2016 IEEE International Conference on Data Science and Advanced Analytics

Robust Online Time Series Prediction with Recurrent Neural Networks

Tian Guo*, Zhao Xu†, Xin Yao‡, Haifeng Chen §, Karl Aberer* and Koichi Funaya†
*EPFL, Lausanne, Switzerland
{tian.guo, karl.aberer }@epfl.ch
†NEC Laboratories Europe, Heidelberg, Germany
{zhao.xu, koichi.funaya}@neclab.eu
‡The University of Birmingham, Birmingham, United Kingdom
x.yao@cs.bham.ac.uk
§NEC Laboratories America, Princeton, USA
haifeng@nec-labs.com

***Abstract*—Time series forecasting for streaming data plays an important role in many real applications, ranging from IoT systems, cyber-networks, to industrial systems and healthcare. However the real data is often complicated with anomalies and change points, which can lead the learned models deviating from the underlying patterns of the time series, especially in the context of online learning mode. In this paper we present an adaptive gradient learning method for recurrent neural networks (RNN) to forecast streaming time series in the presence of anomalies and change points. We explore the local features of time series to automatically weight the gradients of the loss of the newly available observations with distributional properties of the data in real time. We perform extensive experimental analysis on both synthetic and real datasets to evaluate the performance of the proposed method.**

I. Introduction

Time series forecasting [4], [9], [32] is an important task in machine learning with a variety of applications, such as cyber-network traffic prediction, web user access estimation, and pedestrian flow prediction. Forecasting a time series is useful in e.g. reliability monitoring, predictive maintenance and intrusion detection, and can effectively improve availability, security, and the overall service experience. On the other hand, with the rapid growth of various tracking and monitoring technologies, time series data in many real-world systems arrives in a streaming fashion. Online learning methods are expected to extract the underlying patterns from the observed time series for real-time learning and prediction.

Due to its practical importance and technical challenges, some online forecasting methods have been investigated in the literature, including linear methods [2], [25], ensemble methods [31], kernel based methods [33], and Gaussian processes [43]. Recurrent neural networks (RNNs), a class of deep learning methods particularly designed to model sequential data, currently receive increasing attention due to the capacity on learning insightful representations of sequences [16], [23], [37], and have been successfully applied to time series forecasting [7], [30], [42]. However in real-world applications, the time series is usually contaminated by anomalies or outliers that are abrupt observations deviating from the behaviour of the majority. A typical example might be cyber-attacks, which are often shown as anomalies in time series monitoring some measurements of network traffic. In addition, the underlying patterns of time series generally keep changing over time. For example, a newly released online service or functionality of a website can change the probabilistic distribution of time series of user access traffic in terms of mean, variance, correlation and period. With these challenges, the learning methods are expected to capture the true pattern and trend of time series, namely being robust against outliers, and enabling to quickly fit new patterns with potential change points.

In this paper, we present an adaptive gradient learning method for long short-term memory (LSTM) recurrent networks [15], [16], [19], to make the streaming time series forecasting robust to outliers and change points. We model time series with LSTM networks, and use a stochastic gradient descent (SGD) based method to learn the model from the streaming time series. As novel observations arrive, the model parameters are updated in an online mode according to the gradients of the loss of the newly available data. With the standard SGD method, outlier observations will make the updated model deviate from the normal patterns and produce oscillated incorrect predictions until the adverse effect is gradually corrected by the following normal observations. To solve the problem, we explore the local features of time series to weight the gradients of the learning method with distributional properties of the local data. In particular, if a newly available observation is a potential outlier behaving differently from the regular patterns, the corresponding gradient will be downweighted, so as to avoid leading the online model to abruptly drift from the underlying normal patterns. Notice that the newly observed unusual point can also be a change point. Although it behaves differently from the majority like an anomaly, it indicates emerging patterns of time series required to learn quickly. In this case, the corresponding gradient will be retained by a high weight and lead the model to fit the new data in real time. Technically we introduce a weight function based on distributional characteristics of the local data to adapt the gradients of the online SGD to the complicated

978-1-5090-5206-6/16 $31.00 © 2016 IEEE
DOI 10.1109/DSAA.2016.92

streaming time series automatically. The weight function is composed of two components: suspicion ratio and difference value drift formulating the statistical properties of the local data, which make the learning procedure robust to outliers and change points. The experimental analysis on synthetic and real datasets demonstrates the performance of the proposed learning method for streaming time series forecasting in the presence of anomalies and change points.

The rest of the paper is organized as follows. Section II presents related work, while Section III defines the problem to be solved and introduces the notations. In Section IV, we present the proposed robust online learning method for RNN. Section V demonstrates the performance of our method on synthetic and real datasets. Finally, the paper is concluded in Section VI.

II. Related Work

A. Modeling Time Series in the Presence of Outliers and Change Points

Many of the existing machine learning models for time series analysis focus exclusively on either change point or outlier detection. Our paper aims to provide a recurrent neural network based approach to handle both change points and outliers simultaneously for online learning of time series.

[1] introduced the Bayesian change point detection framework and [35] extended it by using the Gaussian process as the underlying predictive model. In [8] a Gaussian process based nonparametric time series prediction model was proposed specifically for periodic time series. [27] detected changes by utilizing relatively density ratio estimation in a batch mode, i.e., they require the entire time series to be present. [44] employed an online discounting learning algorithm to incrementally learn mixture and regression models. [22], [39] put forward scalable anomaly/outlier detection frameworks integrating various predictive models. [13] and [18] provided detailed surveys on change and outlier detection.

Another line of research is to build robust models over noisy and non-stationary time series. [14] proposed a new sequential algorithm for making robust predictions based on Gaussian Process in the presence of change points. [3] extended traditional autoregressive moving average (ARMA) models to the case that less strict assumptions are on the noise term for the purpose of online time-series modeling. [34] developed time-dependent loss function to include the information about the distribution change in time series directly in the SVM learning process.

B. Recurrent Neural Networks for Time Series Analysis

Recurrent neural networks have recently shown promising results in a variety of applications, especially when there exist sequential dependencies in data [11], [28], [38]. [12] proposed a robust learning algorithm for recurrent neural networks on time series. This algorithm is based on filtering outliers from the data first and then training the neural network on the filtered data. Such a filtering-then-learning scheme focuses on dealing with outliers and is not suitable for online training and prediction.

