electronics

Article

Spatial and Temporal Normalization for Multi-Variate Time Series Prediction Using Machine Learning Algorithms

Alimasi Mongo Providence [1], Chaoyu Yang [2,*], Tshinkobo Bukasa Orphe [1], Anesu Mabaire [1] and George K. Agordzo [3]

[1] School of Economics and Management, Anhui University of Science and Technology, Huainan 232000, China
[2] School of Artificial Intelligence, Anhui University of Science and Technology, Huainan 232000, China
[3] School of Mathematics and Big Data, Anhui University of Science and Technology, Huainan 232000, China
* Correspondence: yangchy@aust.edu.cn

Abstract: Multi-variable time series (MTS) information is a typical type of data inference in the real world. Every instance of MTS is produced via a hybrid dynamical scheme, the dynamics of which are often unknown. The hybrid species of this dynamical service are the outcome of high-frequency and low-frequency external impacts, as well as global and local spatial impacts. These influences impact MTS's future growth; hence, they must be incorporated into time series forecasts. Two types of normalization modules, temporal and spatial normalization, are recommended to accomplish this. Each boosts the original data's local and high-frequency processes distinctly. In addition, all components are easily incorporated into well-known deep learning techniques, such as Wavenet and Transformer. However, existing methodologies have inherent limitations when it comes to isolating the variables produced by each sort of influence from the real data. Consequently, the study encompasses conventional neural networks, such as the multi-layer perceptron (MLP), complex deep learning methods such as LSTM, two recurrent neural networks, support vector machines (SVM), and their application for regression, XGBoost, and others. Extensive experimental work on three datasets shows that the effectiveness of canonical frameworks could be greatly improved by adding more normalization components to how the MTS is used. This would make it as effective as the best MTS designs are currently available. Recurrent models, such as LSTM and RNN, attempt to recognize the temporal variability in the data; however, as a result, their effectiveness might soon decline. Last but not least, it is claimed that training a temporal framework that utilizes recurrence-based methods such as RNN and LSTM approaches is challenging and expensive, while the MLP network structure outperformed other models in terms of time series predictive performance.

Keywords: spatial-temporal systems; neural networks; machine learning; information systems; forecasting; time series

check for updates

Citation: Providence, A.M.; Yang, C.; Orphe, T.B.; Mabaire, A.; Agordzo, G.K. Spatial and Temporal Normalization for Multi-Variate Time Series Prediction Using Machine Learning Algorithms. *Electronics* **2022**, *11*, 3167. https://doi.org/10.3390/electronics11193167

Academic Editor: Jordi Guitart

Received: 18 July 2022
Accepted: 26 September 2022
Published: 1 October 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Correction Statement: This article has been republished with a minor change. The change does not affect the scientific content of the article and further details are available within the backmatter of the website version of this article.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

A crucial part of many industry sectors is forecasting, which is the method of predicting the present significance of time series data [1]. Forecasting distribution networks and airline requests, finance price levels, power, and traffic or weather systems are just a few examples of its applications. As disputed to univariate (solitary time series data) predicting, multi-variate time series analysis is frequently necessary for large statistics of linked time series data. Suppliers may, for instance, need to forecast the sales and requests of millions of different commodities at tens of thousands of varying locations, resulting in billions of marketing time series data. The statistical modeling research has widely covered multivariate time series prediction, which entails learning from traditional multi-variate data to estimate the prospective qualities of several factors. However, the majority of the studies (also latest) concentrate on linear methods, deal with situations in which only a small number of variables are available, and only occasionally use forecasting horizons

greater than another uses. In financial predicting research, particularly Dynamic Factor Models (DFM) [2], which are typically limited to linear methods, will more frequently consider huge difference setups. The big information transformation is currently calling for methods that can handle very multiple amounts of non-linear time series, possibly closely associated or predicated, and forecast their transformation over longer timeframes [3]. The Internet of Things (IoT) devices, of which the key effect is the development of a constant flow of spatiotemporal transmissions predicted to be generated and evaluated, serve as the most obvious source of inspiration [4]. This is already taking place in an increasing number of research and applied fields, including financial services, meteorology, industrial activities, atmospheric engineering, and physical sciences. As a result, performing univariate forecasting instead of multi-variate forecasting is also a common strategy. The most popular statistical forecasting techniques in use today in business include exponential smoothing (ES), auto-regressive AR and ARIMA designs, and more overall state space designs [5]. These techniques utilize a simple mathematical model of historical data and future predictions. Until recently, these techniques consistently surpassed machine learning techniques such as RNNs in large-scale predicting contests. Multi-task univariate projections, which shares deep learning design variables across all sequence, possibly including some series-specific basis functions or parameterized range of fundamental, is a major factor in the latest achievements of deep learning for forecasting. For instance, a hybrid ES-RNN framework [6] simultaneously learns different seasons and level ES variables for every sequence to regulate those gained by the M4 predicting competition. This model forecasts each series using a single cohesive univariate RNN prototype.

In many commercial and industrial implementations, time series predictions are a critical issue. For example, if a public transportation provider can predict that a specific area will experience a supply issue in the coming hours, they could allocate enough capacity to reduce queuing times in that area in advance [7]. As another illustration, a Robo-advisor that can anticipate a prospective financial collapse can help an investor avoid financial loss. Real-world time series frequently exhibit varied dynamics because of the complicated and constantly changing influencing factors. This makes them exceptionally non-stationary. For example, the state of the road, its location, the time of day, and the climate all have an important influence on the amount of circulation that passes over it. The latest season, value, and product all play a role in determining a product's sales within the retail industry [8]. Time series forecasting faces a great deal of difficulty as a result of the diverse interactions. This would then research multi-variate time series prediction in this research, their several variables changing over time. Numerous disciplines, including physical sciences, engineering, weather forecasting, and tracking of human health, are conducting extensive research on time series prediction [9]. We proposed the most appropriate technique to test, evaluate, and verify the most popular forecasting methodologies using a collection of information. It appears that using just a few of the models cannot be a suitable method for simulating the hydrological time series. In such cases, time series modeling and artificial intelligence models might be combined to account for hydrological processes rather than utilizing a single model [10]. It is well recognized that the experimental dataset, the machine learning model, and the use of efficient variables for model creation depending on such a challenge are all very important components in building a reliable machine learning technique [11].

A non-linear multi-variate (or variable) time series' multi-step ahead prediction is recognized to be quite challenging. If a forecasting task is modeled as an autoregressive procedure, this task poses several formidable difficulties for every learning machine, including the high dimensional space of outputs and inputs, cross-sectional and seasonal high dependence (which results in both non-linear multi-variate connections within inputs as well as a nonlinear framework within outputs), and last but not least, the danger of error reproduction [12]. Earlier work has concentrated on a particular sub-problem product's sale within the issue of the one-step-ahead predictive model of sequential multi-variate time series, discussions of the issue of multiple-step-ahead predicting of such a univariate time series, and the latest manuals that individuals take into account linear methods [13].

In a wide range of works, the topic of dimensionality decrease is protected more normally, although without addressing how to expand it to multiple steps predicting.

