mTSeer: Interactive Visual Exploration of Models on Multivariate Time-series Forecast

Ke Xu
kexu@nyu.edu
New York University
New York

Jun Yuan
junyuan@nyu.edu
New York University
New York

Yifang Wang
ywangjh@connect.ust.hk
The Hong Kong University of Science and Technology
Hong Kong

Claudio Silva
csilva@nyu.edu
New York University
New York

Enrico Bertini
enrico.bertini@nyu.edu
New York University
New York

ABSTRACT

Time-series forecasting contributes crucial information to industrial and institutional decision-making with multivariate time-series input. Although various models have been developed to facilitate the forecasting process, they make inconsistent forecasts. Thus, it is critical to select the model appropriately. The existing selection methods based on the error measures fail to reveal deep insights into the model's performance, such as the identification of salient features and the impact of temporal factors (e.g., periods). This paper introduces mTSeer, an interactive system for the exploration, explanation, and evaluation of multivariate time-series forecasting models. Our system integrates a set of algorithms to steer the process, and rich interactions and visualization designs to help interpret the differences between models in both model and instance level. We demonstrate the effectiveness of mTSeer through three case studies with two domain experts on real-world data, qualitative interviews with the two experts, and quantitative evaluation of the three case studies.

CCS CONCEPTS

• **Human-centered computing** → **Visualization**; • **Computing methodologies** → **Machine learning**; **Model development and analysis**.

KEYWORDS

visualization; machine learning; multivariate time series; model evaluation; forecasting; feature extraction

ACM Reference Format:
Ke Xu, Jun Yuan, Yifang Wang, Claudio Silva, and Enrico Bertini. 2021. mTSeer: Interactive Visual Exploration of Models on Multivariate Time-series Forecast. In *CHI Conference on Human Factors in Computing Systems (CHI '21), May 8–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3411764.3445083

CHI '21, May 8–13, 2021, Yokohama, Japan

ACM ISBN 978-1-4503-8096-6/21/05…$15.00
https://doi.org/10.1145/3411764.3445083

1 INTRODUCTION

Time-series forecasting is the extrapolation of future values of a series based on the historical information that is related to that series. It is becoming increasingly indispensable as the knowledge derived from time series forecasting can provide the industrial and scientific fields with crucial information for their decision-making process [43]. In today's data-driven world, many real-world time series generated by applications such as finance, weather, biology, industry demand have multiple variables or attributes. For example, gold price forecasting apply RNNs on sequences of multivariate data where each dimension represents an individual feature with semantic meaning such as inflation or deposit rates. Various forecasting algorithms have been developed with these multivariate time series as input [23]. These methods can be categorized into: the classical linear regression methods [10] and non-linear regression methods [14], and the modern techniques based on machine learning and deep neural networks [62]. However, different models that contain different parameter sets and fundamental assumptions may be specialized for different feature spaces in dataset and application domains. In addition, these models often fail to consider the characteristics of time-series problems in their model development process, such as the periodicity and spikes in the time series and the requirement to predict multiple time steps into the future. Thus, given a time-series dataset, it is important for domain experts to evaluate, compare, and select existing forecasting models in a steerable manner using different model settings.

Recent studies have addressed the problem of evaluating multivariate forecasting models using various statistical analysis results. The results are often represented by accuracy/error measures (e.g., the root mean squared error) computed on the testing period of the time series. Moreover, the cross-validation method is used for evaluating machine learning and other auto-regression forecasting methods on the random partitions of the entire dataset [7]. Such measurements are helpful and we retain some of them in our work. However, issues remain when dealing with multivariate time series with these methods. First, they cannot assess whether the quantity of used variables has affected the accuracy. Although some works

take the correlations between different variables into considerations [13, 59], they still fail to show the relationships between input features and output forecasts directly. Second, most of them only provide the overall model-level performance comparisons and lack the details in the instance level. It is a common requirement for users or domain experts to inspect some specific forecasting results to help them understand the model better and make reasonable decisions. Finally, these accuracy measures do not involve the domain experts to steer, explore and interpret the models with their insights and domain-knowledge, which in turn facilitates their model evaluation and selection by providing related information.

For these reasons, it is necessary to develop interactive visualization tools that make multivariate time-series forecasting models and their evaluations more explainable to domain experts. This has driven the development of some visualization systems for time-series model explanations (e.g., MultiRNNExplorer [54]) and model selection (e.g., TiMoVA [8]). However, they have a limited focus on specific forecasting models like RNN, or merely deal with the univariate time series [58]. By contrast, the works for more general models ([35, 41, 50]) usually focus on the feature importance, having no comparison between models. Finally, none of these visualization techniques take the inherent patterns (e.g., periodicity) of time series or the anomalous forecasts into account in the model implementation and evaluation process.

To solve these issues, we introduce mTSeer, a visual analytics system developed to help domain experts explore and evaluate multivariate time-series forecasting models in a steerable and interpretable manner with both model and instance level information. Major research contributions include:

- **System.** We propose an integrated visual analytics system for a user-guided, steerable exploration and evaluation of multivariate time-series forecasting models. We formulate the design requirements through cooperation with experts in both machine learning and visualization. Candidate forecasting models are integrated with periodicity detection and spike detection algorithms to make a more reasonable time-series model estimation. The visualization and interaction designs support the visual exploration of models from the overall comparison, the correlation analysis to the particular feature importance and instance inspection.
- **Visualization and Interactions.** We propose a set of visualization and interaction designs to facilitate users' evaluation of forecasting models in both model level and instance level. Specifically, we combine the stacked bar chart, line chart and a responsive tooltip to simultaneously show the forecast results and the feature attributions versus time. The differences between models are presented with a line glyph. The variety of parameter settings improves the uncertainty explanation of data and models. Rich interaction designs and side views are provided to promote the model evaluation.
- **Evaluation.** The effectiveness of mTSeer is demonstrated in multiple forms of evaluation. We first describe how mTSeer works through three case studies conducted by two domain experts with real-world datasets, and collect feedback from the two experts after the qualitative expert interviews. For each case study, we then conduct a accuracy-based quantitative evaluation of the forecasting models' performance when they are trained in different input lengths and predicted with different steps ahead. All the evaluations validate the effectiveness of mTSeer in evaluating model performances on multivariate time-series forecasting.

2 RELATED WORK

In this section, we provide an overview of the papers that are most related to our work, which includes: techniques for time-series forecast and their evaluation, visualizations for time series and forecast models.

2.1 Time-series Forecasting and Evaluation

Various time-series forecast related techniques have been developed in many research communities and application domains for several decades, including the classical statistic methods, machine learning approaches for time series forecasting and their evaluation methods.

Time-series Forecasting Techniques. The time-series forecasting is aimed to predict future values based on historical data, which plays a key role in data-driven decision making in most business and medical analysis. The traditional methods can be broadly summarized into three categories based on the modeling principles: the auto-regression based methods [9, 57], the exponential smoothing (ES) methods [32], and the machine learning and deep learning forecasting methods [3, 19, 19, 46]. However, most of them are only designed for univariate time-series forecasting, and do not take into account other exploitable time series with related attributes in the same dataset. Thereby, multivariate time-series forecasting models have been developed via extending these traditional models [29]. For example, Mukherjee et al. [47] presented the Vector Error Correction Model (VECM) to predict the stock market, and Kane et al. [30] applied Random Forest (RF) to predict the avian influenza H5N1 outbreaks. More recently, the breakthrough of deep neural network and their variants have been applied for multivariate forecasting, which includes different types of Recurrent Neural Networks (RNNs) like LSTM [26], the multi-layer perceptron (MLP) as an extension of the linear regression models [1], the convolutional Neural Networks (CNN) [62] and WaveNet [49]. There are also some hybrid methods that combine different types of deep learning techniques. All of the techniques discussed are not comprehensive but represent different approaches. Our system implements some of the most representative algorithms from different categories as the baseline algorithms.

Time-series Forecasting Techniques Evaluation. A variety of evaluation methods have been proposed to compare the performance of different time-series forecasting models. Traditionally, the accuracy-based methods are used to evaluate the performance of timse-series predictors. The most basic method is mean squared error (MSE) that measures the average squared difference between the actual and predicted temporal values [5, 27]. Such evaluation ways have been enhanced and summarized into different categories: (1) the scale-dependent measures such as the root MSE (RMSE) [12, 34, 45] , (2) the percentage errors that overcome scale-dependency such as the mean absolute percentage error (MAPE) [6, 53], and (3) the relative estimation compared with a benchmark method like the Theil's inequality coefficient [17, 40]. In addition to the accuracy-based metrics, cross-validation is also popular for

evaluating regression and classification methods [4]. Furthermore, these methods have been improved by taking the characteristics of time series into account, such as calculating MSE based on the turning points of time series [24], conducting cross-validation with stationary time series [15] and blocked subsets [56]. A wider perspective considers both the accuracy and the correlation between different varieties. For example, Carmack et al. [13] presented the far casting cross-validation (FCCV) by defining a neighborhood radius to remove dependent data in forecasting. E. Toth et al. [59] used a correlation coefficient between different varieties to conduct a multi-step rainfall prediction. A more advanced work derives non-parametric risk bounds on the expected inaccuracy of the predictions to help select forecasting models [44]. Although these methods are able to evaluate different multivariate forecasting models, they fail to provide the explanations of such comparisons and assessments, and also lack a deep understanding of the importance of different features in the forecasting process. Thus, the method proposed in our work offers a more comprehensive and interpretative evaluation from the overall correlations of different models to the detailed comparisons in the feature context.