Long short-term memory (LSTM) [19], [28], [41], 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 and to allow online training of sequence data of indefinite duration. Some recent work [10], [24], [29] explored the performance of LSTM in time series modeling. [24] used LSTMs to recognize patterns in multivariate time series, especially multi-label classification of diagnoses. [10], [29] evaluated the ability of LSTMs to detect anomalies in ECG time series. In this paper we present an LSTM online learning method for time series forecasting in presence of both outliers and change points.

III. Problem Formulation

In this section, we discuss the properties and challenges of time series containing outliers and change points, and formulate the problem to be resolved.

Time series, a sequence of data points consisting of successive measurements made over time, often arises when monitoring dynamic processes in a variety of applications, e.g., Internet of Things, sensor networks, and mobile computing. We denote a univariate time series by $\{x_1, \ldots, x_T\}$, where the subscript represents the time instant and each data point x_t is a real value. In this paper, we focus on univariate time series. The proposed method can be naturally generalized to multivariate time series as well.

Since many time series are generated from observations and measurements of physical entities and events, the data is inevitably anomaly-contaminated and non-stationary in the sense that it contains both outliers and change points. An *outlier* in time series is a data point, which is significantly different from the behaviour of the majority of the time-series. A *change point* indicates that the behaviour and hidden data distribution of the time-series are significantly different before and after this point. For instance, a time series might undergo a sudden shift in its mean at a certain time. The major difference between change points and outliers is that change points correspond to more sustained, long-term changes in time series compared to volatile and abrupt outliers. The presence of outliers and change points can adversely affect the time series analysis and complicate the learning process [13], [18], [22], [39].

In many cases, time series are continuously observed, and often need to be analyzed in real time. The online learning methods are thus required, which learn and update models incrementally as new data arrives, in order to better capture the timely trend and the underlying patterns in the time series [3], [13], [20].

In this paper we focus on online time series prediction in the presence of both change points and outliers. The work aims to incrementally learn the time series and provide robust predictions in real time. Given the observed time series of

length $t-1$, the predicted new data point $\hat{x}_t$ at the next time t can be computed as:

$$\hat{x}_t = g(x_{t-1}, x_{t-2}, \ldots, x_1; \theta^*_{t-1}) \quad (1)$$

$g(\cdot)$ is a continuous function which captures the dynamic patterns of the time series. In this paper $g(\cdot)$ is modeled according to the recurrent neural network that can approximate any function with sufficient hidden layers and nonlinear units to arbitrary precision. θ^*_{t-1} denotes the parameters of the recurrent network learned with the $t-1$ sequential observations.

As the new data point x_t arrives, we update the model by minimizing the loss function, e.g., the squared error:

$$\theta^*_t = \arg\max_{\theta_t}(x_t - g(x_{t-1}, x_{t-2}, \ldots, x_1; \theta_t))^2 \quad (2)$$

Recursivly, the new parameters θ^*_t will be used to predict the next data $\hat{x}_{t+1}$. The online learning process is to continuously minimize the loss function over the newly available data.

We notice that, in the learning process, an outlier at time t can lead the model to deviate from the normal patterns. Given the current model θ^*_{t-1}, the loss of the outlier will be high. For reducing the loss (2), θ^*_t will deviate from the normal ones to best fit the outlier, and produces oscillated predictions untile the adverse effect of the outlier is gradually corrected by the following observations [12], [18], [44]. On the other hand, since a change point often leads to a shift in data distribution and makes the current model unfit to the new data, high loss is also expected [14], [22]. In this case, the high loss will take positive effect and lead the model to quickly fit to the new patterns.

To meet these challenges, the online learning method is supposed to react to outliers and change points differently. On the first hand, the online learning model should be robust to outliers by mitigating the model deviation caused by outliers. On the other hand, it should be able to promptly adapt to the new changed data by updating the model using the latest data, so as to provide timely and precise predictions.

TABLE I: Notations

Symbols	Meaning
x_t	the data point at time t of the time series
$\hat{x}_t$	the predicted value of x_t by the neural network
p_t	p-value of x_t
α	significance level
w	the size of the sliding window for feature extraction
h_t	the output of a LSTM layer at time t
o_t	the output gate of a LSTM layer at time t
c_t	the memory unit of a LSTM layer at time t
f^j_t	the forget gate of a LSTM neuron j at time t
i^j_t	the input gate of a LSTM neuron j at time t
W_t	the set of parameters of all layers in the neural network at time t
β_t	gradient weight for updating the model at time t
d_t	difference value drift
s_t	suspicious ratio

IV. Weighted Gradient Online Learning for LSTM Neural Networks

In this section, we present the proposed online learning method for Long Short Term Memory (LSTM) neural networks, a widely used variant of Recurrent Neural Networks (RNN). The method is capable of performing robust predictions over time series in the presence of both outliers and change points.

A. Online learning of time series

In the online process at each time instant, e.g. t, a desirable model should be able to automatically retain useful past information in $\{x_1, \ldots, x_{t-1}\}$, discover dependences for making prediction $\hat{x}_t$ and then update itself using x_t. Recurrent neural networks (RNNs) are rising as a powerful tool to model sequence data [16], [20]. They are suitable for dealing with sequential problems due to the internal hidden states and short term memory realized by recurrent feedback connections.

Among the variants of RNNs, it is well established that the Long Short-Term Memory (LSTM) based networks work well on sequence-based tasks with long-term dependencies [19], [20] by using specialized gates and memory cells in the neuron structure. The LSTM achieves state-of-the-art results for problems spanning natural language processing, image captioning, handwriting recognition, and genomic analysis [20], [24], [26], [38].