Extraction of structures and characteristics that characterize the key characteristics of the data is frequently the first step in the analysis and investigation of a time series (and other kinds of analysis). The exploitation of worldwide time series characteristics (including spectral characteristics measured with a conversion, such as the Discrete Cosine or Wavelet Transforms) and utilization of such global characteristics (that characterize time series characteristics as a collective) for archiving are standard techniques in the research [14]. Worldwide, fingerprints of multi-variate time series data could be extracted using correlations, transfer functions, statistical groupings, spectral characteristics, Singular-Value Decomposition (SVD), and related eigen decomposition. Tensor degradation is the equivalent analysis procedure on a tensor that can be utilized to depict the time dynamics of multi-modal information [15]. Costly methods include tensor and matrix degradation processes, probabilistic methods (such as Dynamic Topic Modeling, DTM), and Autoregressive Integrated Moving Average (ARIMA) predicated analysis, which divide a statistical model into informatics, moving average, and autoregressive elements for simulation and prediction. A dependable structure for modeling and learning time series structures is offered by conventional time series forecasting techniques, such as ARIMA and state-space models (SSMs). These techniques, however, have a great implication for the normality of a time series, which poses serious practical challenges if the majority of the influencing factors are not accessible. Deep learning methods have recently advanced to the point where they can now handle complex dynamic nature as a single entity, even in the absence of increased affecting variables. Recurrent neural networks (RNN) [16], long-short term memories (LSTM), Transformer, Wavenet [17], and temporal convolution networks are popular neural structures used on time series information (TCN). The key would be to further modify various components of different kinds from the initial measurement. Interactions that set dynamics apart from the spatial or temporal perspective can then be collected. This research offers two different types of normalization configurations that individually improve its high-frequency and local elements: Spatial Normalization (SN) and Temporal Normalization (TN) [18]. To do this, academics have become interested in applying ML approaches to create models that are more potent with greater accuracy. The shortcomings of traditional modeling methods were widely addressed by ML approaches to solve complicated environmental technical challenges [19]. This paper's contribution is the refinement of further categories of original measuring elements. Connections that set dynamics apart from the temporal or spatial view can then be represented. In this study, two different types of normalization modules are presented that individually enhance the high-frequency and local elements: temporal normalization (TN) and spatial normalization (SN). In particular, the local component makes it easier to separate dynamics from the spatial perspective, and the high-frequency component helps to distinguish dynamics from the temporal view. The system can uniquely fit every cluster of data because of its difference over space and time, specifically those long-tailed samples. The paper also demonstrates how the approach compares to existing state-of-the-art (SOTA) methods that use mutual relationship development to discern between dynamics.

Numerous applications produce and/or use multi-variate temporal data, but experts frequently lack the tools necessary to effectively and methodically look for and understand multi-variate findings. Efficient prediction models for multi-variate time series are crucial because of the incorporation of sensory systems into vital applications, including aviation monitoring, construction energy efficiency, and health management. Time series prediction methods have been expanded from univariate predictions to multi-variate time series predictions to meet this requirement. However, naive adaptations of prediction approaches result in an unwanted rise in the expense of model simulation, and more crucially, a significant decline in prediction accuracy since the extensive models are unable to represent the fundamental correlations between variates. However, research has shown that investigating both geographical and temporal connections might increase predictive

performance. These effects also influence how MTS will develop in the future, making it crucial to include them in time series forecasting work. Conventional approaches, however, have inherent limitations in separating the components produced by each type of effect from the source data. Two different normalization components are suggested with machine learning techniques to do this. The local component underlying the original data as well as the improved high-frequency element is separated by the suggested temporal and spatial normalization. Additionally, these modules are simple to include in well-known deep learning architectures like Wavenet and Transformer. Mixtures and original data can be difficult to distinguish using conventional methods. In this way, it incorporates well-known neural networks, such as the multi-layer perceptron (MLP), complex deep learning techniques, such as RNN and LSTM, two recurrent neural networks, support vector machines (SVM), and its application to regression, XGBoost, and others.

2. Related Works

Modern applications, such as climatic elements and requirement predicting, have high-dimensional time series estimation difficulties. In the latter requirement, 50,000 pieces must be predicted. The data are irregular and contain missing values. Modern applications require scalable techniques that can handle noisy data with distortions or missing values. Classical time series techniques often miss these issues. This research gives a basis for data-driven temporal learning and predicting, dubbed temporal regularized matrix factorization (TRMF). Create new regularization methods and scalable matrix factorization techniques for high-dimensional time series analysis with missing values. The proposed TRMF is comprehensive and includes multiple time series assessment methods. Linking autoregressive structural correlations to pattern regularization is needed to better comprehend them.

According to experimental findings, TRMF is superior in terms of scalability and prediction accuracy. Specifically, TRMF creates greater projections on real-world datasets such as Wal-Mart E-commerce data points and is two requirements of magnitude quicker than some other techniques [20].

Using big data and AI, it is possible to predict the citywide audience or traffic intensity and flow. It is a crucial study topic with various applications in urban planning, traffic control, and emergency planning. Combining a big urban region with numerous fine-grained mesh grids can display citywide traffic data in 4D tensors. Several grid-based forecasting systems for citywide groups and traffic use this principle to do reevaluating the intensity and in-out flow forecasting issues and submitting new accumulated human mobility source data from a smartphone application. The data source has many mesh grids, a fine-grained size distribution, and a high user specimen. By developing pyramid structures and a high-dimensional probabilistic model based on Convolutional LSTM, we offer a new deep learning method dubbed Deep Crowd for this enormous crowd collection of data. Last but not least, extensive and rigorous achievement assessments have been carried out to show how superior its suggested Deep Crowd is when compared to various state-of-the-art methodologies [21].

Regional forecasting is crucial for ride-hailing services. Accurate ride-hailing demand forecasting improves vehicle deployment, utilization, wait times, and traffic density. Complex spatiotemporal needs between regions make this job difficult. While non-Euclidean pair-wise correlation coefficients between possibly remote places are also crucial for accurate predicting, typical systems focus on modeling Euclidean interrelations between physically adjacent regions. This paper introduces the spatiotemporal multi-graph convolution network for predicting ride-hailing consumption (ST-MGCN). Non-Euclidean pair-wise relationships between regions are encoded into graphs before explicitly modeling correlation coefficients using a multi-graph transform. Perspective Landscaping recurrent neural networks, which add context-aware limits to re-weight historical observational data, are presented as a way to use global data to build association coefficients. This tests the suggested concept using two large-scale ride-hailing requirement data sources from the true world and finds that it consistently outperforms benchmarks by more than 10% [7].

To address multi-horizon probability forecasting, we use a data-driven technique to predict a time series distribution over upcoming horizons. Observed changes in historical data are vital for predicting long-term time series. Traditional methodologies rely on building a temporal relationship by hand to explore historical regularities, which is unrealistic for predicting long-term series. Instead, they propose learning how to use deep neural systems to display hidden knowledge and generate future predictions. In this study, an end-to-end deep-learning structure for multi-horizon time series prediction is proposed, along with temporal focus procedures to more efficiently capture latent patterns in historical data that are relevant for future prediction. Based on latent pattern properties, several future projections can be made. To accurately demonstrate the future, we also suggest a multi-modal fusion process that combines characteristics from various periods of history. Results from experiments show that the method produces cutting-edge outcomes on two sizable predicting data sources in various fields [22]. Multi-horizon prediction sometimes uses static (time-invariant) confounders, recognized future components, and other external time series only identified in the past. Deep learning techniques abound. "Black-box" designs do not describe how they employ real-world inputs. The Temporal Fusion Transformer (TFT) demands maximum multi-horizon prediction using temporal insights. TFT uses self-attention structures to create long-term connections and factors for learning temporal correlation at different scales and uses elements to pick relevant characteristics and gating pieces to override superfluous components. This improves performance above baselines on a wide range of real data sources and exhibits three TFT use cases [23]. Academic research struggles with hierarchical time series prediction. The precision of each hierarchical system, notably the interrupted time series at the bottom, is explored thoroughly. Hierarchical reunification boosts system productivity. This article provides a hierarchical prediction-to-alignment technique that considers bottom step projections changeable to improve upper hierarchical prediction performance. The bottom stage employed Light GBM for occasional time series and N-BEATS for constant time series. Hierarchical prediction with orientation is a simple but effective bottom-up improvement that adjusts for biases hard to discover at the bottom. It increases the average accuracy of less accurate estimates. The first author developed this study's technique in M5 Predicting Precision tournaments. The business-oriented approach may be effective for strategic business planning [24].

3. Methodology

3.1. Normalization

Since normalization is initially used in deep image processing, now almost all deep learning activities have seen a significant improvement in model performance. Each normalization approach has been future to report a specific gathering of computer vision applications, including group normalization, instance normalization, positional normalization layer normalization, and batch normalization [25]. Instance normalization, which was initially intended for image generation due to its ability to eliminate style data from its images, does have the highest opportunity for research. Researchers have discovered that attribute statistical data could collect an image's design and that after initializing the statistical data, the remaining characteristics are in charge of the image's substance. This ability to deliver an image's material in the fashion of another image, also recognized as extracting features, is made possible by its distinguishable assets. Similar to scale details in the time series is the style data in the image. Another area of research investigates the rationale behind the normalization trick's facilitation of deep neural network learning. One of their key findings is that normalization could improve the evaluation of an attribute space, allowing the framework to retrieve characteristics that are more different.