2.2 Visualization for Temporal Data

Time-series data visualization has gained momentum in recent years due to its wide application in many fields [55]. A variety of surveys have summarized the state-of-the-art visualization methods to display the temporal patterns of time-series data based on their feature space. For example, Müller et al. [48] first discussed the general visual representations of time-series data based on whether they are time-dependent or not. Afterwards, Aigner et al. [2] summarized the visual methods for analyzing temporal data into three aspects based on the characteristics of time series, i.e., ordinal, discrete, and continuous time series. In this paper, we only discuss the continuous visualization techniques as they are most relevant to our work.

Specifically, the visual representation of multivariate time series can be categorized into four types [16]: (1) Line charts. This is the most common visualization form for time series. For multidimensional time series, collections of line charts can be overlaid in the same space to convey global trends or to make local comparisons in the data. A popular example is Storyline [33], which usually portrays the temporal dynamics of social interactions in a single image plane. However, bundling multiple lines in the same space will cause visual clutter issues. (2) Stacked graphs. This is another approach, where individual line charts are accumulated at each point [11]. Projects such as NameVoyager [61] and sense.us [22] implemented the stacked graphs with demographic data. Although stacked graphs could display the general trend of time series, they limit the comparison between individual series and the understanding of space between curves. (3) Horizon charts. It can compare individual time series by dividing and layering filled line charts, which were first proposed by Saito et al. [51] and then optimized in terms of graphical scalability and perception by Few et al. [31] and Heer et al. [21]. But this method cannot provide exact comparison when values change drastically; (4) Glyphs-based method. As an extension or abstraction of traditional designs for time-series visualization, this method is useful for temporal performance summary and comparison [18]. For example, MatchPad [39] allows a scale-adaptive glyph to analyze real-time sports performance.

However, none of these visualization techniques are designed for time-series forecasting problems that have to be analyzed with both the input data and the forecast results. In our work, we enhance the line chart with a specifically ordered stack graphs to display the forecasts in the context of input feature importance.

2.3 Visual Exploration for Forecasting Models

Visualization has recently been used to help humans in the model explanation, evaluation and selection process [20, 52]. On the one hand, a variety of visual analytics techniques have been developed to help users understand forecasting models. For example, RetainVis [37] provided a visual analytics tool to increase the interpretability of RNN model and to improve users' exploration of EMR data in the context of prediction tasks. Most relevantly to our work, MultiRNNExplorer [54] used visualization designs to interpret RNNs in high-dimensional time-series forecasts. However, these two systems only analyzed the internal working mechanism of prediction models, which are limited in capability to explain general forecasting models. By contrast, sensitivity analysis methods focus on the general relationships between input features and output prediction. For example, two recent works, LIME [50] and SHAP [41], calculate feature importance from a local perspective. In addition, a more comprehensive visualization system, Prospector[36], provided an interactive partial dependence diagnostics to estimate the features effect to the forecasts. However, the aforementioned visualization methods were merely for model explanation rather than model evaluation. Thus, on the other hand, other visualization techniques have been proposed to facilitate the comparison of forecasting models. For example, Bögl et al. [8] developed TiMoVA for a detailed comparison of ARIMA models based on different parameter settings. Dong et al. [58] proposed DFSeer to facilitate the model comparison and selection for demand forecasting with historical data. The problem of these model evaluation systems is that they only deal with the univariate time series, and not consider the characteristics of time series in the model evaluation process.

To the best of our knowledge, mTSeer is the first visual exploration system that compares and evaluates different multivariate time-series forecasting models in a steerable manner, which takes the important concepts like the period and the forecast steps in the model implementation process. Rich visualization and interaction designs are also introduced to help users interactively interpret and evaluate the forecasting models in both model level and instance level.

3 SYSTEM OVERVIEW

mTSeer was developed as a part of a two-year explainable artificial intelligence project. This project is aimed at developing automated model discovery systems and integrating visual analytics techniques to allow interactive data augmentation and visual model selection. We hold weekly meetings with two experts in machine learning and visualization for two months, at which a variety of design requirements were clarified and preliminary designs were discussed via the iterative user-centered design process. Below we

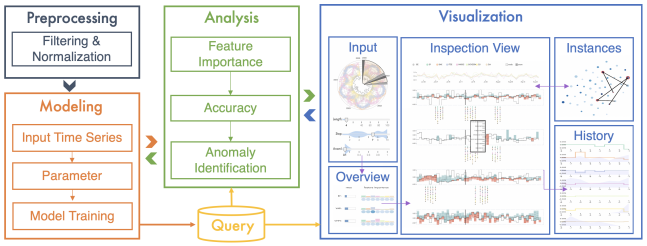


Figure 1: mTSeer system overview and data processing pipeline.

describe the most critical requirements (**R1–R4**) to guide the design of our system.

- **R1 Construct models for multivariate time series forecasts.** The system should provide effective model construction methods to help users process the large-scale multivariate time-series data, as well as set the forecast models according to the characteristics of temporal data, such as the length of input time series.
- **R2 Evaluate models in multifaceted ways.** Having a comprehensive evaluation of models is essential for model comparison and selection. Thus, the system should facilitate the model evaluation from the overall summarization (model level) of model performance to the detailed analysis at each timestamps (instance level), as well as identify the outliers that have irregular forecast results.
- **R3 Interpret the exploration results from different perspectives.** The visualization should be able to present model behaviors with a full range of information including the raw data, the critical time steps, the feature importance and the correlation between different models, to help users understand "when and why" some forecasting models are good or not with a given dataset.
- **R4 Enable an easy and interactive evaluation.** To provide efficient exploration and evaluation across models, it requires to incorporate flexible interactions that help users quickly navigate the input, the forecast results and the interpretations with substantial details.

We have designed mTSeer, an interactive model evaluation framework of multivariate time-series forecasting models, to meet the above requirements. Fig. 1 illustrates the system architecture and the model analysis pipeline, which contains four major modules: (1) the data preprocessing module, (2) the model construction module, (3) the analysis module, and (4) the visualization and interaction module.

In particular, in the data preprocessing module, the multivariate time series are filtered (e.g., missing values or useless features), transformed (e.g., categorical values) and normalized for the following training and testing (**R1**). The model construction module specifies the training and testing periods, the parameters for the candidate forecasting models, and then trains the model to get the forecasting results (**R1**). The analysis module computes a range of measurements for model evaluation, which consist of the feature importance for different models at different timestamps, the accuracy estimation and the variance of forecasts. An outlier detection algorithm is also utilized to identify the anomalous forecasts for each model (**R2, 3**). The visualization and interaction module employs several coordinated views to support a comprehensive visual evaluation and interpretation of forecasts in a multi-level context (**R3**). Various interaction designs are created to support a quick and responsive exploration of the model performance (**R4**). All these modules work together to form a steerable mechanism that enables an effective procedure to promote the information-seeking space.

4 EVALUATION ON MULTIVARIATE TIME-SERIES FORECAST

In this section, we describe the techniques used in mTSeer to conduct a steerable model evaluation on multivariate time series forecast (Fig. 2), which mainly consists of two parts: the model construction part for input setting and multivariate time-series forecast, and the model interpretation part. The detailed procedures are introduced below.

4.1 Model Description

The input time series of multivariate forecasting models (Fig. 2(1)) is a sequence $\boldsymbol{X_n} = \{x_{n-L}, x_{n-(L-1)}, ..., x_{n-1}, x_n\}$, where L is the pre-defined lagged value/input length, x_n is the multi-dimensional feature vector at time step n with each feature dimension denoted as $x_n^f \in \mathbb{R}$. The multivariate forecasting model predicts the value at timestamps equal to or greater than n. We make multi-step forecasts with output denoted as $\boldsymbol{y_n} = \{x_{n+1}, x_{n+2}, ..., x_{n+t}\}$ (Fig. 2(3)), where t is the user-defined forecast steps into the future. The length L can be adjusted according to its latent period, which could show the influence of periodicity as a salient temporal factor to the time-series forecasts. Moreover, the reason to conduct multi-step forecasting is based on the practical requirement of most real-world time-series forecasting tasks.