We present the formal definition of a neuron of a LSTM layer as follows [11], [19]. Each j-th neuron in the LSTM layer maintains a memory c^j_t at time t. The output h^j_t or the activation of this neuron is then expressed as:

$$h^j_t = o^j_t tanh(c^j_t) \quad (3)$$

where o^j_t is an output gate that modulates the amount of memory content exposure. The output gate is calculated by

$$o^j_t = \sigma(W_o x_t + U_o h_{t-1} + V_o c_t)^j \quad (4)$$

where σ is a logistic sigmoid function. h_{t-1} and c_t are respectively the vectorization of h^j_{t-1} and c^j_t, i.e. $h_{t-1} = \{h^j_{t-1}\}$ and $c_t = \{c^j_t\}$. V_o is a diagonal matrix. Then, the memory cell c^j_t is updated through partially forgetting the existing memory and adding a new memory content $\tilde{c}^j_t$:

$$c^j_t = f^j_t c^j_{t-1} + i^j_t \tilde{c}^j_t \quad (5)$$

where the new memory content at time t is expressed as:

$$\tilde{c}^j_t = tanh(W_c x_t + U_c h_{t-1})^j \quad (6)$$

The extent to which the existing memory (i.e., at time $t-1$) is forgotten is modulated by a forget gate f^j_t, and the degree to which the new memory content (i.e., at time t) is added to the memory cell is modulated by an input gate i^j_t. Then, such gates are computed by

$$f^j_t = \sigma(W_f x_t + U_f h_{t-1} + V_f c_{t-1})^j \quad (7)$$

$$i^j_t = \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1})^j \quad (8)$$

Note that V_f and V_i are diagonal matrices.

Unlike the traditional recurrent neuron which overwrites its content at each time instant [11], [28], a LSTM neuron is able to decide whether to keep the existing memory via the introduced gates. Intuitively, if it detects an important feature from an input sequence at early stage, it easily carries this information (the existence of the feature) over a long period, thereby capturing the hidden long-term dependencies [19], [28].

In our model for online learning of time series, a dense layer is built to connect the LSTM layer so as to map the outputs of LSTM neurons to the target prediction. Denote by W_d and b_d the weights and biases of the dense layer. The output of the neural network is formulated as:

$$y = g(W_d h_t + b_d) \tag{9}$$

where $g(\cdot)$ is the activation function of the dense layer, e.g., $tanh$, $sigmoid$, $linear$, etc. h_t is the vector of the outputs of LSTM neurons, i.e., $h_t = \{h_t^j\}$. Online learning of a neural network is to continuously update the set of parameters in each layer, i.e., $W_t = \langle W_o, U_o, V_o, W_c, U_c, W_f, U_f, W_i, U_i, W_d, b_d \rangle$ at each time instant to minimize a loss function. Then, the prediction at time t of the time series by the LSTM neural network can be expressed as:

$$\hat{x}_t = g(x_{t-1}, \ldots, x_1; W_{t-1}) \tag{10}$$

LSTM neural network employs gradient decent based iterative approaches to update the network parameters at each time instant [19], [28], [41]. The general update process can be expressed as:

$$W_t = W_{t-1} - \eta \bigtriangledown E_t(W_{t-1}) \tag{11}$$

where W_t and W_{t-1} represent the set of parameters W at different time instants. η is the learning rate, $E_t(W_{t-1})$ is the loss function, i.e. squared-error function, and $\bigtriangledown E_t(W_{t-1})$ is the gradient of the loss function at W_{t-1}. There are many variants of gradient descent based methods for updating the network [21], [36], [45],

Since online learning aims to continuously update the model using new arrived data, the loss function is specified as the squared-error of one training instance without accumulating the errors [20], [41]:

$$E_t(W_{t-1}) = e_t^2 = (x_t - \hat{x}_t)^2 \tag{12}$$

Our proposed framework below could be extended to situations where data comes on mini-batches of data points as well.

Then, the gradient is derived as:

$$\bigtriangledown E_t(W_{t-1}) = -2(x_t - \hat{x}_t) \bigtriangledown g(x_{t-1}, \ldots, x_1; W_{t-1}) \tag{13}$$

where the detailed derivation of $\bigtriangledown g(x_{t-1}, \ldots, x_1; W_{t-1})$ is referred to [19], [41]. Such a gradient descent scheme to incrementally update parameters enables the LSTM neural network to naturally evolve with time series in the error-driven way and to adapt to changing time series without explicit detection of change points [13].

B. Effect of Outliers and Change Points on LSTM neural networks

In this part, we first define two basic concepts. Based on statistical time series modeling theory [4], [10], [12], [40], the set of errors $\{e_t\}$ can be fitted to a parametric distribution, referred to as the error reference distribution and maintained online by moving average style methods [5], [8], [40]. During our experiments, we observe that it is reasonable to fit the errors to a Gaussian distribution, though alternative distributions can be chosen based on applications. The associated mean and variance are respectively denoted by μ_t and v_t.

Definition 4.1 (p-value): The p-value of data point x_t, denoted by p_t, is defined as the probability of obtaining a value less or equal to e_t under the error reference distribution $\mathcal{N}(\mu_t, v_t)$, where $e_t = x_t - \hat{x}_t$. Formally, $p_t = \Phi(\frac{e_t - \mu_t}{\sqrt{v_t}})$, where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution.

For the predictive model based time series analysis [8], [10], [12], [18], p-value is a fundamental indicator for outliers or changes. Then, we further define:

Definition 4.2 (Suspicious and Normal Points): In the online learning process of LSTM neural network over time series, if the p-value of a data point, e.g., x_t, is lower than α or higher than $1 - \alpha$ ($\alpha \in (0, 1)$), x_t is a suspicious data point indicating that it could be an outlier or a change point. Otherwise, we call x_t as a normal point. α is the confidence level.

Straightforward solutions help little:

Given the neural network learnt over time series, outliers could lead to dramatical large prediction errors. Consequently using such outliers to update the neural network could drag away the gradient from the previous direction and derail the so-far trained neural network, thereby incurring unreliable prediction [12].

For instance, we use an example experiment to illustrate the effect of outliers on the LSTM neural network. We perform the conventional online learning of LSTM neural network [41] on a real time series from Yahoo S5 dataset[1]. We adopt the interleaved test-then-train framework [13], which means that at each time instant a prediction for the next data point is performed using the current neural network and then the network is updated when the next data point arrives. The experiment is run at the learning rate (0.005), which achieves the least prediction error (i.e., RMSE). For the details of the experiment setup, refer to Section V.

We can observe in Figure 1(*b*) that outliers incur oscillated prediction around them. One possible solution might be to skip the neural network update [12] when a new coming data point is considered as a suspicious point by Definition 4.2. However, such an approach would not necessarily help for the reasons below.