Figure 1 presents a high-level assessment of the structure used. Along the computation path, a few significant variables and their shapes have been branded at the appropriate locations. The structure normally has a Wavenet-like structure, with the addition of modules for spatial and temporal normalization, collectively referred to as ST-Norm or STN.

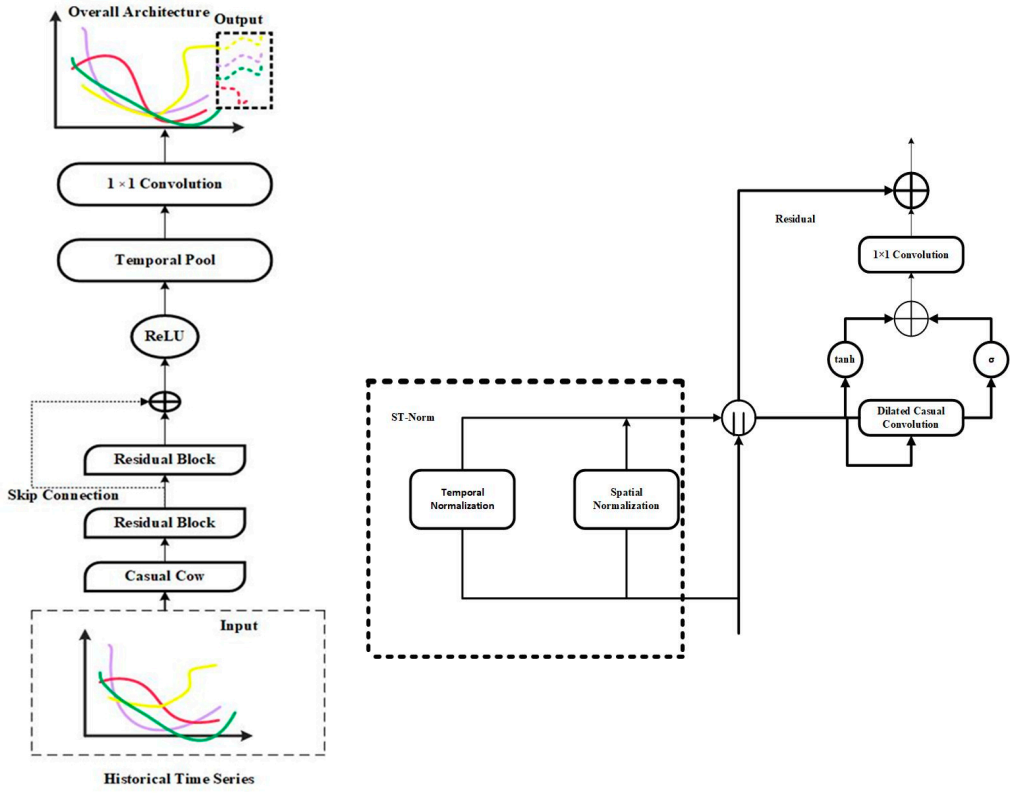

Figure 1. Overall high-level assessment structure.

3.2. Dilated Causal Convolution

This segment provides a brief introduction to dilated causal convolution, which applies the filter while skipping beliefs. The causal convolution on component t for the 1D signal $z \in \mathbb{R}^T$ and a filter $f : \{0, \ldots, k-1\} \to \mathbb{R}$ is described in Equation (1) as follows:

$$F(t) = (c * f)(t) = \sum_{i=0}^{k-1} f(i).c_{t-i} \tag{1}$$

This formula is simple to categorize for multi-dimension signals, but for the sake of brevity, it will not be included it here. To guarantee length continuity, padding (zero as well as recreate) to the dimension of $k-1$ is added to a left tail of a transmitter [26]. To give so every component a larger receptive ground, we combined several causal convolution layers.

Figure 2 shows the structure of dilated causal convolution. Trying to cause an outburst of characteristics when predicting long history is one drawback of using causal convolution, because the diameter of a kernel or its number of layers grows linearly with the dimensions of a receptive sector [27]. The obvious solution to this problem is pooling, but doing so compromises the signal's order details. To achieve this, dilated causal convolution is used, a form that encourages the exponential growth of an approachable pitch. The structured computing method is expressed in Equation (2).

$$F(t) = (c *_d f)(t) = \sum_{i=0}^{k-1} f(i).c_{t-d}.i, \tag{2}$$

In Equation (2), d is its component for dilation. Typically, d grows exponentially with network depth (namely, 2^l at stage l of a system). The variable d which denotes the dilated convolution operator decreases to the $*_d$ a normal convolution controller if d is 1 or (2^0).

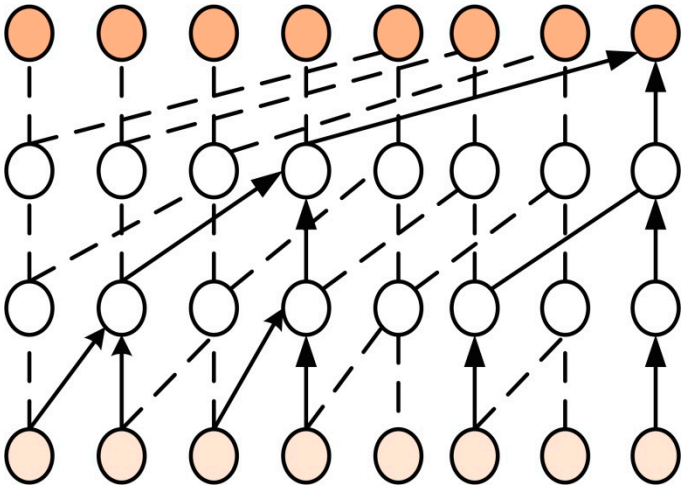

Figure 2. Dilated causal convolution architecture.

3.3. Neural Networks

3.3.1. Multi-Layer Perceptrons (MLPs)

A brief overview of Random Forests, multi-layer perceptrons (MLP), XGBoost, long-short term memory (LSTM) systems, and support vector regressors (SVR) is provided as an introduction to creating the research self-contained. MLP is regarded as an effective technique for capturing interactions among the parameter estimation that are not linear. It is being used effectively in hydrology, particularly time series modeling, hazard identification, and sediment supply [28]. The most popular artificial neural networks (ANN) for the classification of a regression issue are multi-layer perceptrons (MLPs). An input layer, each or many hidden layers, as well as an output layer, make up this category of designs [29]. A three-layer MLP is shown in Figure 3.

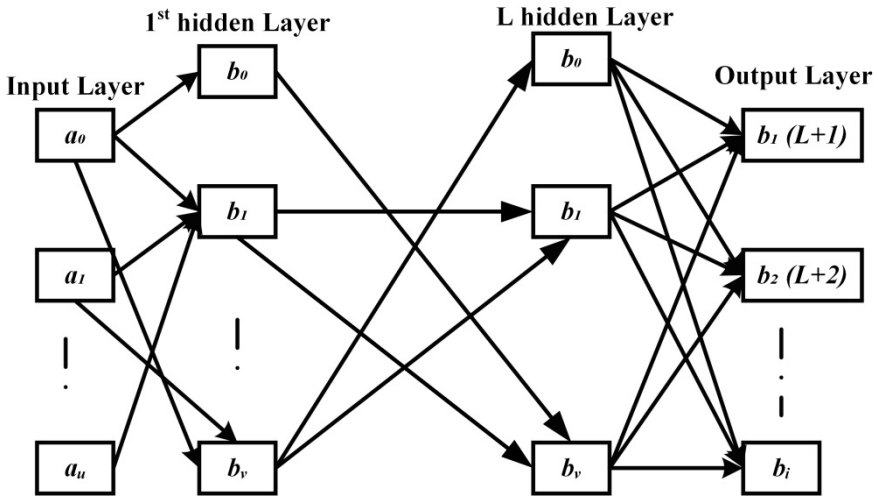

Figure 3. Multi-layer perceptron network graph layer.

