공학석사학위논문

다 변수 시계열 예측을 위한 주의 기반 LSTM 모델 연구

Study on Attention-based LSTM Model for Multivariate Time-series Prediction

2019년 8월

서울대학교 대학원
컴퓨터공학부
Yang Bai (바이양)

다 변수 시계열 예측을 위한 주의 기반 LSTM 모델 연구

Study on Attention-based LSTM Model for Multivariate Time-series Prediction

지도교수 강 유

이 논문을 공학석사 학위논문으로 제출함

2019년 6월

서울대학교 대학원

컴퓨터공학부

Yang Bai (바이양)

Yang Bai (바이양)의 석사 학위논문을 인준함

2019년 7월

위원장 전병곤 (인)

부위원장 강 유 (인)

위원 김건희 (인)

Abstract

Study on Attention-based LSTM Model for Multivariate Time-series Prediction

Yang Bai
Department of Computer Science & Engineering
The Graduate School
Seoul National University

Given previous observations of a multivariate time-series, how can we accurately predict the future value of several steps ahead? With the continuous development of sensor systems and computer systems, time-series prediction techniques are playing more and more important roles in various fields, such as finance, energy, and traffics. Many models have been proposed for time-series prediction tasks, such as Autoregressive model, Vector Autoregressive model, and Recurrent Neural Networks (RNNs). However, these models still have limitations like failure in modelling non-linearity and long-term dependencies in time-series. Among all the proposed approaches, the Temporal Pattern Attention (TPA), which is an attention-based LSTM model, achieves state-of-the-art performance on several real-world multivariate time-series datasets.

In this thesis, we study three factors that effect the prediction performance of TPA model, which are the Recurrent Neural Network RNN layer, the attention mechanism, and the Convolutional Neural Network for temporal patter detection. For recurrent layer, we implement bi-directional LSTMs that can extract information from the input sequence in both forward and backward directions. In addition, we design two attention mechanisms, each of which assigns attention weights in different directions. We study the effect of both attention mechanisms on TPA model. Finally, to validate the Convolutional Neural Network (CNN) for temporal pattern detection, we implement a TPA model without CNN. We test all of these factors using several real-world time-series datasets from different fields. The experimental results indicate the validity of these factors.

Keywords : Time-series, Attention mechanism, LSTM, Prediction

Student Number : 2017-22669

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Given previous observations of a multivariate time-series, how can we accurately predict the future value of several steps ahead? Nowadays, the highly developed sensor systems and computer systems are generating massive time-series data all the time, such as household electricity consumption, road occupancy rate, currency exchange rate, and solar power production. Since most of time-series data come from practical fields, it is of great significance to predict these time-series. The accurate prediction of time-series can help people better manage the resources, for example, prediction of household electricity consumption can help the power supply department to properly distribute the electrical load. Another example is that people can make great profit by precisely predicting the stock prices and currency exchange rates.

Time-series prediction and modelling is an important interdisciplinary research, involving Computer Science and Statistics. Traditional statistical predicting approaches combine linear Autoregressive (AR) and moving average. However, since time-series data often consist of complex non-linear patterns, these traditional approaches cannot work well all the time. Therefore, the need for non-linear predicting approaches arises.

In machine learning, the model can automatically learn the useful patterns (both linear and non-linear) from data. However, even though traditional machine learning models (e.g. support vector regression) can extract both linear and non-linear patterns in time-series, they fail to model the long-term dependencies through time steps. Recently, deep learning models, especially the Recurrent Neural Networks (RNNs), are widely used in time-series prediction tasks because of their ability to model the long-term dependencies. Long Short-term Memory (LSTM) neural network is a kind of RNN that has proven to be successful in processing time-series data. With an input gate and a forget gate, an LSTM network can selectively memorize or forget historical information in its memory cells, which makes modelling the long-term dependencies possible.

Even though LSTM networks work well in time-series data, the attention mechanism can further improve their performance. The key idea of attention mechanism is that it first assesses the previous states of RNN layers and selects the relevant ones, then it extracts a context vector form these states. Because the context vector is a mixture of previous relevant states, it could be used to improve the RNN performance.

Temporal Pattern Attention (TPA) model is a multivariate time-series predicting model. With LSTM and attention mechanism, the TPA model can make satisfying prediction on multivariate time-series data. To our best knowledge, the TPA model achieves state-of-the-art prediction accuracy in several real-world multivariate time-series datasets. In this thesis, we introduce our study on Temporal Pattern Attention model and determine the effects of three main components of TPA model. More specifically, we study how recurrent neural network layer, attention mechanism, and convolutional neural network for temporal pattern detection influence the performance of TPA model. The contributions of this thesis are summarized as below:

1. We compare the effect of bi-directional LSTM and uni-directional LSTM layers in the TPA model.

2. We implement two attention mechanisms, which are horizontal and vertical

attentions. We apply both attention mechanisms in TPA model and study their differences.

3. We study the influence of convolutional neural network for temporal pattern detection.
4. We did extensive experiments to validate the three aforementioned components in TPA model. The experimental results show their effects.

The rest of this thesis is organized as follows. In Section 2, we introduce preliminaries. In Section 3, we formally define the multivariate time-series prediction problem, and then describe our study details. Experimental results are presented in Section 4. After discussing related works in Section 5, we conclude in Section 6.

Chapter 2

Preliminaries

In this section, we explain the preliminaries, that is, Long Short-term Memory, attention mechanism, and Temporal Pattern Attention model.

2.1 Long Short-term Memory

Long Short-term Memory (LSTM) is a variant of Recurrent Neural Network that can model long-term dependencies. In the LSTM cell, an input gate, a forget gate, and an output gate can control the passing of information, which can capture long-term

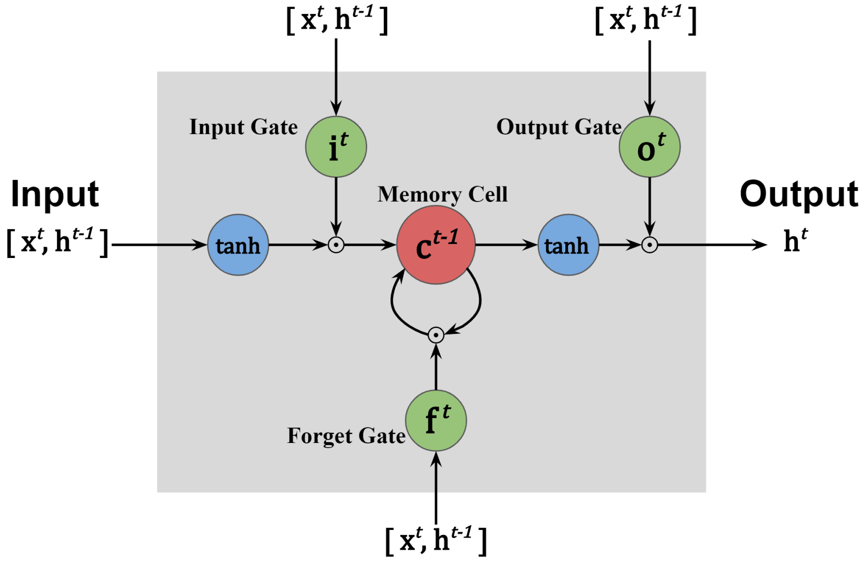


Figure 1: LSTM cell

dependencies along time. Figure 1 shows the structure of an LSTM cell. In an LSTM cell, at each time step t, hidden state $\mathbf{h}^t \in \mathbb{R}^m$ is updated by current input data at the same time step $\mathbf{x}^t \in \mathbb{R}^d$, the hidden state at the previous time step $\mathbf{h}^{t-1}$, the input gate $\mathbf{i}^t$, the forget gate $\mathbf{f}^t$, the output gate $\mathbf{o}^t$, and a memory cell $\mathbf{c}^t$. The updating equations are given as follows:

$$\begin{aligned}
\mathbf{i}^t &= \sigma(\mathbf{W}_i\mathbf{x}^t + \mathbf{U}_i\mathbf{h}^{t-1} + \mathbf{b}_i) \\
\mathbf{f}^t &= \sigma(\mathbf{W}_f\mathbf{x}^t + \mathbf{U}_f\mathbf{h}^{t-1} + \mathbf{b}_f) \\
\mathbf{o}^t &= \sigma(\mathbf{W}_o\mathbf{x}^t + \mathbf{U}_o\mathbf{h}^{t-1} + \mathbf{b}_o) \\
\mathbf{c}^t &= \mathbf{f}^t \odot \mathbf{c}^{t-1} + \mathbf{i}^t \odot tanh(\mathbf{W}_c\mathbf{x}^t + \mathbf{U}_c\mathbf{h}^{t-1} + \mathbf{b}_c) \\
\mathbf{h}^t &= \mathbf{o}^t \odot tanh(\mathbf{c}^t)
\end{aligned} \tag{2.1}$$

where the weights and bias to be computed during training process are $W_i, W_o, W_f, W_c \in \mathbb{R}^{m\times d}, U_i, U_o, U_f, U_c \in \mathbb{R}^{m\times m}$, and $b_i, b_o, b_f, b_c \in \mathbb{R}^{m\times 1}$. The symbol "$\odot$" is element-wise multiplication of two vectors (Hadamard product). The symbol "σ" is element-wise logistic sigmoid activation function. $tanh$ is element-wise hyperbolic tangent activation function.

2.2 Typical Attention Mechanism

The typical attention mechanism selectively extracts information from the previous hidden states [1]. More specifically, in an RNN, given the previous hidden states $\mathbf{H} = \{\mathbf{h}^1, \mathbf{h}^2, ..., \mathbf{h}^{t-1}\}$ and the current hidden state $\mathbf{h}^t$, a context vector $\mathbf{v}^t$ is computed as a weighted sum of each column $\mathbf{h}^i$ in $\mathbf{H}$, which represents the information relevant to the current step. Then, $\mathbf{v}^t$ can be further combined with present hidden state $\mathbf{h}^t$ to compute the prediction.

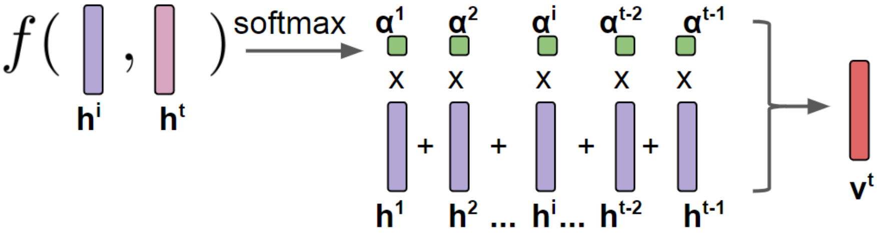


Figure 2: Typical attention mechanism: for each hidden state $\mathbf{h}^i$, we first compute its relevance value with $\mathbf{h}^t$, then put the relevance value through a $softmax$ function and get its attention weight α^i. Finally, we calculate the weighted sum of all hidden states, which is the context vector $\mathbf{v}^t$.

Assume a scoring function $f : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$, which evaluates the relevance between two input vectors, the following formula can calculate the context vector $\mathbf{v}^t$:

$$\begin{aligned} \alpha^i &= \frac{exp(f(\mathbf{h}^i, \mathbf{h}^t))}{\sum_{j=1}^{t-1} exp(f(\mathbf{h}^j, \mathbf{h}^t))} \\ \mathbf{v}^t &= \sum_{i=1}^{t-1} \alpha^i \mathbf{h}^i \end{aligned} \tag{2.2}$$

2.3 Temporal Pattern Attention Model

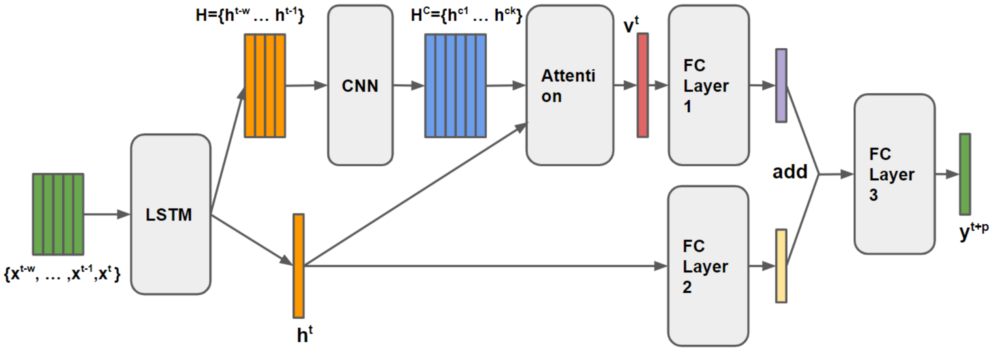


Figure 3: Temporal Pattern Attention Model.

To our best knowledge, Temporal Pattern Attention [2] (TPA) model is the state-of-the-art model for multivariate time-series prediction. As shown in Figure 3, the TPA model consists of four parts: an LSTM layer, a Convolutional Neural Network for temporal pattern detection, a temporal pattern attention mechanism, and three fully-connected layers for regression. The workflow of TPA model can be described as four steps:

Step 1. At time step t, the LSTM layer computes the hidden state $\mathbf{h}^t$ for the current observation $\mathbf{x}^t$.

Step 2. Given w previous hidden states $\mathbf{H} = \{\mathbf{h}^{t-w}, \mathbf{h}^{t-w+1}, ..., \mathbf{h}^{t-1}\}$, where $\mathbf{h}^i \in \mathbb{R}^m$, the convolutional layer extracts temporal patterns from the hidden states by applying CNN filters on the row vectors of $\mathbf{H}$. Specifically, we have k filters $\mathbf{C}_i \in \mathbb{R}^{1\times w}$, where w is size of the window. Convolutional operations yield $\mathbf{H}^C \in \mathbb{R}^{m\times k}$, where $\mathbf{H}^C_{i,j}$ represents the convolutional value of the i-th row vector and the j-th filter [2]. Formally, this operation is given by

$$\mathbf{H}^C_{i,j} = \sum_{l=1}^{w} \mathbf{H}_{i,(t-w-1+l)} \mathbf{C}_{j,l}.$$

Step 3. A temporal pattern attention mechanism is applied to the CNN output $\mathbf{H}^C$. The model computes the weighted sum of row vectors of $\mathbf{H}^C$, instead of computing weighted sum of columns as typical attention mechanism does. Defined below is the scoring function $f : \mathbb{R}^k \times \mathbb{R}^m \to \mathbb{R}$ that evaluates relevance:

$$f(\mathbf{H}^C_i, \mathbf{h}^t) = (\mathbf{H}^C_i)^T \mathbf{W}_a \mathbf{h}^t$$

where $\mathbf{H}^C_i$ is the i-th row vector of $\mathbf{H}^C$, and $\mathbf{W}_a \in \mathbb{R}^{k\times m}$ is a trainable matrix.