Recall that we focus on modeling time series with both outliers and change points. Both an outlier and a change

[1]https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70

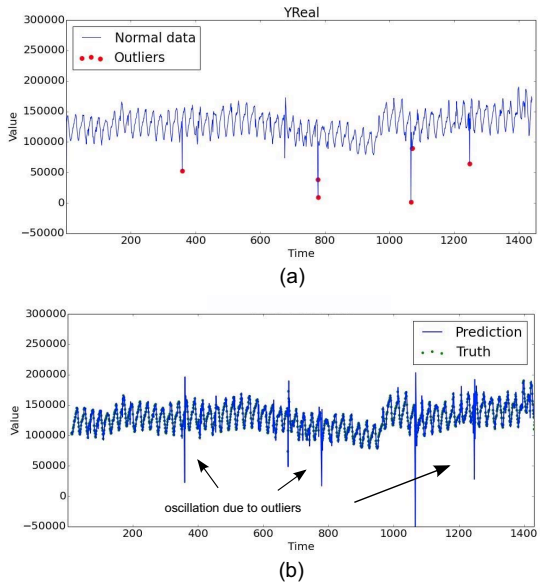


Fig. 1: Effect of outliers on the LSTM neural network. (a) the original time series (b) online prediction results. (best viewed in colour)

point could lead to the identification of suspicious points. For instance, Figure 2 shows the p-values at each time instant during the learning process of LSTM neural network in the experiment as Figure 1. It is observed that outliers, change points and some regular points are flagged as suspicious points. It is difficult to distinguish them from each other in such a noisy and non-stationary environment [5], [13].

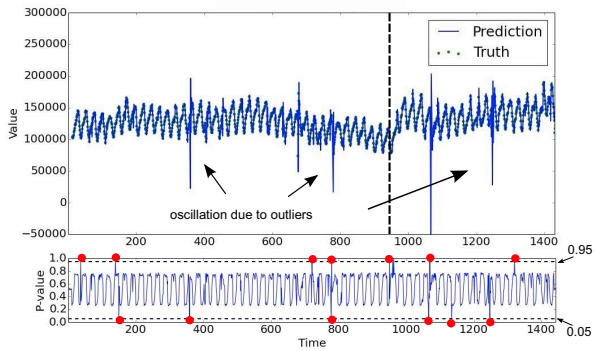


Fig. 2: The observed time series (top) and p-values of data points in the time series modeled by conventional online learning of LSTM neural network (bottom) (best viewed in colour). The vertical dotted line in the top figure indicates the possible change point. Red points in the bottom figure represent the time instants identified as suspicious points.

Meanwhile, when a real change point is misclassified as an outlier and removed from the neural network update, the data points immediately following the change point would then likely be regarded as outliers as well, since they follow the new distribution after the change point [5], [6]. Simply skipping such data points results in important information being lost to the LSTM updating. It could delay the learning towards the new data distribution after the change point and significantly degrade the performance of LSTM neural network afterwards.

Integrating change point detection into LSTM will hardly help as well. Most of change point detection techniques are the instances of the following framework: at time instant t statistical tests [5], [6], [13], [27] are performed over two data subsets, the intermediate past data points $\{x_i\}_{i=t-m_1,\ldots,t-1}$ and the intermediate future data points $\{x_k\}_{k=t+1,\ldots,t+m_2}$ to measure the dissimilarity, so as to determine whether t is a change point. Therefore, it requires to accumulate data to locate a change point in the history. In this paper, we consider the setting where the learning process of LSTM neural network is supposed to response to individual data points online, so as to adapt to data changes promptly and provide timely prediction.

C. Weighted Gradient Online learning of LSTM for time series

In this part, we present a novel online learning algorithm for LSTM neural network to be capable of making robust prediction over time series in the presence of both outliers and change points.

Overview: We propose the **w**eighted **g**radient online learning algorithm, referred to as WG-Learning. The key idea is to leverage local features, which are extracted from a sliding window over time series, to dynamically weight the gradient, i.e. Eq. (13), at each time instant for updating the current neural network. Intuitively, if a suspicious point is highly possible to be an outlier based on local features, the corresponding gradient is down-weighted, so as to avoid deviating the so-far trained LSTM neural network from the current underlying data distribution. Otherwise, if it is believed to be a change point, the associated gradient is retained by a high weight, such that the LSTM model can quickly adapt to the new data.

A common characteristic of time series analysis is that temporal continuity plays a key role [22], [34], [39], [40]. In this sense, time forms the contextual variable with respect to which all analysis is performed. Therefore, our WG-Learning maintains a sliding window over the time series and associated p-values for the data points in the sliding window.

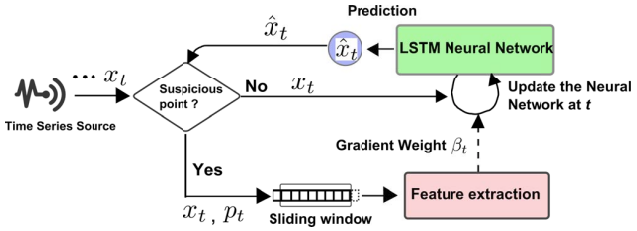


Fig. 3: Work-flow of WG-Learning based LSTM neural network in online learning of time series

As is shown in Figure 3 which outlines the work-flow of our WG-Learning, if the data point at the current time instant is a suspicious point based on Definition 4.2, two features

presented below are extracted from the sliding window and took into account to weight the gradient used in updating the LSTM neural network at this time instant. For instance, at time t, the local features *w.r.t.* data point x_t are discovered in $\{x_{t-w}, x_{t-w+1}, \ldots, x_{t-1}\}$ and $\{p_{t-w}, p_{t-w+1}, \ldots, p_{t-1}\}$. w is the length of the sliding window (for the extension of adaptive windows, please refer to Section IV-D).

Then, in WG-Learning formally the neural network is updated by the following formula:

$$W_t = W_{t-1} - \eta \cdot \{\beta_t \bigtriangledown E(W_{t-1})\} \tag{14}$$

where β_t is the time-varying gradient weight based on the features extracted from data, which will be defined below. Previous adaptive learning rate approaches [36], [45] fail to take into account the data characteristics in updating parameters of the neural network. Before presenting the details, we first summarize the benefits of our approach for learning noisy and non-stationary time series as follows:

- Reduced impact of outliers as a sudden increase of the prediction error - our WG-Learning provides reliable and smooth prediction robust to outliers as well as avoiding volatile huge prediction errors.
- Adaptability to changing data - the dynamic gradient weighting mechanism in WG-Learning enables to retain useful information in updating the neural network for converging to changed data.
- Reduction of the overall prediction errors.

Next, we will first present the features and then how to leverage them to weight the gradient.