A network diagram, for example, can produce the following results in Equation (3)

$$A(t) = \alpha_0 + \sum_{l=1}^{L}\left(\sum_{j=1}^{q}\alpha_{jl}g\left(\beta_{0jl} + \sum_{i=1}^{p}\beta_{ijl}X_{t-i}\right)\right) + \epsilon_t \quad (3)$$

The numbers $L_{,P,q}$ represent the number of hidden sections, inputs $X_t(i = 1, 2, \ldots p)$, and nodes in a specific hidden layer, respectively. The ReLU function $(g(x) = 1/(1 + e^{-a}))$ or the convolution $((e^a - e^{-a})/(e^a + e^{-a}))$ are some examples of activation functions

$ReLU(g(x) = \max(0, x))$. Equation (4) becomes simpler for networks including a single hidden layer:

$$X_t = \alpha_0 + \sum_{j=1}^{q} \alpha_j g\left(\beta_{0j} + \sum_{i=1}^{p} \beta_{ij} X_{t-i}\right) + \epsilon_t \tag{4}$$

3.3.2. LSTM

The LSTM is a design for such a recurrent neural network composed of three gates and two states: Input gate, Output gate, Forget gate, Cell state, and Hidden state.

Figure 4 displays the network's total schematic. In the mentioned reasoning, we will use it as a constant. This Figure 4 includes the hyperbolic tangent $\tanh(a) = (e^a + e^{-a})/(e^a - e^{-a})$ and the sigmoid $\sigma(a) = 1/(1 + e^{-a})$. The elements of the vector are subjected to activation functions [30]. Additionally, the element-wise addition and multiplication processes are denoted by $\odot$ and $\oplus$. The two related matrices are finally concatenated vertically when two lines intersect. The formula for this procedure is $\dagger : A \dagger B = \begin{pmatrix} A \\ B \end{pmatrix}$.

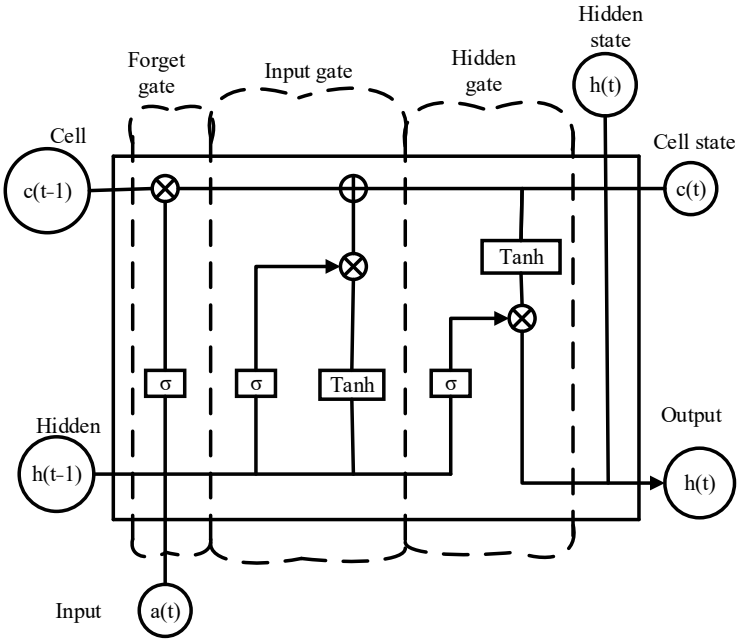

Figure 4. The cell structure of Long-Short Term Memory.

One could demonstrate $h^{\langle r \rangle}$ and $c^{\langle r \rangle}$ as a component of $h^{\langle r-1 \rangle}$, $c^{\langle r-1 \rangle}$, and $a^{\langle r \rangle}$ without using these formulae in Equation (5):

$$\begin{gathered} c^{\langle r \rangle} = \left(c^{\langle r-1 \rangle} \oplus \sigma\left(\ominus_f\left(h^{\langle r-1 \rangle} \dagger a^{\langle r \rangle}\right)\right)\right) \oplus \\ \left(\left(\tanh\left(\ominus_c\left(h^{\langle r-1 \rangle} \dagger x^{\langle r \rangle}\right)\right) \odot \sigma\left(\ominus_i\left(h^{\langle r-1 \rangle} \dagger a^{(r)}\right)\right)\right)\right) \\ h^{\langle r \rangle} = \tanh\left(c^{\langle r \rangle}\right) \odot \sigma\left(\ominus_o\left(h^{\langle r-1 \rangle} \dagger a^{\langle r \rangle}\right)\right) \end{gathered} \tag{5}$$

where $a^{\langle r \rangle}$, $c^{\langle r \rangle}$, and $h^{\langle r \rangle}$ stand for the input signal (time series amount at time r), approximate output importance for time r, and cell condition at time r, respectively. The characteristics of an LSTM framework were its matrices $\ominus_f, \ominus_i, \ominus_c, \ominus_o$.

3.3.3. SVM

The ϵ-insensitive loss capability is used by its support vector regression (SVR) algorithm, which was developed. The time series analysis A_t in SVR is transformed non-

linear, Φ, from its input space to such greater dimensional storage, which is denoted in Equations (6) and (7):

$$\Phi = \mathbb{R}^n \rightarrow F \tag{6}$$

$$f(a) = < u,\ \Phi(a) > + y \tag{7}$$

where the linear function f is minimized by a vector of characteristics (also known as weights) called w ^{2} F, and b ^{2} R is continuous. SVR typically selects the insensitive loss function again for minimization procedure instead of more traditional loss functions, such as the mean absolute percentage error (MAPE) and the least mean average error (MAE) [31]. One must reduce the risk formalized function to reduce its weight vector w, and subsequently, the function of f in Equation (8):

$$\begin{array}{c} min 1/2 \left| |w| \right|_2 + C \sum\limits_{i=1}^{l} \left(\xi_i + \xi_i^*\right) \\ s.t. b_i - < w, \Phi(a_i) > - b \leq \epsilon + \xi_i \\ < w, \Phi(a_i) > + y - b_i \leq \epsilon + \xi_i^* \end{array} \tag{8}$$

where $\epsilon \geq 0$ represents the separation among the real charge of y and the assumed shape of f. Slack variables $\xi, \xi^* \geq 0$ are added to accommodate errors bigger than that ϵ. When fitting training data, the regularization constant C is utilized to define the trade-off between generalization and precision.

In actuality, the Lagrangian multiplier-based expressions of w and f have been utilized in Equation (9):

$$\begin{array}{c} w = \sum\limits_{i=1}^{t} \left(\alpha_i - \alpha_i^*\right) \Phi(a_i) \\ f(a) = \sum\limits_{i=1}^{t} \left(\alpha_i - \alpha_i^*\right) K(a_i - a) - y \end{array} \tag{9}$$

where $\left(\alpha_i - \alpha_i^*\right) \leq C$ denotes the partial derivative of $\Phi(xi)$ and $\Phi(x)$, also known as the kernel features, and where $K(a_i, a)$. The literature provides more information on support vector machines and how to use them to solve regression issues.

3.3.4. Tree-Based Techniques

(a) Random forest regressor

As ensemble learning methods, random forests (RF) [32] could be used for correlation by averaging many different regression trees $z_n(a, \ominus_m, D_n)$, where $\ominus$ has been the model's parameters variable and D_n would be the training set $(A_1, B_1), \ldots, (A_n, B_n)$. The assessment of the combined trees' regression function in Equation (10):

$$\bar{r}_n(A, D_n) = \mathbb{E}_{\ominus}[r_n(A, \ominus, D_n)] \tag{10}$$

where $\mathbb{E}$ has been the assumption or conditional mean for the specified posterior distribution. To create a forest of nonlinear individual trees, packing and feature stochastic are also used. When predictions are made by a committee rather than by individual trees, the results are more precise. References provide thorough configurations of a random forest classifier.