The attention weight α^i is calculated by

$$\alpha^i = sigmoid(f(\mathbf{H}_i^C, \mathbf{h}^t)).$$

Unlike the typical attention mechanism which obtains α^i via $softmax$ function, the temporal pattern attention mechanism uses $sigmoid$ function instead. By using $sigmoid$ function, the attention mechanism can find more than one useful variables for forecasting [2]. Finally, the context vector $\mathbf{v}^t \in \mathbb{R}^k$ is computed as the weighted sum of row vectors of $\mathbf{H}^C$:

$$\mathbf{v}^t = \sum_{i=1}^{m} \alpha^i \mathbf{H}_i^C$$

Step 4. Given hidden state $\mathbf{h}^t$ and context vector $\mathbf{v}^t$, the final prediction is calculated by three fully-connected layers:

$$\begin{aligned} \mathbf{h\prime}^t &= \mathbf{W}_h \mathbf{h}^t + \mathbf{W}_v \mathbf{v}^t \\ \mathbf{y}^{t+p} &= \mathbf{W}_{h\prime} \mathbf{h\prime}^t \end{aligned}$$

where $\mathbf{h}^t, \mathbf{h\prime}^t \in \mathbb{R}^m$, $\mathbf{W}_h \in \mathbb{R}^{m \times m}$, $\mathbf{W}_v \in \mathbb{R}^{m \times k}$, $\mathbf{W}_{h\prime} \in \mathbb{R}^{d \times m}$, and $\mathbf{y}^{t+p} \in \mathbb{R}^d$.

According to the paper [2], the TPA model achieved state-of-the-art performance on several multivariate time-series. However, we are interested in which component of TPA model can really contribute to its good performance. Therefore, we design several variants of TPA, to help further analyse the performance of TPA model.

Chapter 3

Study on Temporal Pattern Attention Model

In this section, we describe our ablation study on Temporal Pattern Attention Model. Before introducing our study details, we first give a formal definition of the multivariate time-series prediction problem in Section 3.1. After presenting the overview of our study in Section 3.2, we describe the recurrent neural network layer in Section 3.3. Then we introduce two different attention mechanisms in Section 3.4. Finally, we describe the TPA model without CNN for temporal pattern detection in Section 3.5.

3.1 Problem Definition

In this thesis, we are interested in multivariate time-series prediction. Formally, given time-series data $\mathbf{X} = \{\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^t\}$ in time order, where $\mathbf{x}^i \in \mathbb{R}^d$ represents the observed values at time step i, the task is to predict the value of $\mathbf{x}^{t+p} \in \mathbb{R}^d$, where $p \geq 1$ is a fixed horizon of prediction. We denote the corresponding prediction as $\mathbf{y}^{t+p}$, and the groundtruth value as $\hat{\mathbf{y}}^{t+p} = \mathbf{x}^{t+p}$. In this thesis, given all the observations, we use only the current observation $\mathbf{x}^t$ and previous w observations $\{\mathbf{x}^{t-w}, \mathbf{x}^{t-w+1}, ..., \mathbf{x}^{t-2}, \mathbf{x}^{t-1}\}$ to predict $\mathbf{x}^{t+p}$. This is based on the assumption that only the observations inside the window are useful for prediction, which is a common practice [3, 4].

3.2 Overview

In this thesis, we study the effects of three components in the TPA model: recurrent neural network layers (Step 1 in Section 2.3), attention mechanism (Step 3 in Section 2.3), and Convolutional Neural Network (CNN) for temporal pattern detection (Step 2 in Section 2.3). Given historical observations $\{\mathbf{x}^{t-w}, \mathbf{x}^{t-w+1}, ..., \mathbf{x}^{t-1}, \mathbf{x}^{t}\}$, the recurrent neural network layer generates hidden states for each time step. For this recurrent layer, we will study the effect of bi-directional LSTM layer compared with uni-directional LSTM layer, which is used in the original TPA model. Then we compare the typical attention mechanism with the attention mechanism proposed in TPA model. The main difference between these two attention mechanisms is the direction of attention. Finally, the CNN for temporal pattern detection is studied to determine its validity.

3.3 Recurrent Neural Network Layer

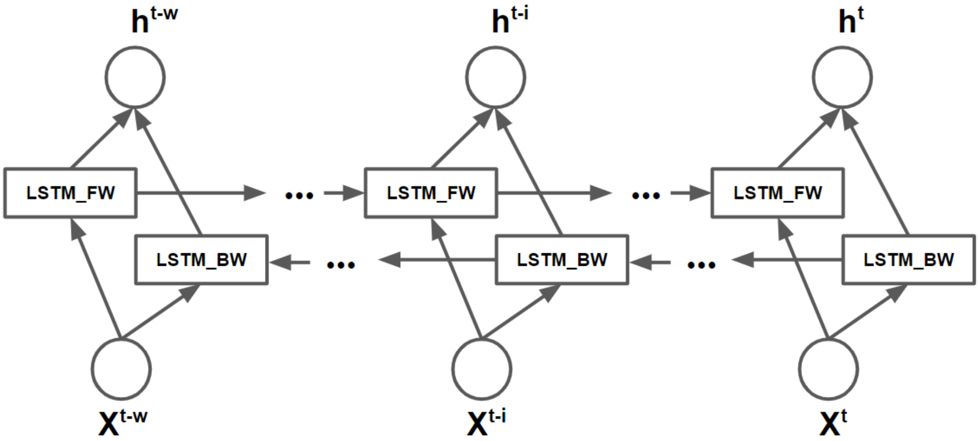


Figure 4: Bi-directional LSTM layer.

For the recurrent neural network layer, we wonder if a bi-directional LSTM layer could improve the performance than the uni-directional LSTM layer used in the original TPA model. The uni-directional LSTM layer is only able to access the previous context of each specific time step. However, time-series data have strong temporal dependencies along time, which makes it meaningful to consider the future context. Therefore, it is natural to replace the uni-directional LSTM with bi-directional LSTM. As shown in Figure 4, a bi-directional LSTM layer is able to process the sequence data in two directions including forward and backward ways with two separate LSTM layers. Then the hidden states of both LSTM layers will be concatenated to form the output of bi-directional LSTM layer. Eq. (3.1) defines the updating equations of the forward LSTM layer:

$$
\begin{aligned}
&\overrightarrow{\mathbf{i}^t} = \sigma(\overrightarrow{\mathbf{W}_i}\ \mathbf{x}^t + \overrightarrow{\mathbf{U}_i}\overrightarrow{\mathbf{h}^{t-1}} + \overrightarrow{\mathbf{b}_i}) \\
&\overrightarrow{\mathbf{f}^t} = \sigma(\overrightarrow{\mathbf{W}_f}\ \mathbf{x}^t + \overrightarrow{\mathbf{U}_f}\overrightarrow{\mathbf{h}^{t-1}} + \overrightarrow{\mathbf{b}_f}) \\
&\overrightarrow{\mathbf{o}^t} = \sigma(\overrightarrow{\mathbf{W}_o}\ \mathbf{x}^t + \overrightarrow{\mathbf{U}_o}\overrightarrow{\mathbf{h}^{t-1}} + \overrightarrow{\mathbf{b}_o}) \\
&\overrightarrow{\mathbf{c}^t} = \overrightarrow{\mathbf{f}^t} \odot \overrightarrow{\mathbf{c}^{t-1}} + \overrightarrow{\mathbf{i}^t} \odot tanh(\overrightarrow{\mathbf{W}_c}\ \mathbf{x}^t + \overrightarrow{\mathbf{U}_c}\overrightarrow{\mathbf{h}^{t-1}} + \overrightarrow{\mathbf{b}_c}) \\
&\overrightarrow{\mathbf{h}^t} = \overrightarrow{\mathbf{o}^t} \odot tanh(\overrightarrow{\mathbf{c}^t})
\end{aligned}
\tag{3.1}
$$

