Research Article

Robust and Adaptive Online Time Series Prediction with Long Short-Term Memory

Haimin Yang,1 Zhisong Pan,1 and Qing Tao2

1College of Command and Information System, PLA University of Science and Technology, Nanjing, Jiangsu 210007, China
21st Department, Army Officer Academy of PLA, Hefei, Anhui 230031, China

Correspondence should be addressed to Zhisong Pan; hotpzs@hotmail.com

Received 27 August 2017; Revised 23 November 2017; Accepted 3 December 2017; Published 17 December 2017

Academic Editor: Pedro Antonio Gutierrez

Copyright © 2017 Haimin Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Online time series prediction is the mainstream method in a wide range of fields, ranging from speech analysis and noise cancelation to stock market analysis. However, the data often contains many outliers with the increasing length of time series in real world. These outliers can mislead the learned model if treated as normal points in the process of prediction. To address this issue, in this paper, we propose a robust and adaptive online gradient learning method, RoAdam (Robust Adam), for long short-term memory (LSTM) to predict time series with outliers. This method tunes the learning rate of the stochastic gradient algorithm adaptively in the process of prediction, which reduces the adverse effect of outliers. It tracks the relative prediction error of the loss function with a weighted average through modifying Adam, a popular stochastic gradient method algorithm for training deep neural networks. In our algorithm, the large value of the relative prediction error corresponds to a small learning rate, and vice versa. The experiments on both synthetic data and real time series show that our method achieves better performance compared to the existing methods based on LSTM.

1. Introduction

A time series is a sequence of real-valued signals that are measured at successive time intervals [1, 2]. Time series data occur naturally in many application areas such as economics, finance, environment, and medicine and often arrives in the form of streaming in many real-world systems. Time series prediction has been successfully used in a wide range of domains including speech analysis [3], noise cancelation [4], and stock market analysis [5, 6]. The traditional methods of time series prediction commonly use a potential model, for example, autoregressive moving average (ARMA) [7], autoregressive integrated moving average (ARIMA) [1], and vector autoregressive moving average (VARMA) [8], to mimic the data. However, these methods all need to deal with the whole dataset to identify the parameters of the model when facing new coming data, which is not suitable for large datasets and online time series prediction. To address this problem, online learning methods are explored to extract the underlying pattern representations from time series data in a sequential manner. Compared to traditional batch learning methods, online learning methods avoid expensive retraining cost when handling new coming data. Due to the efficiency and scalability, online learning methods including methods based on linear models [9], ensemble learning [10], and kernels [11] have been applied to time series prediction successfully.

Long short-term memory (LSTM) [12], a class of recurrent neural networks (RNNs) [13], is particularly designed for sequential data. LSTM has shown promising results for time series prediction. Its units consist of three gates: input gate, forget gate, and output gate. It is popular due to the ability of learning hidden long-term sequential dependencies, which actually helps in learning the underlying representations of time series. However, the time series data in real world often contains some outliers more or less especially in cyberattacks, which are commonly shown as anomalies in time series data monitoring some measurements of network traffic. Those outliers mislead the learning method in extracting the true representations of time series and reduce the performance of prediction.

In this paper, we propose an efficient online gradient learning method, which we call RoAdam (Robust Adam)
for LSTM to predict time series in the presence of outliers. The method modifies Adam (Adaptive Moment Estimation) [14], a popular algorithm for training deep neural networks through tracking the relative prediction error of the loss function with a weighted average. Adam is based on standard stochastic gradient descent (SGD) method without considering the adverse effect of outliers. The learning rate of RoAdam is tuned adaptively according to the relative prediction error of the loss function. The large relative prediction error leads to a smaller effective learning rate. Likewise, a small error leads to a larger effective learning rate. The experiments show that our algorithm achieves the state-of-the-art performance in the presence of outliers to model the normal behavior in advance. In this paper, our online learning method for time series prediction is robust to outliers through adaptively tuning the learning rate of the stochastic gradient method to train LSTM.

3. Preliminaries and Model

In this section, we formulate our problem to be resolved and introduce some knowledge about Adam, a popular algorithm for training LSTM.