(a) XGBoost

The gradient tree boosting (GBT) development known as XGBoost (eXtreme Gradient Boosting) is indeed a tree ensemble machine learning technique. The forecast is described as follows at the time (or phase) r in Equation (11):

$$\hat{b}^{(r)} = \sum_{k=1}^{t} f_k(a_i) = \hat{b}_i^{(r-1)} + f_i(a_i) \tag{11}$$

where a_i would be the feature variable, also known as the input observation, which refers to the prior time values within the time series analysis set. Moreover, at time t, $f_i(a_i)$ seems to be the learner, which is typically a regression tree. The XGBoost framework employs a normalized objective function to protect the excessively of its training examples, as shown in Equation (12):

$$O^{(t)} = \sum_{k=1}^{n} l\left(\hat{b}, b\right) + \sum_{k=1}^{t} \Omega(f_i) \tag{12}$$

where t denotes the leaf count, Ω denotes the leaf score, and O denotes the regularization attribute. The leaf node splitting minimum loss value O is represented by the parameter. The research of Chen and Guestrin provides more information on the XGBoost framework and how it was put into practice.

3.4. Temporal Normalization

The goal of temporal normalization (TN) is to smooth out the high frequency of local and global elements of a hybrid signal. High-frequency elements and the low-frequency elements are each summarized separately here, using the two notations that are demonstrated in Equation (13):

$$C_{i,t}^{high} = C_{i,t}^{lh} C_t^{gh} \cdot C_{i,t}^{low} = C_{i,t}^{II} C_t^{gl}. \tag{13}$$

The reasonable inference that low-frequency element evolving charges are significantly slower than high-frequency element evolving rates is the foundation for the suitability of TN. Technically speaking, every low-frequency element over a period roughly equals a constant [33]. This presumption allows us to pertain TN to time sequence without its need for additional characteristics that characterize the frequency. Many real-world issues where the specialized frequency is unavailable can benefit from this characteristic. To achieve a desirable form for whom unique amounts could be determined from information, begin by expanding $C_{i,t}^{high}$ in Equation (14):

$$\begin{aligned} C_{i,t}^{high} &= \frac{C_{i,t}^{high} - EC_{i,t}^{highl} i}{\sigma C_{i,t}^{high} i + \epsilon} \sigma C_{i,t}^{highl} i + EC_{i,t}^{highl} i \\ &= \frac{C_{i,t}^{high} C_{i,t}^{low} - C_{i,t}^{low} EC_{i,t}^{highl} i}{C_{i,t}^{low} \sigma C_{i,t}^{highl} i + \epsilon} \sigma C_{i,t}^{highl} i + EC_{i,t}^{highl} i \\ &= \frac{C_{i,t} - EC_{i,t} C_{i,t}^{low}, i}{(\pm)\sigma Z_{i,t} Z_{i,t}^{low}, i + \epsilon} \sigma C_{i,t}^{highl} i + EC_{i,t}^{highl} i \\ &= \frac{C_{i,t} - EC_{i,t} C_{i,t}^{low}, i}{\sigma Z_{i,t} Z_{i,t}^{low}, i + \epsilon} (\pm) \sigma C_{i,t}^{highl} i + EC_{i,t}^{highl} i, \end{aligned} \tag{14}$$

where $C_{i,t}$ is perceptible, ϵ is a minor constant to maintain numerical stability, and the high-frequency impact mostly on ith time series over time is represented mostly by vectors $E\, C_{i,t}^{highl} i$ and $(\pm)\sigma\, C_{i,t}^{highl} i$, which can be estimated by a sequence of the learnable feature vector γ_i^{high} and β_i^{high} with a size of d_z, the values of $E\, C_{i,t} C_{i,t}^{low}, i$ and $\sigma\, C_{i,t} C_{i,t}^{low},\ i$ can be calculated by Equation (15):

$$\begin{aligned} EC_{i,t} C_{i,t}^{low}, i &\approx 1/\delta \sum_{t'=1}^{\delta} C_{i,t-t'+1}^{low} C_{i,t}^{low} \\ &\approx 1/\delta \sum_{t'=1}^{\delta} C_{i,t-t'+1}^{low} C_{i,t-t'+1}^{low} \\ &\approx 1/\delta \sum_{t'=1}^{\delta} C_{i,t-t'+1}^{low} - EC_{i,t} C_{i,t}^{low}, i^2 \end{aligned} \tag{15}$$

where δ is a time interval when the low-frequency element roughly stays continuous. For the sake of easiness, make several input appropriate actions in the task equitable. Research

can acquire the recognition of the high-frequency element by replacing the estimates of four non-observable parameters in Equation (16):

$$C_{i,t}^{high} = \frac{C_{i,t} - EC_{i,t}C_{i,t}^{low}, i}{\sigma C_{i,t}C_{i,t}^{low}, i + \epsilon}\gamma_i^{high} + \beta_i^{high} \tag{16}$$

Notably, TN and instance normalization (IN) for image data have a special connection in which style acts as a low-frequency element and material as a high-frequency element [33]. The research is novel because it identifies the source of TN within the perspective of MTS and pieces together TN step-by-step of its source.

3.5. Spatial Normalization

Refining local elements, which are made up of the native high-frequency component as well as the regional low-frequency element, is the goal of spatial normalization (SN) [34]. The first step in achieving this goal is to get rid of global elements, which are caused by factors corresponding to the time of day, day of the week, climate, etc. This also presents two notations for enumerating regional and global elements as given in Equation (17):

$$C_t^{global} = C_t^{gh}C_t^{gl}, C_{i,t}^{local} = C_{i,t}^{lh}C_{i,t}^{ll}. \tag{17}$$

The suitability of SN is also predicated on the idea that all time series will be affected similarly by global impacts. Here, it is significant to say that we do not also strictly need global effects to have identical effects on every time series. The specified local element could supplement those impacts which are not equally identified in each time series. This begins by extending $C_{i,t}^{local}$ to a representation where another term can either be delegated with trainable parameters or an approximation based on data in Equation (18):

$$\begin{aligned} C_{i,t}^{local} &= \frac{C_{i,t}^{local} - EC_{i,t}^{local}t}{\sigma C_{i,t}^{locall}t + \epsilon}\sigma C_{i,t}^{local}t + EC_{i,t}^{local}t \\ &= \frac{C_{i,t} - EC_{i,t}C_t^{global}, t}{(\pm)\sigma C_{i,t}C_t^{global}, t + \epsilon}\sigma C_{i,t}^{global}t + EC_{i,t}^{global}t \\ &= \frac{C_{i,t} - EC_{i,t}C_t^{global}, t}{\sigma C_{i,t}C_t^{global}, t + \epsilon}(\pm)\sigma C_{it}^{local}t + EC_{it}^{local}t \end{aligned} \tag{18}$$

where $C_{i,t}$ is easily observable and $(\pm)\sigma\left(C_{i,t}^{local}t\right)$ and $E\left(C_{i,t}^{local}t\right)$ have been estimated by two teachable vectors[2] (γ local and β local), the prediction of $EC_{i,t}C_t^{global}$, t and $\sigma\, EC_{i,t}C_t^{global}$, t could be derived from it so information in the following methods in Equation (19):

$$\begin{aligned} EC_{i,t}C_t^{global}, t &= \tfrac{1}{N}\sum_{j=1}^{N} C_{j,t}^{global}C_t^{global} \\ &\tfrac{1}{N}\sum_{j=1}^{N} c_{j,t} \\ \sigma^2 C_{i,t}C_t^{global}, t &= E[\left(C_{i,t} - EC_{i,t}C_t^{global}, t\right)\Big|C_t^{global}, t \\ &= \tfrac{1}{N}\sum_{j=1}^{N} C_{i,t} - EC_{i,t}C_t^{global}, t^2 \end{aligned} \tag{19}$$

This can get the composite illustration of local elements by putting the estimations of four non-observable factors into Equation (18), which reads as follows (Equation (20)):

$$C_{i,t}^{global} = \frac{C_{i,t} - EC_{i,t}C_t^{global}, t}{\sigma C_{i,t}C_t^{global}, t + \epsilon}\gamma local + \beta local \tag{20}$$

In the spatial domain, SN is TN's counterpart because it uses high-frequency elements to represent local elements and low-frequency elements to represent global elements. The

model takes into account fine-grained variability by removing the local and high-frequency elements with the actual signal, which is extremely helpful in time series prediction.