Eq. (3.2) defines the updating equations of the backward LSTM layer:

$$
\begin{aligned}
\overset{\leftarrow}{\mathbf{i}}^t &= \sigma(\overset{\leftarrow}{\mathbf{W}}_i\ \mathbf{x}^t + \overset{\leftarrow}{\mathbf{U}}_i \overset{\leftarrow}{\mathbf{h}}^{t+1} + \overset{\leftarrow}{\mathbf{b}}_i) \\
\overset{\leftarrow}{\mathbf{f}}^t &= \sigma(\overset{\leftarrow}{\mathbf{W}}_f\ \mathbf{x}^t + \overset{\leftarrow}{\mathbf{U}}_f \overset{\leftarrow}{\mathbf{h}}^{t+1} + \overset{\leftarrow}{\mathbf{b}}_f) \\
\overset{\leftarrow}{\mathbf{o}}^t &= \sigma(\overset{\leftarrow}{\mathbf{W}}_o\ \mathbf{x}^t + \overset{\leftarrow}{\mathbf{U}}_o \overset{\leftarrow}{\mathbf{h}}^{t+1} + \overset{\leftarrow}{\mathbf{b}}_o) \\
\overset{\leftarrow}{\mathbf{c}}^t &= \overset{\leftarrow}{\mathbf{f}}^t \odot \overset{\leftarrow}{\mathbf{c}}^{t+1} + \overset{\leftarrow}{\mathbf{i}}^t \odot tanh(\overset{\leftarrow}{\mathbf{W}}_c\ \mathbf{x}^t + \overset{\leftarrow}{\mathbf{U}}_c \overset{\leftarrow}{\mathbf{h}}^{t+1} + \overset{\leftarrow}{\mathbf{b}}_c) \\
\overset{\leftarrow}{\mathbf{h}}^t &= \overset{\leftarrow}{\mathbf{o}}^t \odot tanh(\overset{\leftarrow}{\mathbf{c}}^t)
\end{aligned}
\tag{3.2}
$$

Finally, the complete bi-directional LSTM hidden state $\mathbf{h}^t$ is the concatenated vector of the outputs of forward and backward LSTM layers as follows:

$$
\mathbf{h}^t = [\overset{\rightarrow}{\mathbf{h}}^t \cdot \overset{\leftarrow}{\mathbf{h}}^t] \tag{3.3}
$$

where $\mathbf{h}^t \in \mathbb{R}^{2m}$, obviously.

3.4 Vertical v.s. Horizontal Attention Mechanism

Given previous hidden states $\mathbf{H} = \{\mathbf{h}^{t-w}, \mathbf{h}^{t-w+1}, ..., \mathbf{h}^{t-2}, \mathbf{h}^{t-1}\}$ from the recurrent layer, there are two methods to compute the context vector:

1) Method 1 is assigning attention weights to each column of $\mathbf{H}$ as Bahdanau et al. [1] did.

2) Method 2 is assigning attention weights to each row instead, similar to the TPA model in Section 2.3.

The first method achieved great success in NLP tasks because it managed to find the most relevant word to the current output. Moreover, in NLP tasks, each time step only contains a single piece of information, which shows the first method to its best advantage. However, for multivariate time-series prediction tasks, there are more than one variable in each time step, making information in one time step more complicated. Therefore, the second method is proposed to extract the dependencies among multiple variables. To work out which method works better in multivariate time-series prediction, we implemented both of these two attention mechanisms in our study.

For the sake of convenience, we call the first method ”vertical attention” because it vertically assigns attention weights to columns. Correspondingly, we call the second method ”horizontal attention”.

Vertical Attention assigns the attention weights to each column of $\mathbf{H}$ and computes the weighted sum of columns as the context vector. Here we adopt the same type of scoring function as TPA attention:

$$f(\mathbf{h}^i, \mathbf{h}^t) = (\mathbf{h}^i)^T \mathbf{W}_v \mathbf{h}^t$$

where $\mathbf{W}_v$ is a trainable matrix.

Horizontal Attention assigns attention weights to each row of hidden states $\mathbf{H}$ and then computes the weighted sum of rows:

$$f(\mathbf{H}_j, \mathbf{h}^t) = (\mathbf{H}_j)^T \mathbf{W}_h \mathbf{h}^t$$

where $\mathbf{H}_j$ is the j-th row vector of $\mathbf{H}$ and $\mathbf{W}_h$ is a trainable matrix.

For both vertical and horizontal attention mechanisms, we also use $sigmoid$ function to compute attention weights as in TPA model. Then, we compute the weighted sum of columns or rows to obtain the context vector $\mathbf{v}^t$. It should be noted that, in the experiments, we only use one of these two mechanisms at a time.

3.5 Temporal Pattern Attention Model without CNN

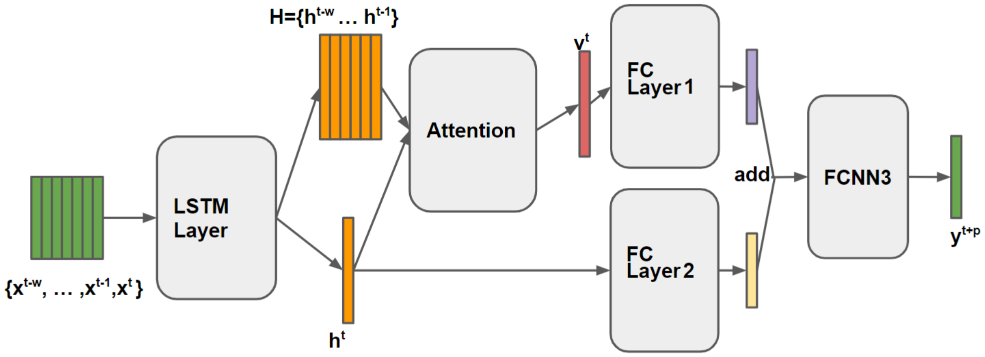


Figure 5: Temporal Pattern Attention model without Convolutional Neural Network for temporal pattern detection.

To determine the validity of Convolutional Neural Network (CNN) for temporal pattern detection, we implement a TPA model without the CNN layer. The original TPA model inputs the previous hidden states $\mathbf{H}$ in to a CNN layer and gets a feature map $\mathbf{H}^C$, then it applies attention onto $\mathbf{H}^C$, as shown in Figure 3. However, to study the effect of CNN, we implement a TPA model without CNN, where the attention mechanism is directly applied to the previous hidden states $\mathbf{H}$, as shown in Figure 5.