4.1.1 Candidate Models. Five representative time-series forecasting models have been selected as candidate techniques for evaluation (Fig. 2(2)), which covers three typical methods: linear, and non-linear regression (statistical) methods, as well as deep learning methods. The details are as follows:

Vector Auto-Regressive Model (VAR) [42] from linear regression modeling algorithms, is a stochastic process that each variable captures the linear interdependencies of the past values of itself and the past values of all the other variables: $\boldsymbol{y_n} = A_1 x_n + ... + A_p x_{n-p} + u_n, u_n \sim \mathcal{N}(0, \sum_u)$, where A_i is a time-invariant coefficient matrix and u_n is a k-vector of error terms. The input length L can be adjusted by the lagged order p in VAR.

Random Forest Model (RF) or Random Decision Forests [25] is a non-linear ensemble estimator that fits a multitude of classifying decision trees on various sub-samples of the dataset and outputs the mean prediction (regression) of the individual trees to improve the predictive accuracy and control over-fitting. However, since RF evaluates data points without bringing forward information from the past to the present, we need to define lagging variables to bring patterns from the past to be evaluated at the

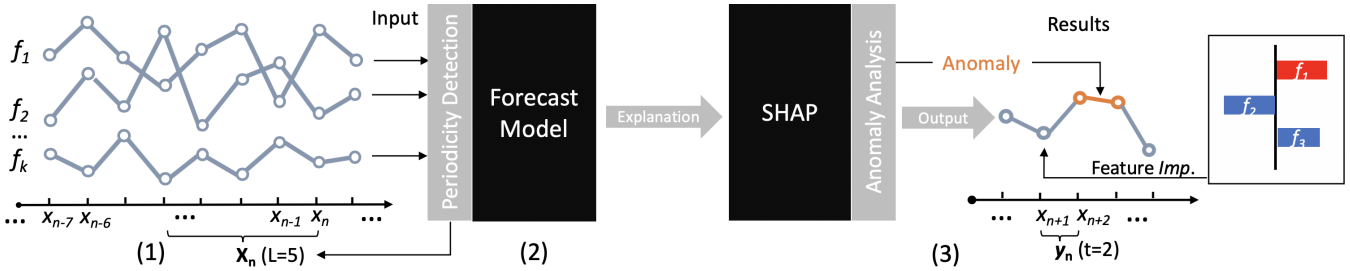


Figure 2: Evaluation pipeline: (1) input settings, (2) multivariate time-series forecast, and (3) results interpretation.

present. In detail, each input X_n is transformed into a $|L \times K|$-vector $X_{n^L} = [x_{n-L}^{f_0}, x_{n-L}^{f_1}, ..., x_{n-L}^{f_K}, ..., x_n^{f_0}, ..., x_n^{f_K}]$, where K is the feature number and L is the lag/length of input.

Long Short-Term Memory Recurrent Neural Networks (LSTM) is one typical variant of RNNs, which predicts time series by integrating three gates and one memory cell to solve the vanishing gradient problem in vanilla RNN. In the data preparation process, we frame the time series as a supervised learning dataset with each input sequence reshaped as a L*K matrix to fit the LSTM model and predicted values as a multi-step $\boldsymbol{y_n}$. We define the LSTM with 50 neurons in the first hidden layer and t (step) neuron in the output layer.

Multilayer Perceptron Models (MLP) is a class of feedforward artificial neural network (ANN). It consists of three or more fully connected layers of nonlinearly activation nodes and is usually used for fitness approximation. However, the multivariate time series is unable to be directly used as input for forecasts. Thus, the entire time series must be segmented into successive lag observations that are flattened into feature vectors as same as the aforementioned $|L \times K|$-vector. Similarly, we have a single hidden layer of 50 nodes and a t-step output layer.

Convolutional Neural Networks (CNN) is a regularized version of MLP. To reduce the complexity and the overfitting problem of MLP, it uses the hierarchical pattern in data and generates more complex patterns by assembling smaller and simpler patterns. In forecasting application, it also needs a data preparation process to transform the sequence of observations into multiple examples ($|L \times K|$-vector) from which the model can learn. The 1D-convolutional (128 neurons) and pooling layers are followed by a dense fully, connected layer (50 neurons) that interprets the features extracted by the convolutional part of the model. A flatten layer is used between the convolutional layers and the dense layer to reduce the feature maps to a one-dimensional vector.

4.1.2 Periodicity Detection. The main motivation for applying the periodicity detection is to find the best lag/length of input time series as periodicity is an important characteristic of time series. Thus, we employ a periodicity detection algorithm to estimate the latent period in the input time series, providing a "hint" to users when choosing the length L of input time series. The algorithm can make an accurate periodicity identification based on a new periodic distance measurement method [60], which is composed of two steps: extraction of candidate period and verification of candidate periods. Specifically, in the first step, we calculate the power spectrum or periodogram to discover the latent periods k by picking the top largest values of the periodogram:

$$P(f_{k/N}) = ||X(f_{k/N})||^2,\ k = 0, 1...\lceil \frac{N-1}{2} \rceil, \tag{1}$$

where $X(f_{k/N})$ is the Discrete Fourier Transform of the input time series at frequency $f_{k/N}$. Secondly, a candidate period from the periodogram is verified as a valid period if it lies on a hill (or the local maximum) of the AutoCorrelation Function:

$$ACF(\tau) = \frac{1}{N} \sum_{n=0}^{N-1} d(\tau) \cdot d(n+\tau),\ \tau = 0, 1...\lceil \frac{N-1}{2} \rceil. \tag{2}$$

This is done by calculating the first derivative using the Savitzky-Golay filter and finding where it crosses zero. Finally, we compute the first derivative value as the score for each candidate period. The higher the value is, the more possible it is the period of input time series.

4.2 Model Interpretation

After making time-series forecasts based on different models and with different lagged values/lengths of input time series, we analyze and interpret the forecasting results to help users understand the model performance and their differences. Thus, as shown in Fig. 2(3), a model interpretation method based on feature importance and an anomaly detection algorithm have been employed to explain the results.

Feature Importance. We apply the SHAP [41] values to analyze the relationship between input feature importance and the output forecasting results. SHAP is a classical sensitivity analysis method for model-agnostic machine learning interpretation. It uses Shapley values from game theory to allocate each feature an optimal importance value for a particular prediction, and the explicit definition is:

$$\varphi_i(x) = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} [f_{S \cup \{i\}}(x) - f_S(x)] \tag{3}$$

where S is a feature subset of all features F, $f_{S \cup \{i\}}(x)$ and $f_S(x)$ represents the model trained with feature i present and withheld, respectively. We use SHAP rather than other measurements (LIME, DeepLIFT, etc) as SHAP values prove more consistent with human intuition [41]. The values can add up to the actual prediction of the true model, which is consistent with our motivation in visual design to interpret feature importance in the context of the forecast results. Moreover, we speed up the computation of Sharply value by using approximations (e.g., the shap.kmeans function).

Anomalous Spikes Detection. In addition to calculate the feature importance at each forecast time step, we also want to get the deep insights into the forecasting models, such as the model performance regarding different inputs (stable or spike time series). Thus, an unsupervised anomaly scoring (AS) algorithm is developed to identify the anomalous spikes in the output forecasts, which is defined as:

$$AS = min\left(\frac{X_n - \mu_{n-1} - 3\sigma_{n-1}}{3\sigma_{n-1}} , 1 \right), \tag{4}$$

where μ_{n-1} and σ_{n-1} denote the mean and variance with respect to the residue set $\boldsymbol{X_n} = \{x_{n-L}, x_{n-(L-1)}, ..., x_n\}$.

5 VISUALIZATION

In this section, we present the design tasks derived from the discussions with our expert users, a brief summary of the user interface, and a detailed description of the visualization designs for each component.

5.1 Design Tasks

A list of design tasks was settled to guide the visualization designs based on the requirements outlined in **R1–R4**. We have discussed with the experts about their expectations when evaluating forecast models and the difficulties in meeting these expectations with visualization. For instance, it is difficult to extend existing techniques to identify salient features among different models and to explain forecasts which could be dependent on temporal factors of time series. Such derived factors are often domain-specific, requiring trials and errors to identify and augment. The problem is further aggravated when dealing with multivariate time series that involves the explanation of multiple features. The experts usually compare models based on the statistical plots with the average prediction accuracy and its variant metrics like F1 Score and Recall Rate, which lacks in-depth analysis of the model's performance. In general, they desired a tool that can facilitate the exploration, interpretation and comparison of multivariate forecast models, helping them in model selection in real applications. Guided by these considerations, we decided on a list of visualization tasks as follows.