3.6. Learning and Forecasting

This designates $C^{(L)} \in \mathbb{R}^{N_l \times T_{in} \times d_z}$ as the result of the final residual block, in which every row of $C^{(L)} \in \mathbb{R}^{T_{in} \times d_z}$ stands for a different variable. Next, temporal accumulation is undertaken for each variable using a temporal pooling block. Depending on the issue being investigated, various pooling processes could be used, including max pooling and mean pooling [35]. In this instance, the pooling result is chosen to represent the entire signal by choosing the vector from the latest time slot. Finally, depending on the recognition acquired by a common fully connected layer, each creates different forecasting for each variable. The goal in the learning stage is to reduce the mean squared error (MSE) among the expected attributes and the standards obtained from the ground truth. Additionally, this goal can be maximized using the Adam optimizer.

4. Result and Discussion

4.1. Data Collection

To verify the efficacy of ST-Norm from various angles, perform comprehensive studies on three common data sources in this section by using Jupyter notebook. Using three real-world datasets, such as PeMSD7, BikeNYC, and Electricity, a framework can be verified, as well as statistics for each dataset and the correlating planned task settings. The principles in every dataset can be simplified to make training easier, and when testing is complete, the principles can be converted back to their original magnitude. In addition to SN and TN, an instance normalization (IN) control unit can be added. The sample size is four, and the batch pattern's input length is sixteen. For every DCC element's kernel size for such Wavenet backbone is set to 2, and its related dilation percentage is 2^i, where i have been the layer's index (counting from 0). Together, these settings allow Wavenet's output to recognize 16 input steps. Each DCC contains 16 hidden channels (d_z). To make the duration of a DCC output equivalent to 16, zero-stuffing can be extended to the left tail of an input. The Adam optimizer has a training set of 0.0001.

4.2. Evaluation Metrics

Then, we use the mean absolute error (MAE), mean squared error (RMSE), and mean absolute error (MSE) proportion to confirm the prototype (MAPE). For every model and every dataset, the test is repeated 10 times, and the average of outcomes is presented. Graph Wavenet has a similar architecture to MTGNN. The primary distinction would be that the former depends on a soft graph with a perpetual probability for every pair of nodes to be attached. This model designs segment-level correlation while capturing long-term correlations in time series information via the use of a consideration method, where its explanations and enquires are produced by the correlational convolution over a specific setting. MTGNN introduces a graph-learning device to create inter-variate connections. The diagram learning method particularly links every center node to its top k closest neighbors in a specified dimensional space. Wavenet is the primary design of MTGNN for sequential simulation.

A graph-learning component is also included in AGCRN to help develop inter-variable relationships. Additionally, it models the sequential connection for every time series using a customized RNN.

LSTNet consists of two parts: an LSTM with a supplemental skip correlation from an overtime component, and a traditional autoregressive design.

Graph Wavenet has a similar structure to MTGNN. The primary distinction would be that the distinction generates a soft chart with a constant likelihood that every pair of base stations is attached.

TCN's architecture is similar to Wavenet's, with the exception that every residual block's nonlinear transition is composed of two rectified linear units (ReLU).

This is also evaluated by using TCN and Inductor function when STN is used similarly before every layer's causal convolution procedure.

4.3. Ablation Study

This creates numerous variations as regards verifying the efficacy of SN and TN. This will also test a combination that includes both STN and a graph-learning module to see if it enhances STN. Since the normal Wavenet backbone is present in all of the variations, Wavenet is left out of the phrase for simplicity.

On each of the three datasets, these variants were tested, and Table 1 presents the overall findings. SN and TN both relate to the improvement. Moreover, STN's achievement slightly improves with a responsive graph-learning device. This shows that STN significantly outperforms and replaces the graph-learning device.

Table 1. Study of ablation.

		STN	***GSTN***	***SN***	***TN***	***Graph***	***Vanilla***
P	*MAE*	2.46	2.48	2.64	2.65	2.98	3.45
	RMSE	5.19	5.22	2.45	4.65	7.35	6.32
	MAPE	19.5	19.3	18.5	18.7	21.5	24.9
B	*MAE*	2.76	2.84	3.04	2.88	3.45	3.95
	RMSE	5.14	5.23	5.33	5.28	5.67	6.54
	MAPE	5.85	5.96	6.34	6.08	7.35	8.34
E	*MAE*	17.6	18.7	20.5	20.87	21.54	22.3
	RMSE	37.5	39.65	39.54	38.32	44.87	43.8
	MAPE	15.3	15.66	14.32	15.32	15.23	15.45

4.4. Model Optimization

To identify any underfitting or overfitting, learning shapes of prediction accuracy upon that train and validation datasets have been used. Each applicant's model's effectiveness was represented as a loss function and plotted against the epochs for both the training and validation sets. One could quickly determine whether the model might result in increased variation (overfit) or bias by correlating and examining the patterns of plotted loss features (underfit). K-fold cross-validation is utilized when a model has several hyperparameters because there were also too several more possible mixtures of hyperparameter attributes. A manual method should, therefore, be avoided because it is time-consuming. This method divides the training set into k smaller sets. Several of the k-folds is processed in the manner described below:

- A structure is trained by utilizing $k - 1$ of folds as training information;
- The architecture was authorized on a residual data portion.

The two stages above were recurring k-times by utilizing another data portion for validations. By computing the mean calculated for the k steps, the effectiveness identified by k-fold cross-validation is obtained. Although this method may be computationally time affordable, it does not necessitate as much information as repairing a specialized validation set. Other methods exist that might be slightly different, but usually adhere to the same fundamentals.

4.5. XGBoost

Several parameters could be given specific values to characterize XGBoost models. They must be chosen towards optimizing the performance of the approach on a particular dataset and in a manner that guards against both underfitting and overfitting as well as unnecessary difficulty. These parameters also include effect, learning algorithm, lambda,

alpha, and the number of boosting repetitions. In XGBoost, the amount of shaped consecutive trees is referred to as the number of boosting iterations. The largest number of splits is determined by the tree's maximum depth; a high maximum depth can lead to overfitting. Before growing trees, random subsampling corresponds to a particular ratio of a training dataset within every iteration. The optimization method is stronger by using a learning rate, which essentially lessens the effect of every individual tree and allows future trees to enhance the framework. Increases in the variables make the model extra conservative because they are normalization terms for weight training. The values for each parameter are obtained by applying a grid search algorithm and a 10-fold cross-validation procedure. In total, there were 500 boosting repetitions, 25 maximum tree depths, and 1 (0.8) for the subsample ratio of the training dataset, 0.5 for the subsample ratio of the columns, and $\lambda = 1$, $\alpha = 0.2$, and 0.1 for the learning rate.

4.6. Analysis of Hyperparameter Configurations

We investigated the impact of various hyperparameter configurations in the suggested modules. The percentage of historical steps entered into the model, the dimensions of a DCC kernel, the batch dimensions, and the element of hidden channels (d_z) are the four hyperparameters that need a manual process set by professionals. The study's findings are shown in Figure 5, which can infer that STN not only improves effectiveness, but also multiplies the achievement stability in various hyperparameter configurations.

