhu, 2014; Chung et al., 2014), a class of recurrent neural networks with sophisticated recurrent hidden and gated units, are particularly successful and popular due to its ability to learn hidden long-term sequential dependencies. (Lipton et al., 2015) uses LSTMs to recognize patterns in multivariate time series, especially for multi-label classification of diagnoses. (Chauhan & Vig, 2015; Malhotra et al., 2015) evaluate the ability of LSTMs to detect anomalies in ECG time series. CNN is often used to learn effective representation of local salience from raw data (Vinyals et al., 2015; Donahue et al., 2015; Karpathy et al., 2014). (Hammerla et al., 2016; Yang et al., 2015; Lea et al., 2016) make use of CNNs to extract features from raw time series data for activity/action recognition. (Liu et al., 2015) focuses on the prediction of periodical time series values by using CNN and embedding time series with the potential neighbors in the temporal domain.

Hybrid neural networks, which combines the strengths of various neural networks, are receiving increasing interest in the computer vision domain, such as image captioning (Mao et al., 2014; Vinyals et al., 2015; Donahue et al., 2015), image classification (Wang et al., 2016), protein structure prediction (Li & Yu, 2016), action recognition (Ballas et al., 2015; Donahue et al., 2015) and so on. But efficient exploitation of such hybrid architectures has not been well studied for time series data, especially the trend forecasting problem. (Li & Yu, 2016; Ballas et al., 2015) utilize CNNs over images in cascade of RNNs in order to capture the temporal features for classification. (Bashivan et al., 2015) transforms EEG data into a sequence of topology-preserving multi-spectral images and then trains a cascaded convolutional-recurrent network over such images for EEG classification. (Wang et al., 2016; Mao et al., 2014) propose the CNN-RNN framework to learn a shared representation for image captioning and classification problems.

5.2 EXPERIMENT SETUP

Dataset: We test our method and baselines on three real time series datasets.

- **Power Consumption** (PC). This dataset[1] contains measurements of electric power consumption in one household with a one-minute sampling rate over a period of almost 4 years. Different electrical quantities and some sub-metering values are available. We use the voltage time series throughout the experiments.
- **Gas Sensor** (GasSensor). This dataset[2] contains the recordings of chemical sensors exposed to dynamic gas mixtures at varying concentrations. The measurement was constructed by the continuous acquisition of the sensor array signals for a duration of about 12

[1] https://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption

[2] https://archive.ics.uci.edu/ml/datasets/Gas+sensor+array+under+dynamic+gas+mixtures

hours without interruption. We mainly use the gas mixture time series regarding Ethylene and Methane in air.

- **Stock Transaction** (Stock): This dataset is extracted from Yahoo Finance and contains the daily stock transaction information in New York Stock Exchange from 1950-10 to 2016-4.

For the ease of experimental result interpretation, the slope of the trends is represented by the angle of the corresponding linear function and thus in a bounded value range $[-90, 90]$. The duration of trends is measured by the number of data points within the trend.

Baselines: We compare TreNet with the following six baselines:

- **CNN**. This baseline method predicts the trend by using CNN over the raw data of time series.
- **LSTM**. This method uses LSTM to learn dependencies in the trend sequence $\mathcal{T}$ and predicts the trend.
- **ConvNet+LSTM(CLSTM)**. It is based on the cascade structure of ConvNet and LSTM in (Bashivan et al., 2015) which feeds the features learnt by ConvNet over time series to a LSTM and obtains the prediction from the LSTM.
- **Support Vector Regression (SVR)**. A family of support vector regression based approaches with different kernel methods is used for the trend forecasting. We consider three commonly used kernels (Liu et al., 2015), i.e., Radial Basis kernel (**SVRBF**), Polynomial kernel (**SVPOLY**), Sigmoid kernel (**SVSIG**). The trend sequence and the corresponding set of local time series data are concatenated as the input features to such SVR approaches.
- **Pattern-based Hidden Markov Model (pHMM)**. (Wang et al., 2011) proposed a pattern-based hidden Markov model (HMM), which segments the time series and models the dependency in segments via HMM. The derived HMM model is used to predict the state of time series and then to estimate the trend.
- **Naive**. This is the naive approach which takes the duration and slope of the last trend as the prediction for the next one.

Evaluation metric: We evaluate the predictive performance of TreNet and baselines in terms of Root Mean Square Error (RMSE). The lower the RMSE, the more accurate the predictions.

Training: In TreNet, CNN has two stacked convolutional layers, which have 32 filters of size 2 and 4. The number of memory cells in LSTM is 600. In addition to the learning rate, the number of neurons in the feature fusion layer is chosen from the range $\{300, 600, 900, 1200\}$ to achieve the best performance. The window size in TreNet is chosen by cross validation. We use dropout and L2 regularization to control the capacity of neural networks to prevent overfitting, and set the values to 0.5 and 5×10^{-4} respectively for all datasets (Mao et al., 2014). The Adam optimizer (Kingma & Ba, 2014) is chosen to learn the weights in neural networks. Regarding the SVR based approaches, we carefully tune the parameters c (error penalty), d (degree of kernel function), and γ (kernel coefficient) for kernels. Each parameter is selected from the sets $c \in \{10^{-5}, 10^{-4}, \ldots, 1, \ldots, 10^4, 10^5\}$, $d \in \{1, 2, 3\}$, $\gamma \in \{10^{-5}, 10^{-4}, \ldots, 1, \ldots, 10^5\}$ respectively.

5.3 Experiment Results

Table 2 continues the evaluation in Table 1 and more thoroughly studies the prediction performances of TreNet and baselines.

Dataset	Model	RMSE @ Duration	RMSE @ Slope
PC	CNN	27.51	13.56
	LSTM	27.27	13.27
	CLSTM	25.97	13.77
	SVRBF	31.81	12.94
	SVPOLY	31.81	12.93
	SVSIG	31.80	12.93
	pHMM	34.06	26.00
	Naive	39.68	21.17
	TreNet	25.89	12.89
Stock	CNN	18.87	12.78
	LSTM	11.07	8.40
	CLSTM	9.26	7.31
	SVRBF	11.38	7.40
	SVPOLY	11.40	7.42
	SVSIG	11.49	7.41
	pHMM	36.37	8.70
	Naive	11.36	8.58
	TreNet	8.86	6.84
GasSensor	CNN	53.99	11.51
	LSTM	55.77	11.22
	CLSTM	54.20	14.86
	SVRBF	62.81	10.21
	SVPOLY	70.91	10.95
	SVSIG	85.69	11.92
	pHMM	111.62	13.07
	Naive	53.76	10.57
	TreNet	52.28	9.57

Table 2: RMSE of the prediction of local trend duration and slope on each dataset.