- **T1 Show the overview of feature distribution and model performances.** The visual design should show the general distribution of the time series with multi-dimensional features to facilitate the initial selection of the training and testing dataset. In addition, the system should provide a visual summary of models' performance based on the selected data and their attributes.
- **T2 Interpret models' temporal patterns in different contexts.** The lack of understanding in the forecast models makes them untrustworthy and further limits their extension to other domain applications. Thus the visualization should be able to present the model's temporal patterns in a full context of the accuracy measurements, the feature importance, the anomalous results, and the relationships between models. Such a schematic representation can help users in evaluating models and finding optimal parameter settings for these models.
- **T3 Enhance model comparisons in model, instance and historical level.** Another key to understanding the characteristic of forecast model is the ability to show the differences between different models in a temporal context. Hence, the system should support model comparisons through intuitive representations and efficient interactions in different perspectives (e.g., the model level vs. the instance level). Also, for a specific model prediction, the feature importance at different historical time steps in the input data is also provided.
- **T4 Display the similarity of input instances.** In addition to displaying the temporal patterns of models, it is also important to show the similarity of different input time series and prediction results. To this end, the system should show the clustering of the input instances based on their similarities in features, revealing some specific prediction results with unique feature attributions.
- **T5 Allow flexible parameter settings of forecast models.** Users should be allowed to conveniently set the input of multivariate time-series forecast models and analysis algorithms with rich interactions and visual cues that indicate the influence of user behaviors.
- **T6 Provide easy access to raw multivariate data.** Despite the significance of the model interpretation and evaluation, the raw multivariate time series that contain different feature values are also essential for users to reason their evaluation and judgement, which should be easy to be accessed during the exploration process.

5.2 User Interface

Guided by the above design tasks and the experts' feedback, we designed our user interfaces (UI). As shown in Fig. 3, the UI of mTSeer system consists of six views to assist users from model construction to model evaluation and reasoning: *the data selection view* (Fig. 3(1)), providing an overview of the raw time series in a circular timeline **(T1)**; *the parameter view* (Fig. 3(2)), allowing the adjustment of model parameters and anomaly detection algorithm **(T5)**; *the model overview* (Fig. 3(3)), displaying the general performance of candidate models **(T1, 3)**; *the inspection view* (Fig. 3(4)), interpreting and comparing the forecast models via associating the temporal forecast results with the feature importance at each timestamp, which contains two modes (the other mode is *the bar chart mode* in Fig. 3(7)) **(T2, 3, 6)**; *the instance view* (Fig. 3(5)), allowing users to select the instances of input time series for further analysis in a multi-dimensional scaling view **(T4)**; *the historical view* (Fig. 3(6)), presenting the feature importance at historical time steps of input sequence based on different metrics **(T3)**. All the views are interactively connected to illustrate the model performance in different contexts. Different color schemes are designed to depict the different information (Fig. 3(g)). In particular, categorical colors are used to represent the features of the input multivariate time series. A linear color scheme, ranging from white to blue, is used for indicating the extent of the instance anomaly degree in Fig. 3(5) or the variance of feature importance in Fig. 3(3) from low to high,

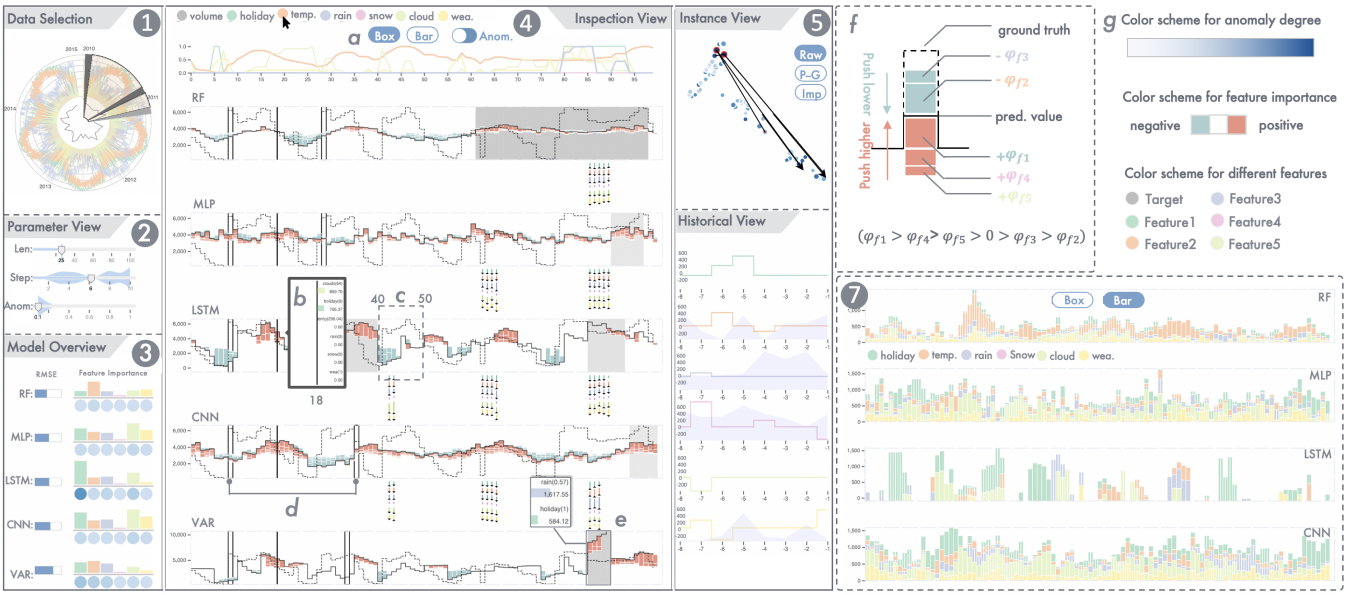


Figure 3: The mTSeer system contains six modules: (1) *Data Selection View*, (2) *Parameter View*, (3) *Model Overview*, (4) *Inspection View* in *the box mode*, (5) *Instance View*, and (6) *Historical View*. *The Inspection view* contains another mode: (7) *the bar chart mode*. Users can switch to different visualization modes and show the anomalous forecast results (e) by buttons in (a), retrieve the detailed feature importance value via tooltip (b), highlight some instances in (d). (f) is the explanation of the box glyph in (4). (g) shows the color schemes used in different views.

while a dichotomous color scheme encodes whether a feature positively (red) or negatively (blue) affects the prediction results. More design details of each view are introduced in the following sections.

In general, the basic workflow of our system can be summarized into: (1) **model construction**, (2) **model evaluation**, and (3) **instance inspection**.

5.3 Model Construction

mTSeer contains two views for the construction of forecast models before evaluating them: (1) *the data selection view* displays the raw multivariate time series and the anomaly degree at each timestamp; (2) *the parameter view* facilitates the parameter adjustment for the forecast models. They provide users a more personalized setting for in-depth model evaluation based on their requirements and domain knowledge.

5.3.1 The Data Selection View. *The data selection view* in Fig. 3(1) helps users observe the general distribution of the raw multivariate time-series data and then query the data based on the temporal patterns of the feature values and anomaly degree. We employ a circular timeline from 0° to 360° in this view to save space. In the outer ring, each categorical color scheme is used to represent a specific feature attribute that has been normalized in the y-aixs (radius). By contrast, there is a black line in the inner circle which shows the anomaly degree of data points at each timestamp, providing additional information for users when selecting data. Users can select two segments with irregular feature values or special temporal patterns as the training dataset (the preceding one) and testing dataset in chronological order. **(T1)**

5.3.2 The Parameter View. *The parameter view* displays a list of parameters for setting the forecasting models and the anomaly detection algorithm to assist users in evaluating forecasting models with diverse conditions (Fig. 3(2)). The shaded area along each slider reveals the sensitivity intervals when adjusting each parameter. Specifically, the first slider is used to change the length/lag of input time series of forecasting models. The height of shadow along the slider encodes the possibility of a length to be the period, which is an important characteristic of time series that might affect the forecasting results. The second slider allows users to set the forecast steps into the future, the shadow of which encodes the differences in forecasting results when changing the step value to its consecutive one. Finally, there is a slider for adjusting the discrimination threshold of anomaly detection logarithm. The shadow encodes the number of anomalous forecast results with each threshold. **(T5)**

5.4 Model Evaluation

After setting the model by selecting the input time series and the parameters, the system provides three views for model interpretation and evaluation, namely, *the model overview*, *the inspection view* and *the historical view*.