Chapter 4

Experiments

We present experimental results to answer the following questions:

- **Q1. (Overall performance)** How do the TPA model and its variants perform on the datasets? (Section 4.2)
- **Q2. (Effects of bi-directional LSTM)** In TPA model, does bi-directional LSTM improve the prediction accuracy? (Section 4.3)
- **Q3. (Effects of CNN for temporal pattern detection)** Compared with TPA without CNN, does the CNN for temporal pattern detection in TPA model improve the prediction accuracy? (Section 4.4)
- **Q4. (Which attention direction is better)** In TPA model, which attention direction is better, horizontal or vertical? (Section 4.5)

4.1 Experimental Setup

Evaluation Metrics. We use Root Relative Squared Error (RSE) as evaluation metric for our experiments:

$$RSE = \frac{\sqrt{\sum_{t=1}^{T}\sum_{i=1}^{d}(\mathbf{y}_i^{t+p} - \hat{\mathbf{y}}_i^{t+p})^2}}{\sqrt{\sum_{t=1}^{T}\sum_{i=1}^{d}(\hat{\mathbf{y}}_i^{t+p} - mean(\hat{\mathbf{Y}}_i))^2}}$$

where $\mathbf{y}_i^{t+p}$ and $\hat{\mathbf{y}}_i^{t+p}$ are the i-th variable of prediction and groundtruth value, respectively, and $mean(\hat{\mathbf{Y}}_i)$ is the mean value of all the i-th variables in the groundtruth data. Smaller RSE indicates better performance since it means the predicted value is closer to the real value.

Datasets. We conduct experiments on four real-world datasets, which are among energy, traffic, and economic fields [2].

- **Solar Energy**: the solar power production records in the year of 2006, which is sampled every 10 minutes from 137 PV plants in Alabama State, US.
- **Traffic**: 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 free ways.
- **Electricity**: the electricity consumption in kWh that was recorded every 15 minutes from 2012 to 2014, for 321 clients.
- **Exchange Rate**: the collection of the daily exchange rates of eight countries including Australia, British, Canada, Switzerland, China, Japan, New Zealand, and Singapore, from 1990 to 2016.

The detailed information of datasets is shown in Table 1. We separate each dataset into three parts: training set, validate set, and testing set. The training set takes the first 60% of the whole dataset. The validate set takes the middle 20%. Finally, the testing set takes the last 20% of the dataset. Further more, to validate the performance decrease along prediction time, we conduct experiments on prediction horizon of 3, 6, 12, and 24 steps ahead.

Competitors. First, we choose two statistical models, which are Autoregressive (AR) model and Vector Autoregressive (VAR) model, as the baseline. Then, to validate

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

Table 1: Dataset statistics.

dataset	# of records	# of attributes	Sampling Spacing	Data Size
Solar Energy[1]	52,560	137	10 minutes	172M
Traffic[2]	17,544	862	1 hour	130M
Electricity[3]	26,304	321	1 hour	91M
Exchange Rate	7,588	8	1 day	534K

the three components of TPA, we implement all possible combinations as competitors. In total, there are eight TPA-based models, including the original TPA model and its seven variants. The description of each model are as below:

- AR: standard autoregressive model, which is the classic uni-variate time-series predicting model.
- VAR: vector autoregressive model, which is a variant of AR model and is able to predict multivariate time-series.
- TPA-h: the original Temporal Pattern Attention model with CNN layer, using uni-directional LSTM and horizontal attention.
- TPA-v: TPA model with CNN layer, using uni-directional LSTM and vertical attention.
- Uni-w/oCNN-h: TPA model without CNN layer, using uni-directional LSTM and horizontal attention.
- Uni-w/oCNN-v: TPA model without CNN layer, using uni-directional LSTM and vertical attention.
- Bi-TPA-h: TPA model with CNN layer, using bi-directional LSTM and horizontal attention.
- Bi-TPA-v: TPA model with CNN layer, using bi-directional LSTM and vertical

attention.

- Bi-w/oCNN-h: TPA model without CNN layer, using bi-directional LSTM and horizontal attention.
- Bi-w/oCNN-v: TPA model without CNN layer, using bi-directional LSTM and vertical attention.

Because the AR model is a uni-variate time-series model, we train one AR model for each variable in the time-series.

Parameter Settings and Model Training. Here we present the model setup and parameter settings of TPA model and all its variants. On Solar Energy, Traffic, and Electricity datasets, the window size w is 24, the number of hidden units m for a single LSTM layer is 25 or 45, and the learning rate is 10^{-3}. On Exchange Rate dataset, the window size, number of hidden units, and learning rate are 30, 6, and $3 \cdot 10^{-3}$. We set the CNN filter number as 32 for all the TPA-based models. Lastly, we use the Mean Absolute Error (MAE) as the loss function for all of these models:

$$loss = \frac{1}{n} \sum_{t=1}^{n} \sum_{i=1}^{d} |\mathbf{y}_i^{t+p} - \hat{\mathbf{y}}_i^{t+p}|.$$

We use Adam optimizer with a decay rate of 0.995 to train each model for 100 epochs.

4.2 Performance Comparison (Q1)

Table 2 shows the Root Relative Squared Error (RSE) of all the competitors on four real-world datasets for all prediction horizons. We highlight the best results for all the 16 experiments (4 datasets $\times 4$ horizons) in bold face in the table. Among all the experiments, the TPA-based models consistently outperform the baseline models with two exceptions: TPA-based models achieve 14 best results while AR and

Table 2: Prediction errors of Temporal Pattern Attention model and its variants in terms of RSE. Lower RSE is preferred. Bold text indicates the lowest RSE error.

Dataset	**Model**	**3-step**	**6-step**	**12-step**	**24-step**
Solar Energy	AR	0.24321	0.37872	0.58656	0.85142
	VAR	0.20350	0.29649	0.47860	**0.72920**
	TPA-h	0.19999	**0.29586**	0.48533	0.75626
	TPA-v	**0.19898**	0.30423	0.48106	0.78049
	Uni-w/oCNN-h	0.1999	0.3024	0.47913	0.78948
	Uni-w/oCNN-v	0.2014	0.30049	**0.47405**	0.78182
	Bi-TPA-h	0.21058	0.32031	0.4947	0.77706
	Bi-TPA-v	0.21005	0.32126	0.50073	0.78129
	Bi-w/oCNN-h	0.2099	0.31968	0.48786	0.76237
	Bi-w/oCNN-v	0.21076	0.32011	0.49943	0.77555
Traffic	AR	0.58702	0.61196	0.61470	0.62161
	VAR	1.25123	0.93693	0.92429	0.94611
	TPA-h	0.48528	0.50002	0.50795	0.53725
	TPA-v	0.48447	0.53205	0.51381	0.53183
	Uni-w/oCNN-h	0.48417	0.53879	0.50668	0.53684
	Uni-w/oCNN-v	0.508	0.49994	0.51362	0.53787
	Bi-TPA-h	**0.47022**	0.4993	0.4955	0.52747
	Bi-TPA-v	0.47527	0.50287	0.49718	0.51593
	Bi-w/oCNN-h	0.47922	0.50914	0.50374	**0.51287**
	Bi-w/oCNN-v	0.47767	**0.4969**	**0.49367**	0.51686
Electricity	AR	**0.08932**	0.09830	0.10211	0.10387
	VAR	0.67934	0.32611	0.34705	0.28077
	TPA-h	0.09073	0.0952	0.10416	0.102
	TPA-v	0.0919	**0.09485**	0.101	0.10052
	Uni-w/oCNN-h	0.09285	0.09901	0.10001	0.10253
	Uni-w/oCNN-v	0.09012	0.09677	0.10267	0.10114
	Bi-TPA-h	0.09504	0.09718	0.102	**0.09953**
	Bi-TPA-v	0.09057	0.09828	0.10029	0.10318
	Bi-w/oCNN-h	0.09339	0.09666	0.10062	0.10044
	Bi-w/oCNN-v	0.09307	0.09841	**0.09794**	0.10066
Exchange Rate	AR	0.01737	0.02430	0.03402	0.04591
	VAR	0.01848	0.02748	0.04247	0.06732
	TPA-h	0.01836	0.02528	0.03364	0.05921
	TPA-v	0.01813	0.02523	**0.03342**	0.04775
	Uni-w/oCNN-h	0.01776	0.02506	0.03581	0.05151
	Uni-w/oCNN-v	**0.01725**	0.0281	0.03398	0.05276
	Bi-TPA-h	0.01733	0.02413	0.03441	0.04682
	Bi-TPA-v	0.01726	0.02547	0.03474	**0.04394**
	Bi-w/oCNN-h	0.01736	**0.0241**	0.0344	0.04398
	Bi-w/oCNN-v	0.01792	0.02411	0.03384	0.04683