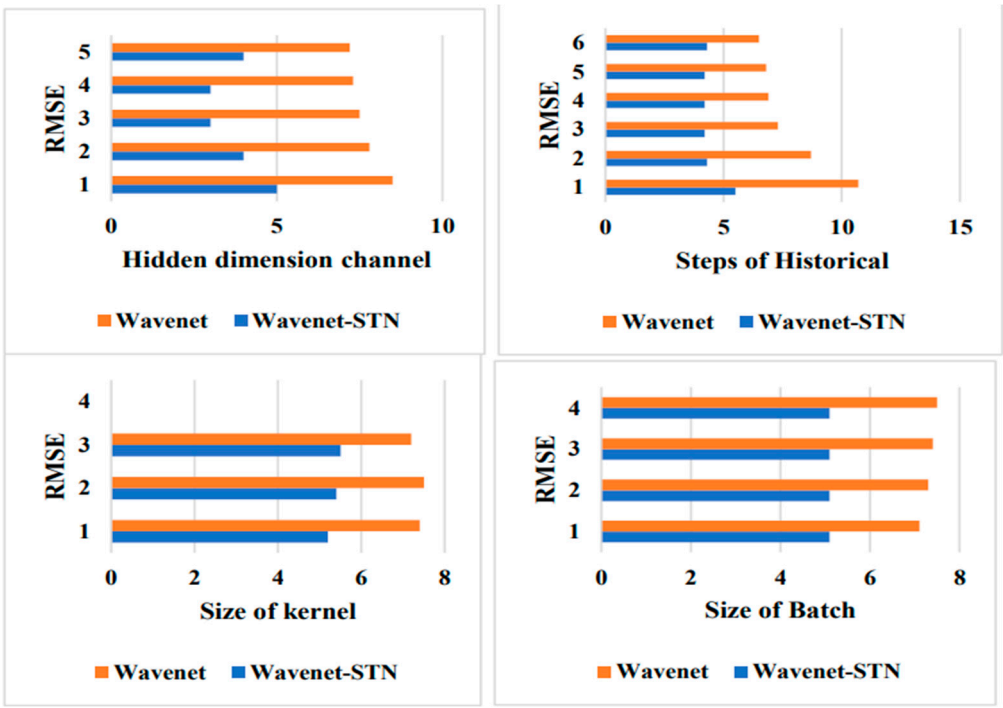

Figure 5. Analysis of hyperparameters.

They can be applied to the raw input data to see if they reduce the problems raised and to show how TN and SN redefine the extracted features. The original quantity is against both the temporally normalized amount and the spatially normalized quantity. There are differences among regions and days within the pairwise connection between the make sure and the temporally normalized quantity as well as among the original measure and the spatially normalized quantity. Insufficient SOTA methods suggest that to improve the local element, various time series should be made to have a mutual relationship with one another. In essence, they highlight the local element of individual time series by contrasting a pair of time series that have similar global elements over time. Multiple connections reflect the individuality of every time series are produced, for instance, by contrasting the

three-time series within a single time series (referred to as an anchor). Dissimilar time series might need to be multiplied along various anchors because it is frequently unknown which ones are eligible anchors. These approaches use a graph-learning component to investigate every potential couple of time series to mechanically recognize the anchor for each time series. Here, $O(TN^2)$ refers to the computational complexity. The method's normalization modules, in contrast to other ones that have been suggested in this field, only call for $O(TN)$ operations.

4.7. Model Comparison

The MASE of every method is established for various lead times to demonstrate their effects on various time series predictions. MASE for every model is then calculated for every lead time. Table 2 shows the proposed model's comparison with existing methods. Figure 6 demonstrates the competitive performances of Multi-variate RNN, LSTM, MLP, and SVM for 5–15 min early time series prediction. However, their prediction accuracy declines significantly as lead time lengthens. Given the strong correlation, a simulation model such as MLP can be supplemented by using both temporal and spatial data as input. As a result, predicting accuracy could be increased. Figure 6 demonstrates that MLP performs better than other methods for every lead time. This happens because MLP, which has a deep architecture having nonlinear properties, can accurately capture and recreate the entire target, whereas other approaches cannot do so, owing to their linear characteristics.

Table 2. Performance comparison with MASE.

Leading Time/ Methods	MASE			
	RNN	LSTM	SVM	MLP
5	98.1	85.3	71.3	68.2
10	94.8	80.1	70.3	64.5
15	96.7	90.2	68.6	56.3
25	98.1	91.4	73.9	78.2
30	90.1	86.3	77.3	60.6

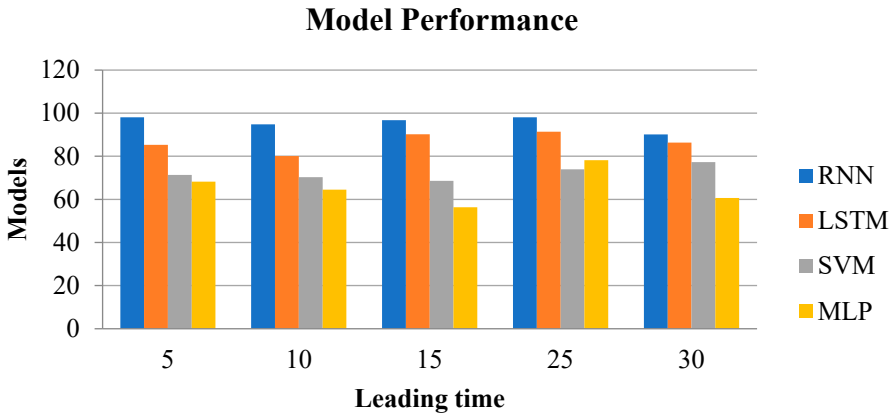

Figure 6. Comparison of the proposed method.

4.8. Computation Time Comparison

Table 3 lists the times for training (on the training set) as well as predicting (on the testing set) for every approach. The training time for the many multi-variate algorithms that have been suggested, including MLP, RNN, LSTM, and SVM, is the total of the training times, as well as the prediction time, determined in the same manner. The persistent method's learning and forecasting times are 0 and are, therefore, not listed. As seen in Table 3, because periodic retraining and refreshing of forecasts are eliminated, the time effectiveness is comparable to other approaches while training spatial-temporal predictions.

The MTS spatial and temporal prediction is taught live by incremental learning, which saves time by avoiding periodic model retraining and refreshing.

Table 3. Time computation comparison.

Leading Method/Methods	RNN		LSTM		SVM		MLP	
Process/(s)	**Train**	**Pred**	**Train**	**Pred**	**Train**	**Pred**	**Train**	**Pred**
5	1545	4.21	22,540	20.09	17,680	0.38	134	0.25
10	1445	3.82	20,020	21.13	16,560	0.33	185	0.22
15	1685	3.45	37,140	22.55	15,580	0.34	182	0.34
20	1510	3.41	25,860	23.42	14,230	0.34	225	0.32
25	1415	3.30	39,770	21.01	17,380	0.36	261	0.37
30	1500	3.29	22,590	24.10	12,730	0.25	228	0.59

5. Conclusions

This paper proposes a novel method for factorizing MTS data. We suggest spatial and temporal normalization following factorization, which improves the local and high-frequency components of the MTS data, respectively. Due to the nonlinear nature of demand variations, current research demonstrates that statistical methods lack predictive value in real-world circumstances. In particular, predictions beyond a few hours may be imprecise when employing these algorithms. As evidenced by the multiple plots in the current edition, these shifts occur after a few hours, particularly when the demand varies drastically. The results of the experiment illustrate the capability and performance of these two components. However, this study has significant limitations, including controlling the modeling process through improved machine learning model parameters and locating appropriate variables for taking the input from the models into consideration. Future studies may take into account the use of various hybrid frameworks with optimization techniques as a unique addition to spatial and temporal prediction.

Author Contributions: Conceptualization, A.M.P. and C.Y.; methodology, A.M.P. and C.Y.; software, G.K.A.; validation, A.M.P. and C.Y.; formal analysis, A.M.P. and C.Y.; investigation, A.M.P.; resources, A.M.P.; data curation, A.M.P.; writing—original draft preparation, A.M.P.; writing—review and editing, A.M.P., T.B.O. and A.M.; visualization, G.K.A.; supervision, C.Y.; funding acquisition, C.Y. All authors have read and agreed to the published version of the manuscript.