5.4.1 The Model Overview. *The model overview* aims at providing an overall comparison of different forecast models via displaying the general performance of each candidate model. As shown in Fig. 3(3), we present two types of measurements for each model, the Root-mean-square Error (RMSE) and the summarized feature importance, to let users make a quick evaluation based on the accuracy measurement and interpret the overall model performance

with different features' contributions. On the one hand, we use the width of the fill rectangle (blue) to encode the RMSE value from 0 (empty) to 1 (full) that is normalized across all the candidate models. The RMSE value for each model is computed based on the same testing time series, which is defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\{y_{T+t} - g(x_{T+t}, \hat{\theta})\}},$$

where y_{T+t} is the ground truth, $g(\cdot)$ is the forecasting function, X_{T+t} is the input time series, and $\hat{\theta}$ is the estimated parameter based on the training dataset. On the other hand, we design a specific glyph to encode the summarized feature contribution to a candidate model. The upper part is a bar chart that shows different features' importance value, with each categorical color represents one feature attribute. The importance values are normalized across all the models, and the higher a bar goes, the larger the feature importance is. Since the importance value for each feature is summarized from the whole testing time series that may contain hundreds of data points, we thus use a blue circle in the lower half to depict the variance of importance values for its corresponding feature (bar) above. The fill color, ranging from white to blue, is used for indicating the variance from small to large. Finally, users are allowed to change the listing order of the candidate models via dragging and moving the model name in this view, which facilitates the pair-comparison of any two models in the following steps. **(T1, 3)**

5.4.2 The Inspection View. *The inspection view* displays a detailed model-agnostic explanation of multivariate time-series forecasting models by associating the feature importance to the model output with the raw time-series information, the forecasting results and the ground truth. As shown in Fig. 3(4), this view can be divided into three parts: (1) a line chart (Fig. 3(a)) on the top that plots the raw testing time series; (2) model explanation plots below that depict the detailed interpretation of each candidate model's performance in the temporal context, which can be switched from the current *box mode* to the *bar chart mode* shown in Fig. 3(7); (3) diff-line glyphs between two models (Fig. 3(e)) that show the explicit difference between two models in terms of feature importance. The three parts share the same timeline. Firstly, the line chart in the top is used to provide the raw time series context for reference (**T6**), where different feature values are scaled and represented by the same categorical color scheme shown in the legend. Secondly, different model explanation plots are placed in the order consistent with the order in *the model overview*. So users can change the order of models to meet their requirements for pair comparison. In each part, the solid line represents the forecasting outputs of the candidate model in the testing dataset, and the dotted line represents the ground truth. Therefore, users can observe the gap between the real value and the predicted value at different timestamps. The time periods with a grey background mean the forecasting results within these periods are identified as anomalies (usually long-term spikes). In addition, in each timestamp, we design a "box glyph" (Fig. 3(f)) to facilitate the exploration of feature contribution to model output in multifaceted contexts. The glyph design is inspired by combining the basic line chart representing the evolution of predictions [54, 58] and the bar charts showing the quantity of feature importance [41, 50] **(T2)**

Glyph Design. This glyph is placed in the y-axis position of the forecast value (base value) at each timestamp. Each box in the glyph represents one feature. The boxes stacked above the base value show features with the negative effect that contribute to pushing the forecast value lower, and those stacked under the base value are positive features pushing the forecast value higher. A dichotomous color scheme is used to show whether the feature contribution is positive (red) or negative (blue). The boxes in the upper side are ranked in the increasing order of the feature importance values (negative) from top to middle, while the boxes in the lower side are in the descending order of positive importance values. The advantages to using this design mainly contain two aspects: (1) the feature importance can be intuitively and seamlessly represented based on the forecasting values across time. It also means that if we connect the dividing points between the positive (red) and negative (blue) boxes at each timestamp, the connecting line (solid line) is the forecasting result itself; (2) the more important features are aligned closer to the base values. Finally, the users can hover on the glyph to retrieve the explicit feature name, feature values (in the brackets) and the feature importance values in the tooltip (Fig. 3(b)), and the order of features in the tooltip is consistent with the box order in the glyph (**T6**).

Alternative Design. Several design alternatives for the "box glyph" were considered, as shown in Fig. 4. The first and most intuitive one was to use the multi-line chart that contains all the feature importance values in a single graph (Fig. 4(a)). Although it can show the continuous evolution of one feature's importance, this visualization has two problems: one is the severe visual clutter when feature number or time span is large, and the other is the lack in revealing the accumulative effect of features at each timestamp. Another alternative was the stacked bar charts with the same placing order as the "box glyph" mentioned above (Fig. 4(b)). But this method fails to display the continuous change of a feature because the order of different features is not fixed at each timestamp. To solve this issue, the third one (Fig. 4(c)) was developed by extending the second one, where we fixed the order of features across time. The height of bar encodes the absolute feature importance value, while the negative feature is highlighted with a strike in the middle of the corresponding bar. All the three designs are unable to interpret the feature importance in the context of the forecast results. In this sense, our design in Fig. 4(d) is superior as it integrates the forecast results and also provides the detailed feature-related information with a tooltip. Finally, after discussing with the experts, they thought the third design as useful in some cases when users just focus on analyzing the features' importance and their evolution trends. Thus, we integrated the third design in *the inspection view* as **the bar chart mode** (Fig. 3(7)). Users can click the button in the top to switch between these two modes.

Thirdly, diff-lines between two models display the explicit difference of feature importance (Fig. 6(b2)). Specifically, each line segment at a timestamp represents one feature. They are encoded

with the same categorical color scheme for features and have fixed order at each timestamp. The segment with a downward arrow shows the descending in importance of its represented feature from the top model to the bottom model, whilst the segment with an upwards arrow shows the increase in feature importance. The absence of an arrow means the unchanged feature importance. The diff-lines are not shown by default, and users can click at a time point to retrieve them. Finally, users can filter out some features in *the inspection view* by clicking the legends at the top (Fig. 3(a)) to make a more precise exploration.

5.4.3 The Historical View. This view (Fig. 3(6)) is designed to show the historical feature importance of each attribute for a specific model forecast. Since the feature importance for each model at different time steps in *the inspection view* represents an average impact of features in the input data, we are also interested in which time step in the input time series has a greater impact to the forecast. For example, the old data points in the input time series might have a larger contribution to the forecast than the most recent data. When clicking one column of a model in *the inspection view*, each feature's historical importance and its trend for this forecast will be displayed in an individual line chart, complemented with the background area chart showing the row feature value. The x-axis represents the timeline of the input sequence before a selected forecast, and the y-axis represents the feature importance for the line chart or raw feature value for the area chart. **(T3)**

5.5 Instance Inspection

The instance view (Fig. 3(5)) shows the spatial distribution of input time-series instances at different timestamps using t-Distributed Stochastic Neighbor Embedding (t-SNE). The multivariate points are scaled into a 2D plane based on their raw feature values (Raw) and clustered into subsets exhibiting a certain similarity. Each node in this view represents one data point whose fill color encodes its anomaly degree calculated by the Local Outlier Factor (LOF). The darker the color, the more abnormal it is. In addition, since one data point has different forecast results from different models, we use the size of the node to encode the variance of results. The larger it goes, the more variant the forecast is. Users can click on a specific node to highlight the corresponding data point simultaneously in *the inspection view* (Fig. 3(d)) for further analysis. Also, a connecting line will appear to show the neighboring instances of a selected node, starting from the previous point (annotated with a red dot) to the next point (annotated with an arrow). Finally, users can select other relationships to be projected and analyzed in this view via the selection buttons, such as the predicted value vs. ground (P-G) value for each model, or the similarity in feature importance (Imp). **(T4)**

5.6 Interactions

The following interactions are designed to help with the exploration and evaluation of the multivariate time-series forecast models. **Query and Filtering.** Users can query different subsets of the input time series by brushing the segments in *the data selection view*. They can also filter out some features for a more clear inspection by clicking the feature legends in the top of *the inspection view*. **Setting Parameters.** Users can set models' input based on

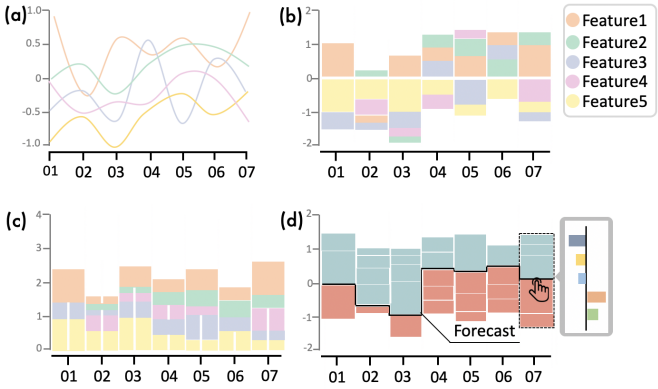


Figure 4: Alternative designs for feature importance: (a) multi-line chart, (b) stacked bar chart, (c) an extended stacked bar chart. Our design (d) improves the representation by integrating the predicted values.