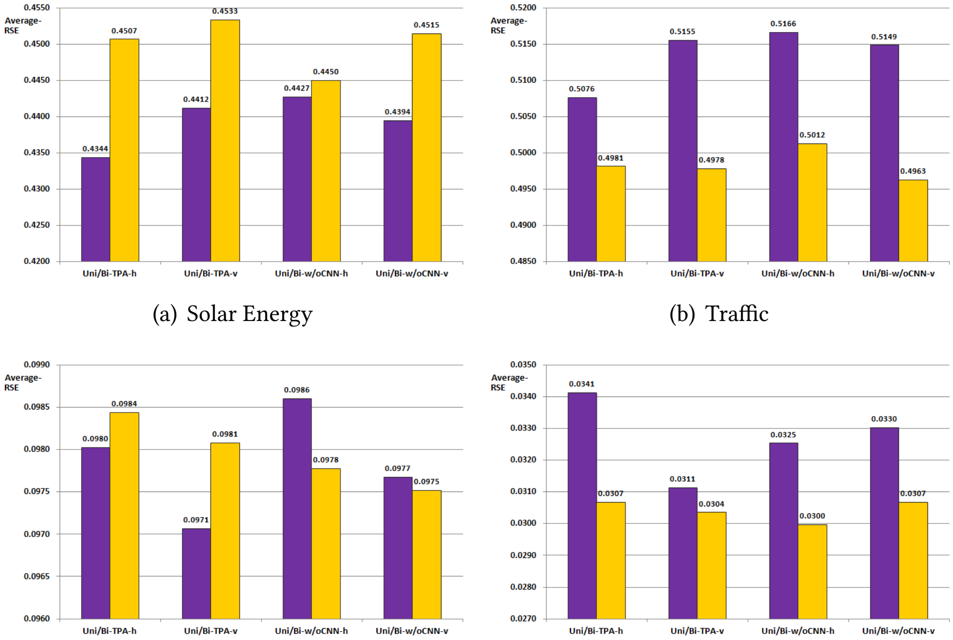
VAR give 2 best results in total. Even though AR and VAR models achieve best accuracy on two experiments, the TPA-based models can still give comparable results. Moreover, the experimental results show that TPA-based models have robust performance on datasets of different sizes with different numbers of variables. To be more specific, TPA-based models can handle datasets of size from 534KB to 172MB, and they also can handle a wide range of number of variables, from 8 to 862. This indicates the superiority of TPA-based models on multivariate time-series. In addition, the original TPA model, namely TPA-h, only achieves one best result, meaning that we can achieve further improvements. We will discuss three possible improvements in following sections, that is, bi-directional LSTM layer, CNN for temporal pattern detection, and two attention mechanisms.

4.3 Effects of Bi-directional LSTM (Q2)

To determine whether bi-directional LSTM could improve the prediction accuracy, we divide the aforementioned eight TPA-based models into four pairs and compare their performance on the four datasets. The only difference between each pair of models is whether the bi-directional LSTM or uni-directional LSTM layer is used. For each model on one dataset, we calculate the averaged RSE of its four prediction horizons as follows:

$$mean(RSE) = \frac{1}{4}(RSE_{3-step} + RSE_{6-step} + RSE_{12-step} + RSE_{24-step})$$

Then we compare the averaged RSEs of each pair of models. In Section 4.4 and Section 4.5, we will also use this averaged RSE for comparison and we will not repeat this later.



(a) Solar Energy

(b) Traffic

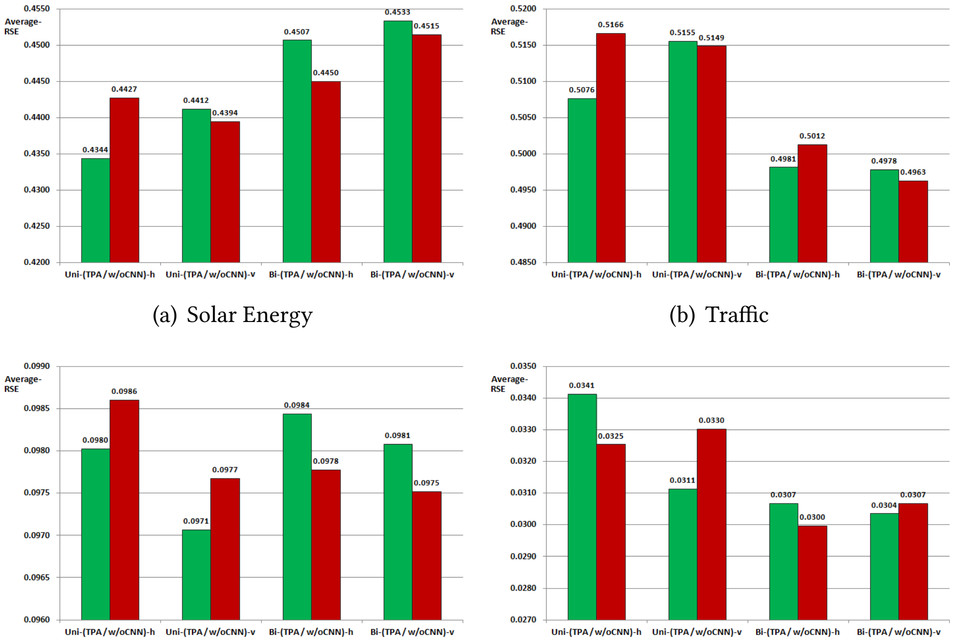
(c) Electricity

(d) Exchange Rate

Figure 6: Effects of uni-directional LSTM vs bi-directional LSTM. In each sub-figure, we compare the averaged RSEs of four pairs of model on a dataset. The purple histograms represent models using uni-directional LSTM and the yellow histograms use bi-directional LSTM. The y axis in each sub-figure is averaged RSE while x axis represent model pairs. Lower histogram means higher accuracy.

Figure 6 shows the effect of bi-directional LSTM layers. The purple histograms in the figure represent models using uni-directional LSTM and the yellow histograms use bi-directional LSTM. For the Traffic and Exchange Rate datasets, the figure clearly shows that using bi-directional LSTM layer indeed improves the prediction accuracy for all horizons. But the experimental results cannot indicate which kind of LSTM layer is more suitable for Electricity dataset as both uni-directional and bi-directional LSTM layers achieve better performance twice. Finally, sub-figure 6(a) shows that the bi-directional LSTM layer cannot improve the performance on Solar Energy dataset at all. We give a possible explanation that bi-directional LSTM layer has too many parameters, which causes over-fitting on the training set, thus gives poor performance on the test set. In summary, we suggest that given a specific dataset, both uni-directional LSTM layer and bi-directional LSTM layer should be tested to determine which one works better on this dataset.