Funding: The National Natural Science Foundation of China under Grant No. 61873004.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fildes, R.; Nikolopoulos, K.; Crone, S.F.; Syntetos, A.A. Forecasting and operational research: A review. *J. Oper. Res. Soc.* **2008**, *59*, 1150–1172. [CrossRef]
2. Forni, M.; Hallin, M.; Lippi, M.; Reichlin, L. The Generalized Dynamic Factor Model. *J. Am. Stat. Assoc.* **2005**, *100*, 830–840. [CrossRef]
3. Perez-Chacon, R.; Talavera-Llames, R.L.; Martinez-Alvarez, F.; Troncoso, A. Finding Electric Energy Consumption Patterns in Big Time Series Data. In *Distributed Computing and Artificial Intelligence, 13th International Conference*; Omatu, S., Semalat, A., Bocewicz, G., Sitek, P., Nielsen, I.E., García García, J.A., Bajo, J., Eds.; Springer International Publishing: Cham, Switzerland, 2016; Volume 474, pp. 231–238. [CrossRef]
4. Galicia, A.; Torres, J.F.; Martínez-Álvarez, F.; Troncoso, A. Scalable Forecasting Techniques Applied to Big Electricity Time Series. In *Advances in Computational Intelligence*; Rojas, I., Joya, G., Catala, A., Eds.; Springer International Publishing: Cham, Switzerland, 2017; Volume 10306, pp. 165–175. [CrossRef]
5. Hyndman, R.; Koehler, A.B.; Ord, J.K.; Snyder, R.D. *Forecasting with Exponential Smoothing: The State Space Approach*; Springer Science & Business Media: West Lafayette, IN, USA, 2008.
6. Makridakis, S.; Hyndman, R.J.; Petropoulos, F. Forecasting in social settings: The state of the art. *Int. J. Forecast.* **2020**, *36*, 15–28. [CrossRef]

7. Geng, X.; Li, Y.; Wang, L.; Zhang, L.; Yang, Q.; Ye, J.; Liu, Y. Spatiotemporal Multi-Graph Convolution Network for Ride-Hailing Demand Forecasting. *Proc. Conf. AAAI Artif. Intell.* **2019**, *33*, 3656–3663. [CrossRef]
8. Ding, D.; Zhang, M.; Pan, X.; Yang, M.; He, X. Modeling Extreme Events in Time Series Prediction. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 1114–1122. [CrossRef]
9. Piccialli, F.; Giampaolo, F.; Prezioso, E.; Camacho, D.; Acampora, G. Artificial intelligence and healthcare: Forecasting of medical bookings through multi-source time-series fusion. *Inf. Fusion* **2021**, *74*, 1–16. [CrossRef]
10. Fathian, F.; Mehdizadeh, S.; Sales, A.K.; Safari, M.J.S. Hybrid models to improve the monthly river flow prediction: Integrating artificial intelligence and non-linear time series models. *J. Hydrol.* **2019**, *575*, 1200–1213. [CrossRef]
11. Gul, E.; Safari, M.J.S.; Haghighi, A.T.; Mehr, A.D. Sediment transport modeling in non-deposition with clean bed condition using different tree-based algorithms. *PLoS ONE* **2021**, *16*, e0258125. [CrossRef] [PubMed]
12. Bontempi, G.; Ben Taieb, S. Conditionally dependent strategies for multiple-step-ahead prediction in local learning. *Int. J. Forecast.* **2011**, *27*, 689–699. [CrossRef]
13. Ben Taieb, S.; Bontempi, G.; Atiya, A.F.; Sorjamaa, A. A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. *Expert Syst. Appl.* **2012**, *39*, 7067–7083. [CrossRef]
14. Kolda, T.; Bader, B.W. Tensor Decompositions and Applications. *SIAM Rev.* **2009**, *51*, 455–500. [CrossRef]
15. De Silva, A.; Hyndman, R.J.; Snyder, R. The vector innovations structural time series framework: A simple approach to multivariate forecasting. *Stat. Model.* **2010**, *10*, 353–374. [CrossRef]
16. Bai, S.; Kolter, J.Z.; Koltun, V. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv* **2018**, arXiv:1803.01271. [CrossRef]
17. Oord, A.V.D.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; Kavukcuoglu, K. WaveNet: A Generative Model for Raw Audio. *arXiv* **2016**, arXiv:1609.03499. [CrossRef]
18. Liu, C.-T.; Wu, C.-W.; Wang, Y.-C.F.; Chien, S.-Y. Spatially and Temporally Efficient Non-local Attention Network for Video-based Person Re-Identification. *arXiv* **2019**, arXiv:1908.01683. [CrossRef]
19. Safari, M.J.S. Hybridization of multivariate adaptive regression splines and random forest models with an empirical equation for sediment deposition prediction in open channel flow. *J. Hydrol.* **2020**, *590*, 125392. [CrossRef]
20. Yu, H.-F.; Rao, N.; Dhillon, I.S. Temporal Regularized Matrix Factorization for High-dimensional Time Series Prediction. *Adv. Neural Inf. Processing Syst.* **2016**, *29*, 9.
21. Jiang, R.; Cai, Z.; Wang, Z.; Yang, C.; Fan, Z.; Chen, Q.; Tsubouchi, K.; Song, X.; Shibasaki, R. DeepCrowd: A Deep Model for Large-Scale Citywide Crowd Density and Flow Prediction. *IEEE Trans. Knowl. Data Eng.* **2021**. [CrossRef]
22. Fan, C.; Zhang, Y.; Pan, Y.; Li, X.; Zhang, C.; Yuan, R.; Wu, D.; Wang, W.; Pei, J.; Huang, H. Multi-Horizon Time Series Forecasting with Temporal Attention Learning. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 2527–2535. [CrossRef]
23. Anderer, M.; Li, F. Hierarchical forecasting with a top-down alignment of independent-level forecasts. *Int. J. Forecast.* **2022**, *38*, 1405–1414. [CrossRef]
24. Lin, S.; Xu, F.; Wang, X.; Yang, W.; Yu, L. Efficient Spatial-Temporal Normalization of SAE Representation for Event Camera. *IEEE Robot. Autom. Lett.* **2020**, *5*, 4265–4272. [CrossRef]
25. Wang, J.-H.; Lin, G.-F.; Chang, M.-J.; Huang, I.-H.; Chen, Y.-R. Real-Time Water-Level Forecasting Using Dilated Causal Convolutional Neural Networks. *Water Resour. Manag.* **2019**, *33*, 3759–3780. [CrossRef]
26. Zhang, X.; You, J. A Gated Dilated Causal Convolution Based Encoder-Decoder for Network Traffic Forecasting. *IEEE Access* **2020**, *8*, 6087–6097. [CrossRef]
27. Zhao, W.; Wu, H.; Yin, G.; Duan, S.-B. Normalization of the temporal effect on the MODIS land surface temperature product using random forest regression. *ISPRS J. Photogramm. Remote Sens.* **2019**, *152*, 109–118. [CrossRef]
28. Botalb, A.; Moinuddin, M.; Al-Saggaf, U.M.; Ali, S.S.A. Contrasting Convolutional Neural Network (CNN) with Multi-Layer Perceptron (MLP) for Big Data Analysis. In Proceedings of the 2018 International Conference on Intelligent and Advanced System (ICIAS), Kuala Lumpur, Malaysia, 13–14 August 2018; pp. 1–5. [CrossRef]
29. Shi, X.; Li, Y.; Yang, Y.; Sun, B.; Qi, F. Multi-models and dual-sampling periods quality prediction with time-dimensional K-means and state transition-LSTM network. *Inf. Sci.* **2021**, *580*, 917–933. [CrossRef]
30. de Myttenaere, A.; Golden, B.; Le Grand, B.; Rossi, F. Mean Absolute Percentage Error for regression models. *Neurocomputing* **2016**, *192*, 38–48. [CrossRef]
31. Aschner, A.; Solomon, S.G.; Landy, M.S.; Heeger, D.J.; Kohn, A. Temporal Contingencies Determine Whether Adaptation Strengthens or Weakens Normalization. *J. Neurosci.* **2018**, *38*, 10129–10142. [CrossRef]
32. Diao, Z.; Wang, X.; Zhang, D.; Liu, Y.; Xie, K.; He, S. Dynamic Spatial-Temporal Graph Convolutional Neural Networks for Traffic Forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 890–897. [CrossRef]
33. Makridakis, S.; Spiliotis, E.; Assimakopoulos, V. Statistical and Machine Learning forecasting methods: Concerns and ways forward. *PLoS ONE* **2018**, *13*, e0194889. [CrossRef]

34. Safari, M.J.S.; Arashloo, S.R. Sparse kernel regression technique for self-cleansing channel design. *Adv. Eng. Inform.* **2021**, *47*, 101230. [CrossRef]
35. Mohammadi, B.; Guan, Y.; Moazenzadeh, R.; Safari, M.J.S. Implementation of hybrid particle swarm optimization-differential evolution algorithms coupled with multi-layer perceptron for suspended sediment load estimation. *CATENA* **2021**, *198*, 105024. [CrossRef]