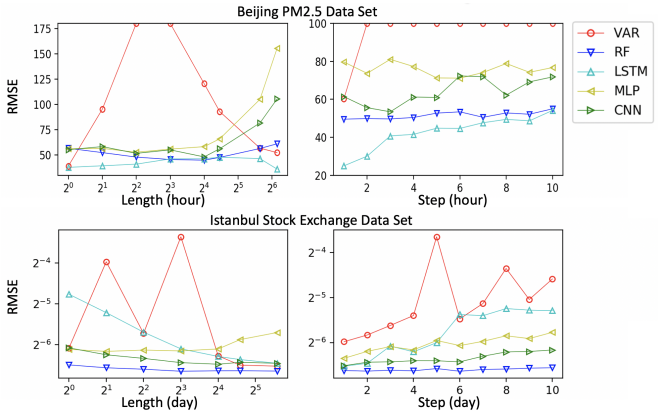


Figure 5: The accuracy of forecast models under different input time-series lengths and forecast steps. The plots on the top and bottom show the RMSE of different models with the data used in case 1 and 3, respectively.

the characteristics of time series, as well the anomaly threshold by tuning the sliders in *the parameter view*. **Switching Context.** Users can switch between different visualization modes for the model explanation in *the inspection view*. In *the model overview*, they can switch the position of any two models by dragging and moving the container of the model to change the listing order of models in *the model overview* and *the inspection view*. **Zooming and Scaling.** Three views support zooming for a large set of data items, namely, *the data selection view*, *the inspection view* and *the instance view*. **Tooltips and Highlighting.** Tooltips are provided in *the inspection overview* to provide the detailed feature information with a bar chart design. In addition, the anomalous periods in the forecasting results can be highlighted with a grey background by clicking the button at the top of *the inspection overview*. In *the instance view*, when clicking a node, its previous and next point will be connected with a line. The corresponding instance will also be highlighted in *the inspection overview*, and vice versa. **Model Comparison.** Users

can click the "box glyph" in *the inspection view* to show the detailed difference between two models with diff-lines.

6 EVALUATION

We evaluated the effectiveness of mTSeer in multiple ways from the visualization community [28, 38]. First, we describe three case studies with two domain expert who had no exposure to our system before the case studies on three real-world multivariate time series from UCI Machine Learning Repository: (1) Beijing PM2.5 Data Set, (2) Metro Interstate Traffic Volume Data Set, and (3) Istanbul Stock Exchange. The datasets (1) and (2) have similar attributes (meteorological data) as input, but the forecasting targets are different. For each case study, we conduct a corresponding quantitative evaluation using RMSE changes to display the effect of input time series length and forecast steps in predict accuracy. Finally, we conduct qualitative expert interviews with the two domain experts about their user-experience and feedback to mTSeer.

Study Set-up and Interview Process. We invited two experts who have rich experience in visual analytics, machine learning, and time series to conduct the case study in a semi-structured format. The one for case study 1 & 2 is a research scientist from a company's visualization group, who is experienced in urban informatics (**E1**). The expert for case study 3 is an algorithm engineer from a start-up, and his main work is to develop models for analyzing stock-related data (**E2**). Both of them had no prior knowledge about the data used in the case studies and were not collaborators on our project. Each interview lasted approximately 1.5 hours, with 30 minutes for the introduction of the system and 60 minutes for data exploration and model evaluation by the experts themselves using our system. Notes and expert feedback were recorded in the process.

6.1 Case Study I: Beijing PM2.5 Data Set

In the first case study, we used a five-year (2010/01/01 – 2014/12/31), hourly sampled time series for pollution (PM2.5) forecasting. The dataset includes seven meteorological attributes from Beijing Capital International Airport, namely, dew point (dew), temperature (temp), pressure (press), combined wind direction (win_dir), cumulated wind speed (win_spd), cumulated hours of snow (snow), and cumulated hours of rain (rain), and the target is PM2.5 value recorded in US Embassy in Beijing. After loading the data into mTSeer, **E1** started the **model construction** by first selecting the training and testing data in *the data selection view* in Fig. 3(1). He noticed the overall one-year seasonality revealed by most features (e.g., temp), and an anomalous period around the start of 2011 indicated by the anomaly line in the inside circle (**T1**). So he brushed this abnormal period (approx. from Jan. 2 – 5) as testing data and its previous one-year time series as training data. Then he set different parameters in *the parameter view* (**T5**). Based on his domain knowledge, he chose to make a one-step forecast as the result would be more accurate and useful, and he also preferred to set a short length of input time series because he believed the next hour's PM2.5 value was most relevant to its previous few hours; thus he tried the parameter length with 1 to 4 hours and finally decided to use 4 due to its higher overall accuracy for five models observed in *the model overview*.

Then he started the **model exploration and evaluation** with other views. Firstly, as shown in Fig. 6(a1), the RMSE rectangle in *the model overview* informed him that LSTM was the best model based on his parameter setting. Also, the bar chart showed that LSTM has the most average and least feature contribution, especially compared with CNN and MLP, which are the same type of model as LSTM. This result attracted **E1**, and he switched to *the inspection view* for detailed analysis of LSTM and others' performance (**T2-3**). The first interesting pattern in this view was the high and unstable PM2.5 forecast results for all models from Jan. 2, 6 pm to Jan. 4, 6 pm (Fig. 6(b4)), compared with the forecasts before and after this period. After checking the raw time series in the top, **E1** said this was caused by the frequently changed wind direction (win_dir, pink) and the high dew value (dew, dark green) within this period. Next, to figure out why LSTM is better than models like RF and MLP, he identified two anomalous periods (Mon 2 am – 8 am & Mon 1 pm – Tue 2 am) in Fig. 6(b1) where the forecasts (solid line) were extremely lower than the ground truth (dotted line) for RF, CNN, and MLP but not for LSTM, which leads to the LSTM's lower error. Then the expert dig deeper into these two periods to find out which features made LSTM better than others. By using the tooltip and the diff-lines, he found that within the two periods, LSTM had lower feature importance in temperature (orange), wind speed (purple) and direction (light green), and higher importance in dew (dark green) and pressure (blue) compared with RF and MLP (Fig. 6(b2)). These findings of feature contribution inspired the expert that the higher forecast accuracy of LSTM in high PM2.5 value should credit to the higher dew and pressure contribution. Furthermore, **E1** also tried *the bar chart mode* in *the inspection view* and filtered out the rain and temperature attributes because of their anomalous high importance in VAR (Fig. 6(b6)). Then the results in Fig. 6(b5) clearly and consistently showed that: for models (CNN, MLP, RF) that had many forecasts lower than ground truth, their dew point and pressure always had large negative importance, while for models that were more accurate like LSTM, there were no long-term standing features across time. Finally, he also double-checked the anomalous instances in *the instance view* (Fig. 6(c1)), and noticed they (Fig. 6(b3)) were also located within the anomalous periods as mentioned above (**T4**).

Quantitative results. We validated how the RMSE changes under different lengths of input time series and the number of forecast steps. Specifically, we summarized the RMSE based on a linearly-growing forecasting steps [1,2,3,4,5,6,7,8,9,10] hours. For the RMSE on input time-series length, we test the models with eight sampling rates growing near-exponentially: [1,2,4,8,16,25,50,71] hours since there is a latent periods (i.e., 71) identified by our periodicity detection algorithm. The results are shown on the upper Fig. 5 (the extremely high values of VAR are not shown in this figure.) As expected that the pollution is more related with its most recent weather data, the RMSE shows an overall increasing trend as the length increases, but some models (VAR, LSTM and RF) have an decreasing trend around 16 and the estimated period 71, which proves that the input time-series length (e.g., around period) can affect the forecast accuracy. Additionally, most models present more errors as expected when the forecast step increases. Moreover, the result shows that LSTM is the best algorithm all the time, which also verifies the evaluation results in the case study.