4.4 Effects of CNN for Temporal Pattern Detection (Q3)



(a) Solar Energy

(b) Traffic

(c) Electricity

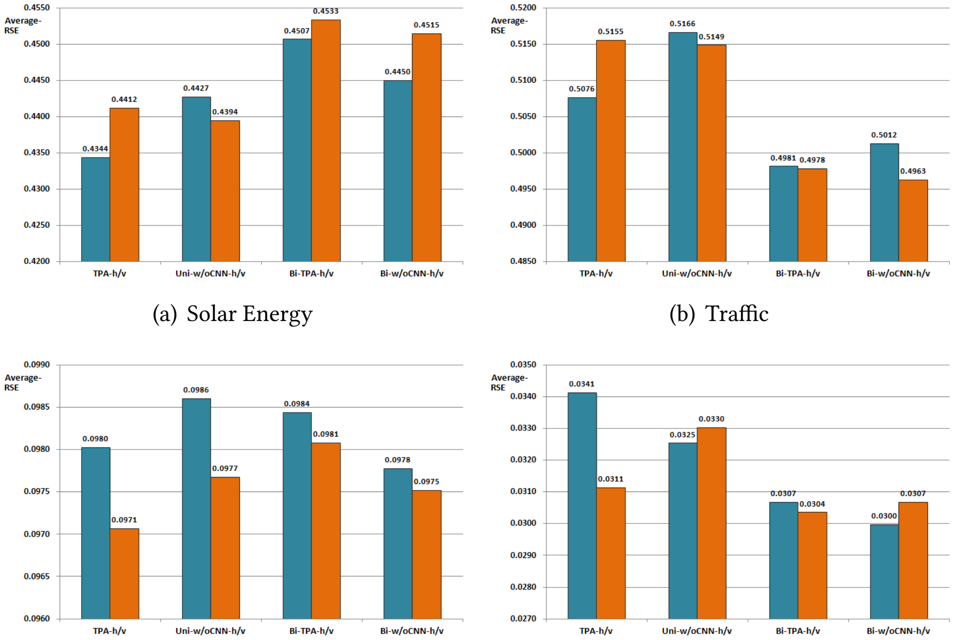
(d) Exchange Rate

Figure 7: Effects of TPA model vs TPA model without CNN for temporal pattern detection. The green histograms represent models using CNN for temporal pattern detection and the red histograms do not use CNN. The y axis in each sub-figure is averaged RSE while x axis represent model pairs. Lower histogram means higher accuracy.

To determine whether the CNN for temporal pattern detection could improve the prediction accuracy, we compare another four pairs of TPA-based models. In each pair of models, the only difference is whether the hidden states will be input through CNN layer for temporal pattern detection. The green histograms represent models using CNN for temporal pattern detection and the red histograms do not use CNN. As shown in Figure 7, TPA-based models with CNN achieve seven better results while TPA-based models without CNN win nine times. The CNN for temporal pattern detection has no obvious superiority on any dataset, which means the CNN layer could not be that effective. Our explanation is that the CNN layer limits the performance of attention mechanism, because a CNN layer computes the weighted sum of neighboring data, which will make the characteristics of each time step (for vertical attention) or variable (for horizontal attention) blurred. Data obscured by CNN makes it harder for attention mechanisms to find useful information, thus limit the model's performance. However, in the TPA-based models without CNN, the attention mechanism is directly applied to the hidden states, where the data is never mixed up with each other. Therefore, the attention mechanism can distinguish the useful information more easily. Finally, we conclude that the CNN for temporal pattern detection should be carefully used because it could possibly limit the effect of attention mechanism.

4.5 Which Attention Direction Is Better (Q4)

To determine which attention direction is more suitable for multivariate time-series prediction tasks, we again divide the eight TPA-based models into four pairs and compare their averaged RSEs. As shown in Figure 8, the blue histograms represent models using horizontal attention mechanism and the orange histograms use



(a) Solar Energy

(b) Traffic

(c) Electricity

(d) Exchange Rate

Figure 8: Effects of horizontal attention vs vertical attention. The blue histograms represent models using horizontal attention mechanism and the orange histograms use vertical attention mechanism. The y axis in each sub-figure is averaged RSE while x axis represent model pairs. Lower histogram means higher accuracy.

vertical attention mechanism. The only difference between each pair of models is whether the horizontal attention or vertical attention is used. From the figures, the horizontal attention mechanism achieve six better performance while vertical attention mechanism wins 10 times. In sub-figure 8(a), horizontal attention outperforms vertical attention for three times, which means horizontal attention is preferred on Solar Energy dataset. However, in sub-figure 8(b) and sub-figure 8(c), vertical attention mechanism achieves most better performance with only one exception. So the vertical attention is more suitable on these two datasets. Finally, in sub-figure 8(d), the horizontal and vertical attention have same performance, both win twice. In conclusion, for a specific dataset, both horizontal and vertical attention mechanisms should be tested to develop a good prediction model.

Chapter 5

Related Works

Many approaches have been proposed to predict the time-series data. Traditional statistical models, such as Autoregressive (AR) model and Autoregression Integrated Moving Average (ARIMA), have been proved to be effective in time-series prediction. Massimiliano Marcellino et al.[5] used an AR model to predict macroeconomic time-series. J. Contreras et al.[6] used an ARIMA model to predict electricity prices, which had good performance. Igor Melnyk et al.[7] used a Vector Autoregressive model to predict multivariate time-series. However, all of the aforementioned models can only model the linear patterns in time-series. None of these models can capture non-linearity in time-series data.

In recent years, machine learning models work well in diverse fields. The machine learning models, such as support vector machine (SVM), random forest (RF), and gradient boosting machines (GBM), achieved fulfilling performance in time-series prediction also. Kyoung-jae Kim et al. and Francis E.H Tay et al.[8, 9] proved the effectiveness of SVM in financial time-series, respectively. A.Lahouar et al.[10] forecast the electrical load using a random forest model. Hristos Tyralis et al.[11] used random forest method to select variables in time-series prediction. Souhaib Ben Taieb et al.[12] proposed a GBM approach to forecast power load. Yanru Zhang et al.[13] improved performance in traffic time-series prediction with a GBM model. Even though the machine learning models show their effectiveness in time-series prediction, they still have shortcomings. Like AR model, the SVM model is a linear approach, which is not able to handle time-series with complex non-linearity. The RF and GBM models can model the non-linearity well, but they fail to handle the temporal dependencies through time steps.