3.1. Online Time Series Prediction with LSTM

In the process of online time series prediction, the desirable model learns useful information from \{x_1, x_2, ..., x_t\} to give a prediction \(\hat{x}_t\) and then compare \(\hat{x}_t\) with \(x_t\) to update itself, where \{x_1, x_2, ..., x_t\} is a time series, \(\hat{x}_t\) is the time series data point forecasted at time \(t\), and \(x_t\) is the real value. LSTM is suitable for discovering dependence relationships between the time series data by using specialized gating and memory mechanisms.

We give the formal definition of a neuron of a LSTM layer as follows. The \(j\)th neuron of a LSTM layer at time \(t\), \(c^j_t\) consists of input gate \(i^j_t\), forget gate \(f^j_t\), and output gate \(o^j_t\), and is updated through forgetting the partially existing memory and adding a new memory content \(c^j_t\). The expressions of \(i^j_t, f^j_t, o^j_t\), and \(c^j_t\) are shown as follows:

\[
i^j_t = \sigma(W_i x_t + U_i h_{t-1} + V_i c^j_{t-1}),
\]

\[
f^j_t = \sigma(W_f x_t + U_f h_{t-1} + V_f c^j_{t-1}),
\]

\[
o^j_t = \sigma(W_o x_t + U_o h_{t-1} + V_o c^j_{t-1}),
\]

\[
c^j_t = f^j_t c^j_{t-1} + i^j_t \tilde{c}^j_t.
\]

Note that \(W_i, W_f, W_o, U_i, U_f, U_o\), and \(V_o\) are the parameters of the \(j\)th neuron of a LSTM layer at time \(t\). \(\sigma\) is a logistic sigmoid function. \(V_i, V_f, V_o\) are diagonal matrices. \(h_{t-1}\) and \(c_{t-1}\) are the vectorization of \(h^j_{t-1}\) and \(c^j_{t-1}\). The output \(h^j_t\) of this neuron at time \(t\) is expressed as:

\[
h^j_t = o^j_t \tanh \left(c^j_t \right).
\]

In our model of online time series prediction, we set a dense layer to map the outputs to the target prediction, which is formulated as

\[
y = g \left(W_d h_t + b_d \right),
\]

where \(g(\cdot)\) is the activation function of the dense layer, \(W_d\) is the weights, \(b_d\) is the bias, and \(h_t\) is the vectorization of \(h^j_t\). The objection of our model at time \(t\) is to update the parameters \(W_t = \{W_i, W_f, W_o, U_i, U_f, U_o, V_i, V_f, V_o, b_d\}. The standard process is

\[
W_{t+1} = W_t - \eta \nabla L(x_t, \hat{x}_t),
\]

where \(\eta\) is the learning rate and \(L(x_t, \hat{x}_t)\) is the loss function.
RoAdam. Parameters carried over from Adam have the same default values: $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. For parameters specific to our method, we recommend default values $\beta_1 = 0.999, k = 0.1, K = 10$.

Require: $\eta$: learning rate
Require: $\beta_1, \beta_2 \in (0, 1)$: exponential decay rates for moment estimation in Adam
Require: $\beta_1 \in (0, 1)$: exponential decay rate for computing relative prediction error
Require: $k, K$: lower and upper threshold for relative prediction error
Require: $\epsilon$: fuzz factor
Require: $l(W)$: loss function

Require: $W_0$: initial value for parameters

$m_0 = v_0 = 0$
$d_0 = 1$
$l(W_0) = l(W_{-1}) = 1$
$t = 0$

while stopping condition is not reached do

$g_t = \nabla_{w_t} l(W_{t-1})$
$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$
$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$
$
\hat{m}_t = m_t / (1 - \beta_1^t)$
$
\hat{v}_t = v_t / (1 - \beta_2^t)$

if $l(W_{t-1})/l(W_{t-2}) \geq K$ then

$r_t = \min\{\max\{k, l(W_{t-1})/l(W_{t-2})\}, K\}$

else

$r_t = \min\{\max\{1/K, l(W_{t-1})/l(W_{t-2})\}, 1/k\}$

end if

$d_t = \beta_3 d_{t-1} + (1 - \beta_3) r_t$

$W_{t+1} = W_t - \eta \hat{m}_t / (d_t \sqrt{\hat{v}_t} + \epsilon)$

$t = t + 1$

end while

return $W_t$

Algorithm 1

3.2. Adam. Adam is a method for efficient stochastic optimization, which is often used to train LSTM. It computes adaptive learning rates for individual parameters from estimates of the first moment and the second moment of the gradients, only requiring first-order gradients. Adam keeps an exponentially decaying average of the gradient and the squared gradient:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t,$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2,$$

where $m_t$ and $v_t$ initialized as zero are estimates of the first moment and the second moment and $\beta_1$ and $\beta_2$ are exponential decay rates for the moment estimates. We can find that $m_t$ and $v_t$ are biased towards zero, when $\beta_1$ and $\beta_2$ close to 1. So Adam counteracts these biases through bias correction of $m_t$ and $v_t$:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t},$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}.$$