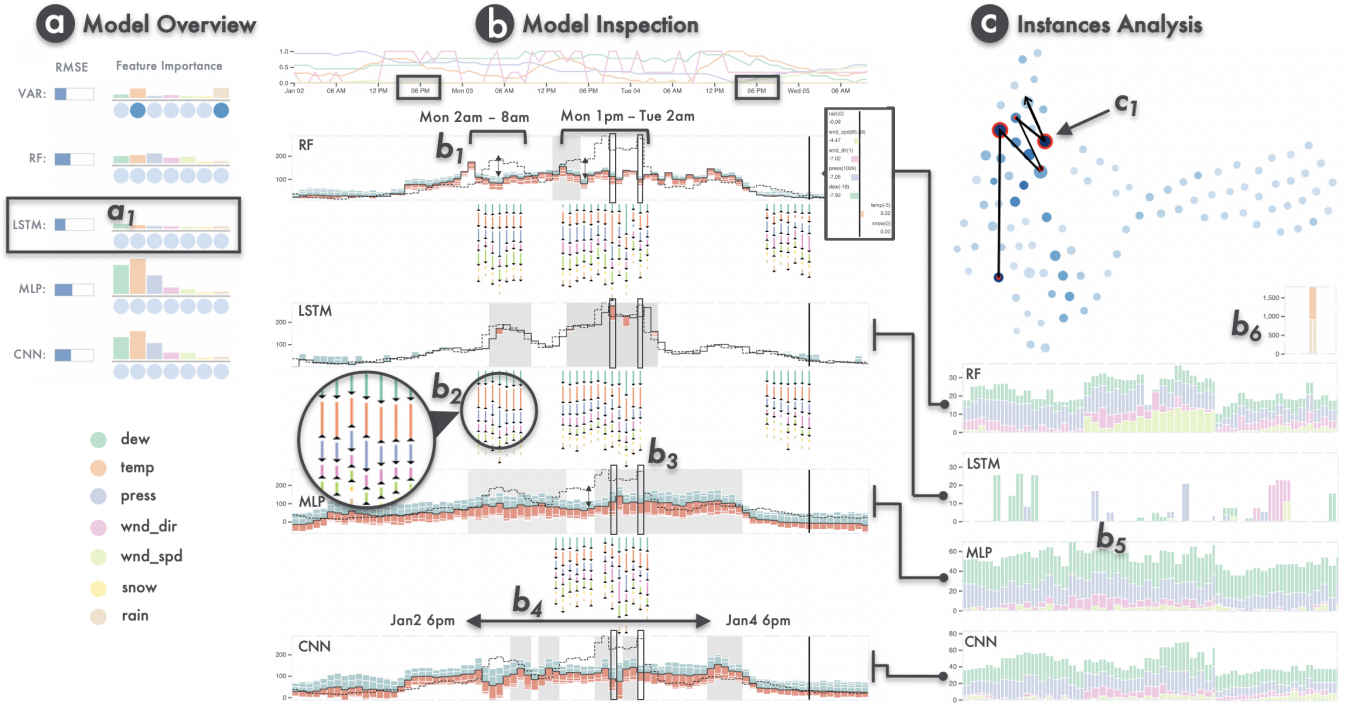
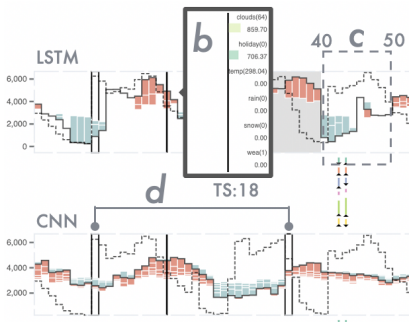


Figure 6: mTSeer facilities the exploration of multivariate time-series forecasting models through three levels of analysis: (a) model overview, (b) model inspection, and (c) instance analysis. The figure showcases some exploration results with Beijing PM2.5 Data Set. The input is meteorological data and the forecast target is PM2.5 value. (a1) shows that LSTM is the most accurate forecasting model based on RMSE value and has the least feature contribution. Specifically, LSTM makes fewer forecast errors than others within two time periods shown in (b1), and the further analysis indicates a higher importance of dew and pressure for LSTM within these two periods by observing the diff-lines in (b2). The bar charts in (b5) also verify this finding because all the models except LSTM have long-term negative feature importance of dew and pressure over time. In general, all the models have high and unstable predicted PM2.5 values within the period (b4). The anomalous points discovered in (c1) are highlighted in (b3) correspondingly.

6.2 Case Study II: Metro Interstate Traffic Volume Data Set

E1 continued cooperating with us in the second case study because the time series is still related to meteorological data. The forecast target is traffic volume, which matches his expertise in urban informatics. This dataset records hourly traffic volume (2012 – 2018) for MN DoT ATR station 301, and hourly weather features and holidays are included for impacts on traffic volume. In the **model construction** step, **E1** selected a 100-point period from 2013-09-02 as testing data and its previous one-year as the training data. Additionally, since *the parameter view* revealed an interesting 24-hour (i.e., one day) period of traffic volume (Fig. 3(2)), the expert decided to set the length of the input time series as 24 hours, and the step as 6 hours (quarter day) based on his experience. Then he started to make **model evaluation**. Firstly, in *the model overview*, he dragged the model container to re-rank them in an increasing order of RMSE value (RF, MLP, LSTM, CNN, VAR) in Fig. 3(3). By observing the bar charts of feature importance, he quickly noted that the best model RF was the only one taking the temperature (orange) as the dominant feature, while most others took the holiday (dark green) and cloud (light green) as the dominant features (**T3**). Thus, he switched to the other views to make a deep evaluation of these models, and here are some important findings summarized in **E1**'s study: (1) the raw time series on the top of *the inspection view* displayed a periodic pattern of feature temperature, which was consistent with the ground truth (dotted line) of volume change shown in model explanation plots below (Fig. 3(4)); thus most models showed a similar periodic pattern in forecasts (solid line) except MLP. Although MLP had the second low RMSE, its performance was not considered as good by the expert since most of its predicted results were average across time, which makes no sense for a forecasting model. By contrast, the forecasts of LSTM fitted to the ground truth at most time, but had some significant errors in some periods like timestamp 40 to 50 (Fig. 3(c)), which resulted in its RMSE only ranked third in five models. However, **E1** still regarded LSTM as good and a detailed discussion about this is described in Sec.6.4 (**T2**). (2) Meanwhile, the expert tried to find more explicit explanations of model performances from the perspective of feature importance. So he observed the "box glyph", and found some models like LSTM and CNN contained large bars in the "box glyph" across time (Fig. 3(b)).



The tooltip further showed that these models' forecasts were significantly affected by feature cloud and holiday (**T5**). Although these features led to the good performance (e.g., around timestamp 18) when there were sudden changes in cloud cover or holiday status in its past 24 hours, they also caused the bad performance in periods (e.g., timestamp 28) when there were no extreme changes in such features in the past. The first ranked algorithm RF, however, was more stable since its dominant feature across time is always the temperature. The bar chart mode shown in Fig. 3(7) verified this. (3) The VAR was worst in RMSE because it contained an anomalous period highlighted with grey background (Fig. 3(e)), where its forecasts were significantly pushed up by the feature rainfall (purple) and led to the huge error in RMSE. Although the raw time series showed the sudden presence of rainfall in its past few hours, **E1** believed the VAR excessively over-weighted the importance of rainfall. (4) Finally, **E1** also clicked several most abnormal (dark blue) points in *the instance view* (Fig. 3(5)), and the connected lines meant that their neighbouring points had little correlation with them. This was verified in *the inspection view* in Fig. 3(d) as these points were spikes where all the models made wrong forecasts. (**T2-4**)

Quantitative results. Similarly, we tested the relationship between RMSE and the lengths of input time series, as well as the forecasting steps. To save space, we displayed the results in supplementary materials as they are similar to the quantitative evaluation results in case study 1.

6.3 Case Study III: Istanbul Stock Exchange Data Set

In the third case study, the dataset includes returns of Istanbul Stock Exchange (ISE) with seven other international index; SP (S&P 500), DAX (Germany), FTSE (UK), NIKKEI (Japan), BOVESPA (Brazil), EU (MSCE), EM (MSCI), from Jun. 5, 2009 to Feb. 22, 2011. **E2** started by selecting the training and testing time series and then tried many parameter settings. Here we only report one special case with the length set as 4 days and the step as 7 days because LSTM was worst in this case. The expert thought this might be caused by the short input length for LSTM and he wanted to explore more about the reasons. After manually ranking the models based on the RMSE value, he began to evaluate models with different views. He first observed the raw time series on top of Fig. 7 and noticed that the seven indexes changed synchronously at most times (**T2, 6**). The first concern of **E2** was to figure out when the forecasts went wrong (**T2**), so he marked the timestamps where most models make errors and analyzed them by referring to the raw data. The results showed that the wrong forecasts often happened when their

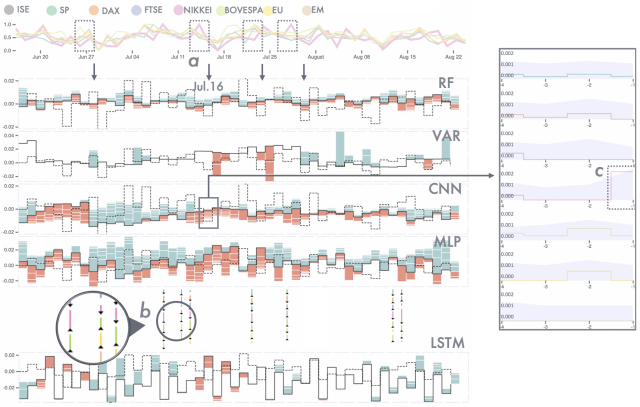


Figure 7: Model evaluation results with Istanbul Stock Exchange Data. (a) The raw time series shows inconsistent changes of input data at some timestamps, which leads to the significant forecast errors in their next few days. (b) The diff-lines explains why LSTM is worse than MLP as BOVESPA has higher importance for LSTM forecast. (c) *The historical view* shows the spike of NIKKEI at timestamp (a) leads to the abnormal larger feature importance than that at other timestamps for CNN.

previous points had inconsistent changes among the seven indexes. For example, as shown in Fig. 7(a), before Jul. 16, all the indexes had a similar trend except NIKKEI that had a spike at Jul.15, which made the wrong forecast on Jul. 16 for all models. *The historical view* (Fig. 7(c)) verified this finding in model CNN: the input data of NIKKEI had an extreme large feature importance at Jul.15, pushing the forecast value at Jul.16 higher than ground value. In addition, he wanted to understand why LSTM was worse than others like MLP in this case, so he observed the "box glyph" and retrieved the diff-lines where LSTM made mistakes whilst MLP not. By inspecting the diff-lines shown in Fig. 7(b), he found there was only one index that always had higher importance in LSTM than MLP – BOVESPA (light green). Thus **E2** guessed that the other models were better because they took little account of BOVESPA in their forecasting process.