Suspicion Ratio.

The suspicion ratio is formulated based on the following observation: outliers are rare events in time series [18], [40], [44] and would lead to the limited appearance of suspicious points in the sliding window; however, when change point happens, the neural network needs to consume some data after the change point for adapting to the new hidden distribution and therefore may yield high proportion of suspicious points in the beginning. Therefore, the ratio of suspicious points in the sliding window can be a factor in weighting the gradient.

Now we give the formal definition. In the sliding window *w.r.t.* time t, $\{x_{t-w}, x_{t-w+1}, \ldots, x_{t-1}\}$, the suspicion ratio is defined as:

$$s_t = \frac{\sum_{i=t-1}^{t-w} \mathbb{I}_{\{p_i > 1-\alpha \text{ or } p_i < \alpha\}}}{w} \tag{15}$$

where $\mathbb{I}_{\{\cdot\}}$ is the indicator function.

Suspicion ratio is positively correlated with the gradient weight. On the first hand, the less the suspicion ratio in the sliding window, the more likely the suspicious point x_t is to be an outlier. Even if it is a change point in reality, the subsequent data points after this change point are highly possible to be suspicious points and thus lead to increased gradient weight in updating the model. It will not significantly delay the convergence of our neural network on changed data.

On the other hand, when the suspicion ratio is high, i.e., the suspicious points appear frequently in the sliding window, further investigation is required. This is because it is possible that some outliers just occur in the sliding window or such suspicious points are due to a real change point. In such a case, additional features should be considered for weighting the gradient weight.

Difference Value Drift.

In this part, we introduce the second feature, difference value drift. This novel feature helps to "amplify" the difference between outliers and change points from the perspective of the value domain of time series, thereby providing enhanced information to adjust the gradient weight.

First, we extract the normal and suspicious points from the sliding window at time t, i.e, $\{x_{t-w}, \ldots, x_{t-1}\}$ and respectively denote them by a set $\mathcal{N}_t = \{x_k | k \in \{t-w, \ldots, t-1\}, \alpha \leq p_k \leq 1-\alpha\}$ and a set $\mathcal{S}_t = \{x_k | k \in \{t-w, \ldots, t-1\}, p_k > 1-\alpha \text{ or } p_k < \alpha\}$. $|\mathcal{N}_t|$ and $|\mathcal{S}_t|$ represent the numbers of data points in the sets.

Then, we define the neighbours of a normal point in the sliding window as:

Definition 4.3 (Neighbours of a Normal Point): The left and right neighbours of a normal point x_i are the most recent two normal data points before and after x_i in the sliding window: $f_l(x_i) = x_{\max\{k:k<i, x_k \in \mathcal{N}_t\}}$ and $f_r(x_i) = x_{\min\{k:k>i, x_k \in \mathcal{N}_t\}}$.

Now we are ready to introduce the proposed *difference value*, which is novel in the sense that it is derived by a bi-directional differencing process and has two forms respectively specific for normal and suspicious points as follows:

- For a suspicious point (e.g., x_k) in the sliding window, the difference value is the average of absolute changes between x_t and its intermediate preceding and succeeding data points, i.e. x_{k-1} and x_{k+1}, no matter whether they are normal points or not. (if any of the two points are out of the sliding window, only one-side change is taken as the difference value).
- For a normal points (e.g., x_i) in the sliding window, the difference value is evaluated as the average difference between x_i and its two neighbours, which are defined in Definition 4.3, such that it captures the normal pattern of time series without the effect of potential outliers and change points.

The motivation behind the difference value is as follows. Differencing is an important tool in time series analysis [40], which helps to stabilize the mean of a time series by removing changes in the level of a time series. On the first hand, outliers are often abrupt and violate changes *w.r.t.* the intermediate preceding and succeeding normal points, thereby yielding high difference values [18], [34]. Even if outliers occur consecutively, their difference values could present abnormal fluctuating patterns compared with the normal data points. On the other hand, for the suspicious points resulting from a change point, they are under the new data distribution and expected to adhere to the property of normal time series as well, i.e., having reasonable magnitudes in difference values.

Based on above analysis, we give the formal definition of the difference value drift. At time instant t, *difference value drift* d_t between suspicious and normal points in the sliding window is defined as:

$$\frac{(x_t - x_{t-1}) + \sum\limits_{x_k \in \mathcal{S}_t} (|x_k - x_{k-1}| + |x_k - x_{k+1}|)}{\sum\limits_{x_i \in \mathcal{N}_t} (|x_i - f_l(x_i)| + |x_i - f_r(x_i)|)} \frac{|\mathcal{N}_t|}{|\mathcal{S}_t| + 1} \tag{16}$$

This is a relative metric to evaluate the discrepancy between the average difference values of normal and suspicious points in the sliding window. The higher the value of the difference value drift, the more probable the suspicious points are to be outliers.

The proposed two features above are complementary by providing different views to weight the gradient in updating the neural network. Now we can define the gradient weight β_t in (14) integrating the suspicion ratio and the difference value shift as:

$$\beta_t = \lambda \cdot e^{-d_t \mathbb{I}_{\{d_t \geq \gamma\}}} + (1 - \lambda) \cdot s_t \tag{17}$$

where $\lambda \in [0, 1]$ is the parameter to control the importance of the two features in the weight. The term $e^{-d_t \mathbb{I}_{\{d_t \geq \gamma\}}}$ models the difference value drift in a bounded range $[0, 1]$. Parameter γ represents the prior belief about the normal difference value drift. $e^{-d_t \mathbb{I}_{\{d_t \geq \gamma\}}}$ gives full weight, i.e., 1 when d_t is below γ. Otherwise, the more the value of d_t, the more probable it is that outliers occur and thus the decreasing weight is given to the gradient, as d_t increases.

As for online prediction, a particular feature of time series modeling is that a target data point at a certain time instant will become the predictor of future data points. For instance, at time t, predictors $(x_{t-1}, x_{t-2}, \ldots, x_1)$ are used to make prediction on t and update the model together with x_t. Then, x_t will be used to predict values on time $t+1, \ldots$ and so on. The problem with such a procedure is that if x_t is an outlier and thus flagged as a suspicious point, the prediction based on x_t is biased, though the model is maintained properly in WG-Learning.