Artificial Neural Networks (ANNs) are another type of popular model in time-series prediction nowadays. Arezou et al.[14] built an autoencoder-based model for time-series prediction. Yi-Shian Lee et al.[15] proposed a model combining ARIMA and neural networks to predict time-series. Among many kinds of ANNs, Recurrent Neural Networks (RNNs)[16, 17, 18] have shown its flexibility in capturing the non-linear patterns. Traditional RNNs, however, suffer from the problem of vanishing gradients[19] and thus have difficulty in capturing long-term dependencies. Recently, Long Short-term Memory (LSTM) neural networks[20] and the Gated Recurrent Unit (GRU)[21] have overcome this limitation and achieved great success in various applications, e.g., neural machine translation[1], speech recognition[22], and image processing[23]. Therefore, it is natural to consider LSTM-based models for time-series prediction. Shuai Zheng et al., Malhotra et al., and Jie Liu et al.[24, 25, 26] have proposed LSTM-based models for time-series prediction and achieved good results in Prognosis and Health Management (PHM) applications. Yang Guo et al.[27] proposed a Convolutional LSTM model, which can predict multi-sensor time-series well. In practice, however, LSTM and GRU cannot memorize very long-term dependencies due to training instability and the limited length context vector[1]. In time-series analysis, this could be a concern since we usually expect to make predictions based on a relatively long time-series. To resolve this issue, Bahdanau et al.[1] proposed an attention-based encoder-decoder network, which employed an attention mechanism to select parts of hidden states across all the time steps. Recently, Yang et al.[28] proposed a hierarchical attention network, which used two layers of attention mechanism to select relevant encoder hidden states of all time steps. Since attention mechanism has achieved great success in Natural Language Processing (NLP), people are curious about how attention-based LSTM models will perform in time-series prediction tasks. Shih et al.[2] proposed a multivariate time-series model combined with attention mechanism and convolutional filters, which achieved state-of-the-art performance in several real-world datasets. Shih's paper piques our interest and motivates us to undertake our study.

Chapter 6

Conclusion

Temporal Pattern Attention (TPA) model is the state-of-the-art model for multivariate time-series prediction tasks, consisting of a recurrent neural network (RNN) layer, a convolutional neural network (CNN) for temporal pattern detection, an attention mechanism, and several fully-connected layers. In this thesis, we study the effects of three main components in the TPA model: the RNN layer, the CNN for temporal pattern detection, and the attention mechanism. To carry out the ablation study, we implement seven variants of Temporal Pattern Attention model and conduct experiments on four real-world multivariate time-series datasets. Our experimental results indicate the effects of these three components: 1) using bi-directional LSTM as RNN layer could improve the model performance on some datasets but degrades the performance on a few other datasets. 2) CNN for temporal pattern detection may limit the performance of attention mechanism because CNN could blur the neighboring data, making it hard for attention mechanism to extract useful information. 3) Both horizontal and vertical attention mechanisms have different performance on different datasets, thus to develop a good prediction model on a specific dataset, both attention mechanisms should be surveyed. Our findings point to the need for further studies on why TPA-based models with bi-directional LSTM perform bad on Solar Energy dataset and modelling the dependencies among multiple variables.

References

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

[2] S.-Y. Shih, F.-K. Sun, and H. yi Lee, “Temporal pattern attention for multivariate time series forecasting,” *ECML PKDD*, 2019.

[3] G. Lai, W. Chang, Y. Yang, and H. Liu, “Modeling long- and short-term temporal patterns with deep neural networks,” 2017.

[4] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, and G. W. Cottrell, “A dual-stage attention-based recurrent neural network for time series prediction,” *arXiv 1704.02971*, 2017.

[5] M. Marcellino, J. H. Stock, and M. W. Watson, “A comparison of direct and iterated multistep ar methods for forecasting macroeconomic time series,” *Journal of Econometrics*, vol. 135, 2006.

[6] J. Contreras, R. Espinola, F. Nogales, and A. Conejo, “Arima models to predict next-day electricity prices,” *IEEE Transactions on Power Systems*, vol. 18, 2003.

[7] I. Melnyk and A. Banerjee, “Estimating Structured Vector Autoregressive Model,” *arXiv:1602.06606*, 2016.

[8] K. jae Kim, “Financial time series forecasting using support vector machines,” *Neurocomputing*, vol. 55, pp. 307–319, 2003.

[9] F. E. Tay and L. Cao, “Application of support vector machines in financial time series forecasting,” *Omega*, vol. 29, 2001.

[10] A. Lahouar and J. B. H. Slama, “Day-ahead load forecast using random forest and expert input selection,” *Energy Conversion and Management*, vol. 103, pp. 1040–1051, 2015.

[11] H. Tyralis and G. Papacharalampous, “Variable selection in time series forecasting using random forests,” *Algorithms*, 2017.

[12] S. B. Taieb and R. Hyndman, “A gradient boosting approach to the kaggle load forecasting competition,” *International Journal of Forecasting*, vol. 30, pp. 382–394, 2014.

[13] Y. Zhang and A. Haghani, “A gradient boosting method to improve travel time prediction,” *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 308–324, 2015.

[14] A. Moussavi-Khalkhali and M. Jamshidi, “Constructing a deep regression model utilizing cascaded sparse autoencoders and stochastic gradient descent,” *IEEE International Conference on Machine Learning and Applications*, 2016.

[15] Y.-S. Lee and L.-I. Tong, “Forecasting time series using a methodology based on autoregressive integrated moving average and genetic programming,” *Knowledge-Based Systems*, vol. 24, 2011.

[16] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by backpropagating errors,” *Nature*, vol. 323(9), pp. 533–536, 1986.

[17] P. J. Werbos, “Backpropagation through time: What it does and how to do it,” *Proceedings of the IEEE*, vol. 78:10, pp. 1550–1560, 1990.

[18] J. L. Elman, “Distributed representations simple recurrent networks and grammatical structure,” *Machine Learning*, vol. 7:2-3, pp. 195–225, 1991.

[19] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *IEEE Transactions on Neural Networks*, vol. 5(2), pp. 157–166, 1994.

[20] H. S. and S. J., “Long short-term memory,” *Neural Computation*, vol. 9.8, 1997.

[21] K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv:1406.1087*, 2014.

[22] A. Graves, A. rahman Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” *ICASSP*, pp. 6645–6649, 2013.

[23] A. Karpathy and F.-F. Li, “Deep visual-semantic alignments for generating image descriptions,” *CVPR*, pp. 3128–3137, 2015.

[24] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, “Long short-term memory network for remaining useful life estimation,” *IEEE International Conference on Prognostics and Health Management (ICPHM)*, 2017.

[25] P. Malhotra, Anand, P. Agarwal, and G. Shroff, “Lstm-based encoder-decoder for multi-sensor anomaly detection,” *International Conference on Machine Learning (ICML)*, 2016.

[26] J. Liu, A. Saxena, K. Goebel, B. Saha, , and W. Wang, “An adaptive recurrent neural network for remaining useful life prediction of lithium-ion batteries,” *Annual Conference of the Prognostics and Health Management Society*, 2010.

[27] Y. Guo, Z. Wu, and Y. Ji, “A hybrid deep representation learning model for time series classification and prediction,” *International Conference on Big Data Computing and Communications (BIGCOM)*, 2017.

[28] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, “Hierarchical attention networks for document classification,” *NAACL*, 2016.

Acknowledgements

First, I would like to express my deepest gratitude to Professor U Kang, my supervisor, for his constant encouragement and guidance. He has walked me through all the stages of my study here in College of Engineering of Seoul National University. Without his enlightening instruction, impressive kindness and patience, I could not have completed my thesis. I gain a lot from Professor U Kang's valuable and professional guidance. I shall also extend my gratitude to Professor Byung-Gon Chun and Professor Gunhee Kim for reviewing my thesis and giving me many insightful and helpful comments. And I would like to thank all the members of Data Mining Lab, who have provided me countless help during my Master's program. Those joyful and painful experiences bind us together and make us stronger.

Last, my thanks would go to my beloved family for their trust and confidence in me all through these years.