The rule of updating parameters is

$$W_{t+1} = W_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t,$$

where $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$ by default.

4. Method

In this section, we introduce our online gradient learning method, which is called RoAdam (Robust Adam) to train long short-term memory (LSTM) for time series prediction in the presence of outliers. Our method does not directly detect the outliers and adaptively tunes the learning rate when facing a suspicious outlier.

In Algorithm 1, we provide the details of the RoAdam algorithm. The main difference between our algorithm and Adam is $r_t$, a relative prediction error term of the loss function. The relative prediction error term indicates whether the point is an outlier. The larger value of $r_t$ means the current point is more suspicious to be an outlier. It is computed as $r_t = l(W_{t-1})/l(W_{t-2})$, where $l(W_{-1}) = l(x_t, \bar{x}_t)$ and $l(W_{-2}) = l(x_{t-1}, \bar{x}_{t-1}, x_{t-2}, \bar{x}_{t-2})$. $l(x_t, \bar{x}_t)$ and $l(x_{t-1}, \bar{x}_{t-1})$ are the absolute prediction errors of $x_t$ and $x_{t-1}$. In practice, a threshold is used to scheme to ensure the stability of relative prediction error term, $k$ and $K$ denote the lower and upper thresholds for $r_t$. We let $r_t = \min\{\max\{k, l(W_{t-1})/l(W_{t-2})\}\}, K\}$ (1), if $l(W_{t-1})/l(W_{t-2}) \geq K$ and $r_t = \min\{\max\{1/K, l(W_{t-1})/l(W_{t-2})\}\}, 1/k\}$ (2) otherwise, which captures both increase and decrease of relative prediction.
error. Our settings consider different situations when the preceding point $x_{t-1}$ and current point $x_t$ are at different status. The details are listed in Table 1.

To get a smoother estimate, we compute the relative prediction error with a weighted average. The final result $d_t$ is $\beta_3 d_{t-1} + (1 - \beta_3)r_t$. Here the effect of $\beta_3$ is the same as $\beta_1$ and $\beta_2$ in Adam. In general, RoAdam is modified in the basis of Adam through multiplying the denominator $\sqrt{V}$ with $d_t$. The large value of $d_t$ corresponds to a small learning rate, and vice versa.

### 5. Experiment

In this section, we illustrate the performance of our proposed algorithm RoAdam compared to RLSTM, SR-LSTM, and RN-LSTM on both synthetic data and real time series.

#### 5.1. Experiment Setup

RLSTM means real time LSTM, which updates the model using the newly coming data without considering the effect of outliers. SR-LSTM stands for LSTM with suspicious point removal. The difference between SR-LSTM and RN-LSTM is that once a suspicious point is
detected as an outlier, SR-LSTM does not update on this point and RN-LSTM updates using a recent normal point. They both use the method proposed in [25] to detect the outlier. In addition, all the algorithms use the same LSTM model besides the optimizer. RLSTM, SR-LSTM, and RN-LSTM adopt the original Adam optimizer. The LSTM model has 3 layers and the number of neurons in each layer is 400. The mean squared error is chosen as the loss function and the L2 regularization with 0.0001 penalty is used. The parameters of RoAdam carried from Adam have the same default values: $\eta = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. For parameters specific to our method, we try different values and recommend default values $\beta_3 = 0.999$, $k = 0.1$, and $K = 10$.

5.2. Data Sets. To examine the prediction performance, we evaluate all the previous algorithms on synthetic data and real time series.