Therefore, we propose to use the forecasting from the learnt neural network to replace suspicious points for the subsequent prediction based on the idea of sequence modeling with RNN [17]. Formally, the prediction process is now expressed as:

$$\hat{x}_t = g(\tilde{x}_{t-1}, \tilde{x}_{t-2}, \ldots, \tilde{x}_1; W_{t-1}) \tag{18}$$

where $\forall k \in \{1, \ldots, t-1\}$,

$$\tilde{x}_k = \begin{cases} \hat{x}_k \text{ as (10)} & x_k \text{ is a suspicous point} \\ x_k & \text{otherwise} \end{cases}$$

D. Discussion

Our proposed WG-Learning can be extended to perform outlier and change point detection with ease. Since our WG-Learning is able to provide robust prediction, we can integrate additional sub-windows into the sliding window to model the error distributions of normal behaviours in time series [5]. Then, the statistics extracted from sub-windows is leveraged to perform various statistical test [5], [6], [13] to locate the occurrences of change points. For the suspicious points not identified as change points, they are reported as outliers.

As for the size of the sliding window, we can adapt time-varying windowing techniques [6], [13] to our WG-Learning, so as to dynamically adjust the window size sufficient for summarizing the error distribution of the current neural network.

V. Experimental Analysis

In this section, we conduct extensive experiments to demonstrate the prediction performance of the proposed method WG-Learning.

A. Experiment Setup

Datasets: we use both synthetic and real datasets for the evaluation.

Synthetic: this synthetic dataset is produced based on a time series generator [1]. It generates segments of time series by randomly sampling the mean μ_i, variance v_i, trend slop u_i and length l_i specific for each segment i from the prior, $\mu_i \in [0, 100]$, $v_i \in [10, 30]$, $u_i \in [-0.5, 0.5]$ and length $l_i \in [500, 1000]$. The data points in each segment are then the samples from the Gaussian distribution with the corresponding mean and variance plus the trend component. The number of points in each segment is the sampled length. In this way, we obtain time series with change points, which are the boundaries between segments. Outliers are injected into each segment based on a Bernoulli distribution specified by β and thus $l_i \cdot \beta$ is the expected number of outliers in the segment of time series. β is chosen as 0.01. The magnitude of outliers is defined as the times of the variance of the segment. By default, the magnitude is 10, which means that the value of an outlier data point is sampled from a Gaussian distribution with 10 times larger mean than the mean of the corresponding segment.

YSyn: the synthetic time series is from Yahoo's S5 dataset[2]. The dataset consists of both real and synthetic time-series. We use the time series in A4Benchmark containing synthetic outliers and change-points. The synthetic time-series have varying noise and trends with pre-specified seasonalities.

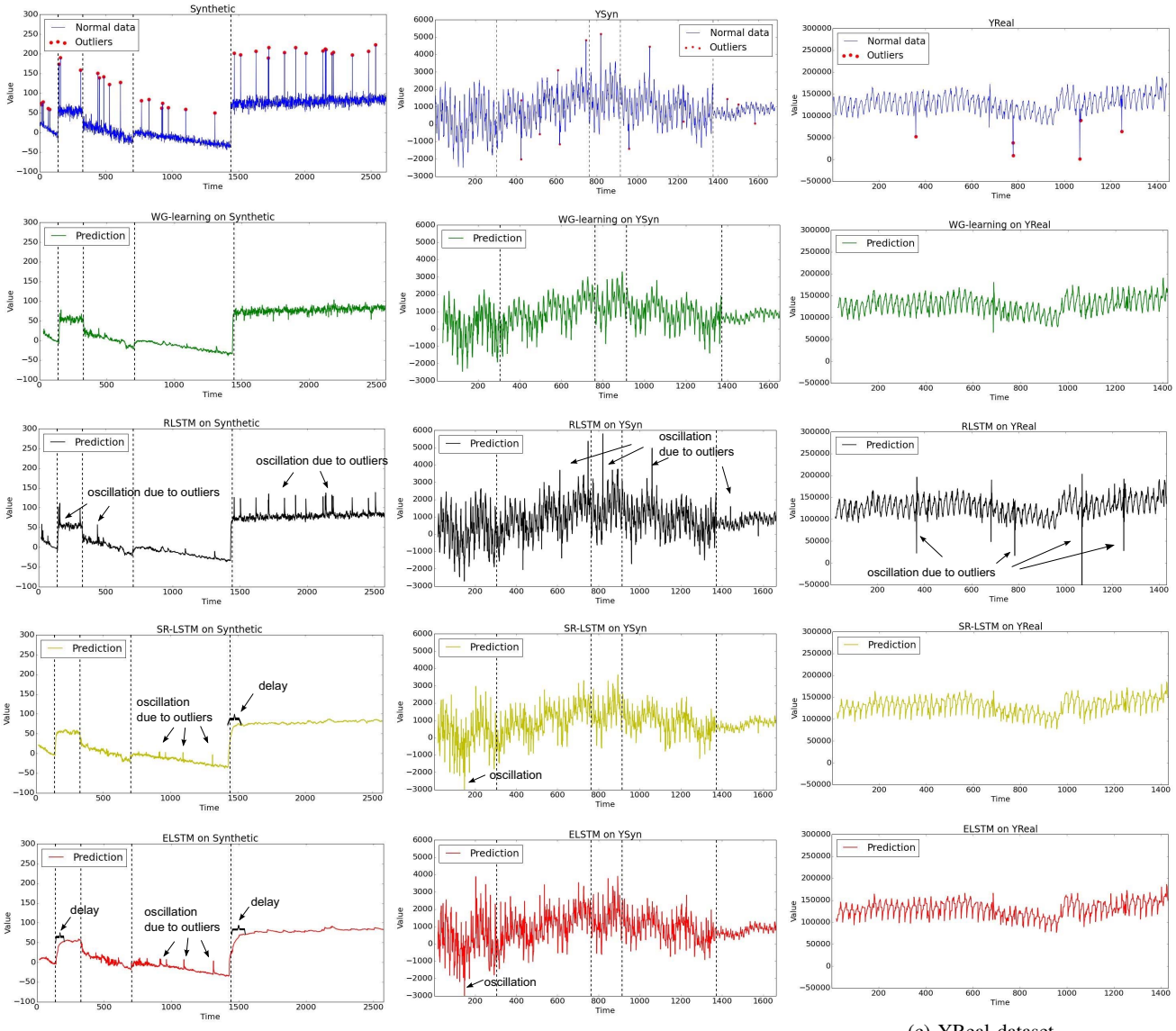
YReal: the real time series is from the A1Benchmark data in Yahoo's S5 dataset. The time series representing the metrics of various Yahoo services contains both outliers and change points.

Baselines: We compare the proposed method with three baselines.

RLSTM: this refers to the conventional **r**eal-time current learning (RTRL) of LSTM neural networks [41]. It updates the network directly using the newly available data without considering the effect of outliers and change points.

SR-LSTM: it stands for the online learning of LSTM with **s**uspicious point **r**emoval based on the idea of [12]. Specifically, once a suspicious point is detected, SR-LSTM skips

[2]https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70



(a) Synthetic dataset (b) YSyn dataset (c) YReal dataset

Fig. 4: Online one-step ahead prediction (best viewed in colour). The figures in each column respectively visualize the observed time series, and the predicted ones with the baselines and the WG-Learning. The proposed method provides a smooth prediction resistant to outliers and change points.

the model update on this point. This method may circumvent the adverse effect of outliers, but could lead to loss of useful information for changing patterns.

ELSTM: this baseline is based on [8] to integrate both outlier and change point detection into LSTM. It uses the statistical control chart (e.g., EWMA) [5], [8] to monitor the distribution of prediction errors. Based on the monitored statistics, for a suspicious outlier, the model is updated by using a recent normal point, while it is trained as usual when the point is identified as a change point.

SR-LSTM and ELSTM have the confidence level parameter in common with WG-Learning, and they are all set to 0.05 as usual in typical statistical tests [5], [22]. Additionally, ELSTM has a parameter, the moving average weight with the usual recommendation range $[0.1, 0.3]$ [5]. In the experiments below, we set it to 0.2. The size of sliding window in WG-Learning is set as 20. λ and γ are respectively set to 0.8 and 5

We use the same LSTM model for the baselines and the proposed method in the experiments. In particular, the number of layers is 3, and the number of nodes (neurons in LSTM layer) is 400. The mean squared error is chosen as the loss function. The L2 regularization with 0.0001 penalty is used.

Nesterov momentum is adopted as the optimizer. Our WG-Learning and baselines are implemented based on Theano[3].

Evaluation Metric: We evaluate the predictive performance of WG-Learning and baselines in terms of root mean square error (RMSE) excluding the outliers. Lower values of RMSE are considered better. At each time instant, we perform one-step ahead prediction using the online model and then update the model when the new data point arrives.

B. Experiment Results

Table II shows the prediction results on each dataset. WG-Learning outperforms the baselines in all datasets by tracking the variation of time series and resisting to outliers. We observe that for the datasets with large outliers e.g., YReal, WG-Learning significantly outperforms RLSTM. For the datasets (Synthetic and YSyn) having medium outliers, WG-Learning still achieves better results. SR-LSTM and ELSTM occasionally get better than RLSTM depending on datasets. For the very dynamic dataset, e.g., YSyn and YReal, SR-LSTM has higher errors than RLSTM, this could be because the aggressive update skipping mechanism of SR-LSTM leads to missing useful information.

TABLE II: Prediction accuracy on synthetic and real datasets (RMSE).

approach \ dataset	Synthetic	YSyn	YReal
RLSTM	11.01	562.56	10104.24
SR-LSTM	10.03	669.63	13816.99
ELSTM	10.31	660.02	13789.21
WG-Learning	**9.62**	**502.79**	**8928.92**

Figure 4 visualizes the predictive performance of the online model for the time period with complex outliers and change points in each dataset. The top plots of Figure 4 visualize the observed time series of the datasets Synthetic, YSyn and YReal. The red points represent the outliers and the vertical dotted lines indicate change points. YReal has no explicit change points and thus no vertical line is shown in the corresponding figure. The other plots of Figure 4 show the predicted time series using the three baselines and WG-Learning.

From Figure 4, we can observe that in dataset Synthetic, our WG-Learning is able to catch the details of time series as well as promptly adapting to the data after change points. Meanwhile, WG-Learning can circumvent the adverse effect of outliers by sticking to the underlying patterns of the time series. It indicates that the weighted gradients of WG-Learning are indeed effective. On the contrary, RLSTM suffers from the outliers by presenting predictions having oscillations around outliers as is shown in the second row of Figure 4. Compared with WG-Learning, the prediction of SR-LSTM and ELSTM resembles a sliding average and may loss details of the time series, though they present robust prediction compared with RLSTM. Meanwhile, SR-LSTM and ELSTM are lagged on change points. In ELSTM, a change point is detected when the data points following it incur sufficient change in prediction errors [5], [6], [13]. This means in the online learning, there is a delay in change point detection, and thus ELSTM may miss the update of changing data points. With similar tendency, our WG-Learning provides consistent robustness and adaptiveness on datasets YSyn and YReal.

In addition, we investigate the weighted gradients of WG-Learning. We record the gradient weight at each data point (i.e., β_t in Eq. (14)) during the online learning process and visualize them associated with the observed time series in Figure 5. We can observe that: most of the gradient weights are close to one at the regular points, and in turn the model can be normally updated in an online fashion. However gradient weights are much less than one at the detected outliers, which avoid deviating the model from the underlying patterns of the time series. On the other hand, gradient weights at the detected change points are mostly retained close to one, such that WG-Learning is able to adapt to the new patterns in real time.

[3]http://deeplearning.net/software/theano/

VI. Conclusion

In this paper we propose an adaptive gradient learning method for recurrent neural networks (RNN) in the context of online learning problem. The WG-Learning aims to incrementally learn the streaming time series and provide robust predictions adapting to the changing patterns as well as resisting to outliers. In the WG-Learning, we introduce the weighted gradient to the online SGD for the RNN models, based on the local features of time series. The method enables to update the RNN models with downweighted gradients for outliers while full gradients for change points. The predictive performance of the WG-Learning in the extensive experiments on both synthetic and real datasets verifies the superiority of the WG-Learning over the baselines.

Acknowledgments This work was partly supported by Nano-Tera.ch through the OpenSense2 project and the EU FP7 Project SMARTIE (contract no. 609062).

References

[1] R. P. Adams and D. J. MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

[2] O. Anava, E. Hazan, S. Mannor, and O. Shamir. Online learning for time series prediction. In *Proceedings of the Conference on Learning Theory*, pages 172–184, 2013.