5.2.1. Synthetic Data. The synthetic data is sampled from a Gaussian distribution with the corresponding mean $u \in [0, 100]$ and variance $\sigma \in [10, 30]$ plus the trend component $T \in [-0.5, 0.5]$. The length $l$ is 2,500. The outliers are injected based on a Bernoulli distribution identified by $\alpha = 0.01$ and $l \cdot \alpha$ is the expected number of outliers. The values of outliers are also sampled from a Gaussian distribution with mean $u \in [0, 1000]$ and variance $\sigma \in [10, 30]$. The expression of $x_i$ is

$$x_i = \begin{cases} 
  x + T, x \sim N(u, \sigma), u \in [0, 100], \sigma \in [10, 30], T \in [-0.5, 0.5], & \text{when } x_i \text{ is a normal point;} \\
  x, x \sim N(u, \sigma), u \in [0, 1000], \sigma \in [10, 30], & \text{when } x_i \text{ is an outlier.} 
\end{cases}$$

(8)
5.2.2. Real Time Series. The first time series data is ECG data, which consists of 70 series of 1000 ECG measurements [26]. We choose 100 samples from ECG data set. The second one is HandOutlines, which is from the commonly used UCR (http://www.cs.ucr.edu/~eamonn/time_series_data/). The last time series data is daily index of Dow Jones Industrial Average (DJIA) during years 1885–1962. We randomly select 1% of each real time series as outliers, whose values are 2 or 3 times bigger than the true ones. Figure 1 presents the true value of synthetic data and real time series. The x-axis is time (the number of samples) and the y-axis is true value.

5.3. Experimental Results. In this section, we test RMSE of the algorithms mentioned above to examine the effectiveness and efficiency.

\[ \text{RMSE} = \frac{1}{T} \sum_{t=1}^{T} (X_t - \bar{X}_t)^2. \]  

(9)

RMSE allows us to compare errors with the number of samples increasing. In addition, we average the results over 100 runs for stability.

Table 2 shows the RMSE of different algorithms both on synthetic data and real time series. We can find that RoAdam outperforms all the other algorithms on RMSE. Figures 2–5 visualize the prediction value of all the algorithms on synthetic data and real time series. The x-axis is time (the number of samples) and the y-axis is prediction value. We can observe that the prediction value produced by R-LSTM has oscillations around outliers. It indicates that the prediction performance of R-LSTM is indeed affected by outliers. Although SR-LSTM, RN-LSTM, and RoAdam have almost the same shape of prediction value, RoAdam has the least RMSE. The reason may be that SR-LSTM and RN-LSTM may lose some information of the normal points when they are mistaken outliers.
Table 2: RMSE on synthetic data and real time series.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Synthetic</th>
<th>ECG</th>
<th>HandOutlines</th>
<th>DJIA</th>
</tr>
</thead>
<tbody>
<tr>
<td>RLSTM</td>
<td>0.7606</td>
<td>0.8505</td>
<td>0.9756</td>
<td>1.8454</td>
</tr>
<tr>
<td>SR-LSTM</td>
<td>0.7329</td>
<td>0.8323</td>
<td>0.9411</td>
<td>1.7574</td>
</tr>
<tr>
<td>RN-LSTM</td>
<td>0.7218</td>
<td>0.8217</td>
<td>0.9376</td>
<td>1.6218</td>
</tr>
<tr>
<td>RoAdam</td>
<td><strong>0.4946</strong></td>
<td><strong>0.5626</strong></td>
<td><strong>0.7633</strong></td>
<td><strong>1.3875</strong></td>
</tr>
</tbody>
</table>

6. Conclusions

In this paper, we propose an efficient online gradient learning method, RoAdam for LSTM, to predict time series, which is robust to outliers. RoAdam is modified on the basis of Adam, a popular stochastic gradient algorithm for training deep neural networks. Through tracking the relative prediction error of the loss function with a weighted average, this method adaptively tunes the learning rate of the stochastic gradient method in the presence of outliers. In the process of prediction, the large value of the relative prediction error corresponds to a small learning rate, and vice versa. The experiments on both synthetic data and real time series show that our method achieves less prediction error compared to the existing methods based on LSTM.

It remains for future work to study whether our approach could be extended to time series prediction with missing data.
Conflicts of Interest

The authors declare no conflicts of interest.

Authors’ Contributions

Haimin Yang participated in the draft writing and experiments. Zhisong Pan and Qing Tao participated in the design of algorithms and commented on the manuscript.

Acknowledgments

Our work is supported by the National Natural Science Foundation of China (nos. 61473149 and 61673394).

References