[3] O. Anava, E. Hazan, and A. Zeevi. Online time series prediction with missing data. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 2191–2199, 2015.

[4] T. W. Anderson. *The Statistical Analysis of Time Series*. John Wiley & Sons, 2011.

[5] M. Basseville, I. V. Nikiforov, et al. *Detection of abrupt changes: theory and application*, volume 104. Prentice Hall Englewood Cliffs, 1993.

[6] A. Bifet and R. Gavalda. Learning from time-changing data with adaptive windowing. In *SDM*, volume 7, page 2007. SIAM, 2007.

[7] E. Busseti, I. Osband, and S. Wong. Deep learning for time series modeling. Technical report, Stanford University, 2012.

[8] V. Chandola and R. R. Vatsavai. A gaussian process based online change detection algorithm for monitoring periodic time series. In *SDM*, pages 95–106. SIAM, 2011.

[9] C. Chatfield. *The Analysis of Time Series: An Introduction*. CRC Press, 2013.

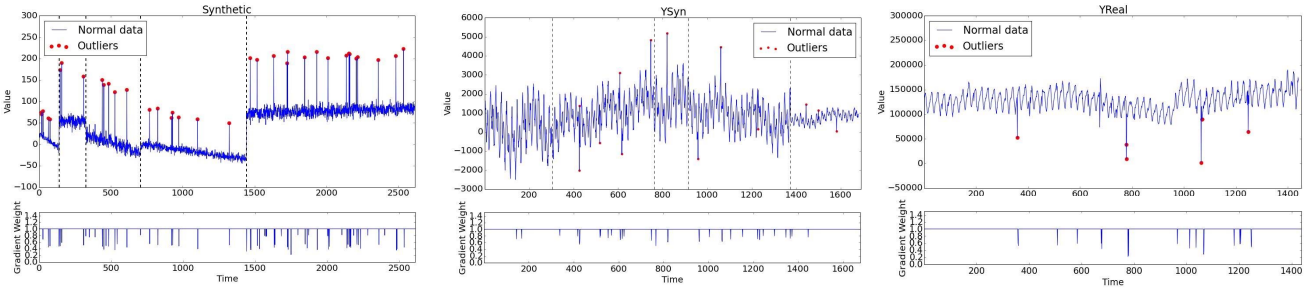


Fig. 5: Gradient weights of WG-Learning (best viewed in colour): the observed time series (top) and the corresponding weights (bottom).

[10] S. Chauhan and L. 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*, pages 1–7. IEEE, 2015.

[11] 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.

[12] J. T. Connor, R. D. Martin, and L. E. Atlas. Recurrent neural networks and robust time series prediction. *Neural Networks, IEEE Transactions on*, 5(2):240–254, 1994.

[13] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys (CSUR)*, 46(4):44, 2014.

[14] R. Garnett, M. A. Osborne, and S. J. Roberts. Sequential bayesian prediction in the presence of changepoints. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 345–352. ACM, 2009.

[15] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471, 2000.

[16] A. Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*. Studies in Computational Intelligence. Springer, 2012.

[17] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[18] M. Gupta, J. Gao, C. Aggarwal, and J. Han. Outlier detection for temporal data. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 5(1):1–129, 2014.

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

[20] L. C. Jain, M. Seera, C. P. Lim, and P. Balasubramaniam. A review of online learning in supervised neural networks. *Neural Computing and Applications*, 25(3-4):491–509, 2014.

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

[22] N. Laptev, S. Amizadeh, and I. Flint. Generic and scalable framework for automated time-series anomaly detection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1939–1947. ACM, 2015.

[23] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

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

[25] C. Liu, S. C. H. Hoi, P. Zhao, and J. Sun. Online arima algorithms for time series prediction. In *Proceedings of the 13th AAAI Conference on Artificial Intelligence*, 2016.

[26] J. Liu, K. Zhao, B. Kusy, J.-r. Wen, and R. Jurdak. Temporal embedding in convolutional neural networks for robust learning of abstract snippets. *arXiv preprint arXiv:1502.05113*, 2015.

[27] S. Liu, M. Yamada, N. Collier, and M. Sugiyama. Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 43:72–83, 2013.

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

[29] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal. Long short term memory networks for anomaly detection in time series. In *European Symposium on Artificial Neural Networks*, volume 23.

[30] J. Martens and I. Sutskever. Learning recurrent neural networks with hessian-free optimization. In *Proceedings of the 28th International Conference on Machine Learning*, 2011.

[31] L. L. Minku and X. Yao. Ddd: A new ensemble approach for dealing with concept drift. *IEEE Transactions on Knowledge and Data Engineering*, 24(4):619–633, 2012.

[32] D. C. Montgomery, C. L. Jennings, and M. Kulahci. *Introduction to Time Series Analysis and Forecasting*. John Wiley & Sons, 2011.

[33] C. Richard, J. C. M. Bermudez, and P. Honeine. Online prediction of time series data with kernels. *IEEE Transactions on Signal Processing*, 57(3), 2008.

[34] G. Ristanoski, W. Liu, and J. Bailey. A time-dependent enhanced support vector machine for time series regression. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 946–954. ACM, 2013.

[35] Y. Saatçi, R. D. Turner, and C. E. Rasmussen. Gaussian process change point models. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 927–934, 2010.

[36] A. Senior, G. Heigold, M. Ranzato, and K. Yang. An empirical study of learning rates in deep neural networks for speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 6724–6728. IEEE, 2013.

[37] I. Sutskever. *Training Recurrent Neural Networks*. PhD thesis, University of Toronto, 2013.

[38] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[39] O. Vallis, J. Hochenbaum, and A. Kejariwal. A novel technique for long-term anomaly detection in the cloud. In *6th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 14)*, 2014.

[40] W. W.-S. Wei. *Time series analysis*. Addison-Wesley publ Reading, 1994.

[41] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[42] W. K. Wong, M. Xia, and W. C. Chu. Adaptive neural network model for time-series forecasting. *European Journal of Operational Research*, 207(2):807–816, 2010.

[43] Y. Wu, J. M. H. Lobato, and Z. Ghahramani. Gaussian process volatility model. In *Advances in Neural Information Processing Systems 27*, 2014.

[44] K. Yamanishi and J.-i. Takeuchi. A unifying framework for detecting outliers and change points from non-stationary time series data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 676–681. ACM, 2002.

[45] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.