Quantitative results. In this experiment, we tested the RMSE of different models with input length growing exponentially: [1,2,4,8,16,24,48], and step linearly growing from 1 to 10 hours. No period is detected in this data. As shown in Fig. 5, the general RMSE are consistent with the exploration results in this case study, where RF is best and LSTM is not good. Moreover, different from the results in case study 1, the RMSE decreases as the input sequence length increases, which makes sense since the exchange rate is more related with the long-term impact of past data.

6.4 User Feedback

We collected the insightful feedback provided by the two experts (**E1**, **E2**) and summarized them into three themes.

Interpretable Model Evaluation. Both experts applauded the interpretability of our system in evaluating time-series forecasting models. **E1** used to take accuracy measures as the major criteria for

model evaluation. However, after the case study, he described an important insight that "Don't use the accuracy [RMSE] as the only criterion." *The inspection view* associating the RMSE with feature importance made him understand that "although some models have large RMSE, they are caused by a few anomalous points []...] these models still have very good performance in general and should be selected in our applications." By contrast, he said that some models might have a good overall accuracy because their prediction results are very average and "within a safe range". But such models have little practical value, especially when we need to make forecasts at critical times. Moreover, **E2** stated, "Your system combines a lot of information to help users evaluate models [...]such as the raw data, feature importance and anomalies.", and this is indeed one goal of designing the system: interpret the exploration results from different perspectives.

Visualization and Interaction. Most visualization and interaction designs were thought to meet the design tasks. Both experts agreed that *the model overview* and *the inspection view* are very useful and informative. "These [two] views provide a comprehensive explanation and comparison of the forecasting models," **E1** said, "I know RMSE and feature importance, but I never consider to combine them together in visualization design." **E2** regarded the *the historical view* as helpful because "it provides more detailed understanding of feature contribution" and he even suggested to dig deeper in this direction. In addition, they both commented the interaction designs, such as the tooltip and the coordinated highlighting as helpful, bringing convenience to retrieve the reference information when needed. Finally, both experts mentioned that it took a while to fully grasp the "box glyph" design in the *the inspection view* as though it is useful.

System and Improvements. The experts can adapt well to the workflow of our system from model construction to model evaluation and other analysis. **E1** liked *the parameter view* as "the shadow gives me the references when selecting parameters." However, both of them suggested that more automatic analysis is an interesting direction. For example, "Although your system has automated anomaly analysis for time series, " **E1** suggested, "More intelligent [time-series] analytic methods can be integrated into the system to provide more specific temporal patterns.", which is indeed helpful to improve the understanding of the forecasting results and the raw time series. **E2** also suggested that our system can recommend some input data and optimal parameter settings at the beginning to alleviate the efforts in adjusting parameters. Although it is difficult to find unified methods that could deal with different types of time series, we believe that more automated analysis in our system will help users perform their tasks more effectively.

7 DISCUSSION

In this section, we discuss the high-level synthetics, the implications, and the limitations of our system designs.

Contributions with respect to previous work. According to the survey of related work, the line chart-based visualization designs [16] are still the basic and most intuitive method to tackle time-series (TS) model evaluation, especially for univariate time series [8, 58]. However, it is more challenging to deal with the multivariate TS model as the univariate model does not need to compare feature importance, which is an essential model-agnostic method to interpret forecasting results based on input attributions. Thereby, we investigate the visualization designs for feature importance [50], preserve their basic format (bar chart), and change their layout to make the feature importance be interpreted along with the prediction results. We also integrate some additional tooltips and views to provide a more comprehensive understanding. Specifically, the design of the "push up/push down" feature importance glyph is a combination of the line chart representing the predicting value and the bar chart representing the feature importance. The reason to introduce the "push up/push down" concept comes from the theoretical background of SHAP model [41] where the expected prediction is attributed to the change of each feature when conditioning on that feature. Since there are positive and negative changes (importances) for different features, we thus adopt the "up/down" concept to show the positive/negative changes that push the expected predicted value higher/lower. By associating the feature importance at each forecasting step, the influences of different features can be efficiently interpreted with more semantic meanings. The lessons that incorporating the theoretic definitions of model explanation methods and the characteristics of input data into the basic visualization forms can be used in designing other visual representations of explainable machine learning models.

Extension to other forecasting model interpretations. Although this work focuses on the model evaluation on multivariate time-series forecast, the interactive framework we proposed can be widely applicable. The high-level synthesis of building the system can be summarized into the following points: (1) the system pipeline as shown in Fig. 1 provides a steerable evaluation method for integrating model construction and analysis process with visualization designs, which can be extended to other model evaluations related with time series or machine learning. (2) The workflow of the visualization interface should be generally consistent with the system pipeline, such as the major model construction and model evaluation modules designed in our interface. (3) To solve time-series problems, we have to pay more attention to the temporal patterns and issues that are inherent characteristics in the data, including but not limited to the length and period discussed in our work. (4) There are some challenges remained to develop the next system of this type. Although our system supports a number of parameters, our evaluation with experts identified a larger parameter space that are valuable topics for future work. For example, the input can be selected by considering more features like the trend of time series, the correlations between different features. A larger number of candidate models and their hyperparameters is also an interesting direction. Finally, the instance-level evaluation can be expanded in both analytic methods and visualization designs.

Limitations. Based on the feedback of our case study and expert interviews, we have identified certain limitations of mTSeer. First, the paper only considers about 10 features in the forecasting, while many real-world applications may involve tens to thousand features, which may lead to a serious visual clutter with the current visualization design. Although we can mitigate the scalability issue by only showing the most important or interesting features for each forecast and allow users to zoom in to get more detailed explanations, there is information about other features omitted from the design. Therefore, we believe that improving the design, especially

the "box" glyph in *the inspection view* will make the system more practical. Second, although mTSeer has a good performance in comparing different models under the same condition, it is limited in comparing the same model with different parameter settings. Preserving the exploration results at different iterations and then retrieve the historical results later for a specific model is a potential solution.

8 CONCLUSION AND FUTURE WORK

We have presented an interactive model exploration system, mTSeer, that enables expert users to evaluate the performance of different models on multivariate time-series forecasts based on a model-agnostic explanation method. The system supports the evaluation process at different levels by integrating the feature importance, some time-series analytic algorithms, and a variety of coordinated contextual views and interaction designs for model interpretation. We demonstrated the effectiveness of mTSeer through three case studies using real-world time series, a corresponding quantitative evaluation on model performance, and follow-up interviews with domain experts. These results are promising, though there are several interesting directions for future work. We plan to add more automatic methods for multivariate time-series analysis in our current system to improve further the reasoning of forecasting results, as well as the efficiency of user exploration. Moreover, we intend to make mTSeer an active-learning system that can dynamically update the evaluation results of models based on human feedback in the exploration process. Finally, we will conduct a formal user study to demonstrate the usability of different views in our system.

ACKNOWLEDGMENTS

We thank all the study participants and reviewers for their comments. This work was partially supported by Capital One and NSF awards CNS-1229185, CCF-1533564, CNS-1544753, CNS-1730396, and CNS-1828576. .

REFERENCES

[1] Nesreen K Ahmed, Amir F Atiya, Neamat El Gayar, and Hisham El-Shishiny. 2010. An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews* 29, 5-6 (2010), 594–621.

[2] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. 2008. Visual methods for analyzing time-oriented data. *IEEE Transactions on Visualization and Computer Graphics* 14, 1 (2008), 47–60.

[3] Naomi S Altman. 1992. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician* 46, 3 (1992), 175–185.

[4] Sylvain Arlot, Alain Celisse, et al. 2010. A survey of cross-validation procedures for model selection. *Statistics Surveys* 4 (2010), 40–79.

[5] J Scott Armstrong and Fred Collopy. 1992. Error measures for generalizing about forecasting methods: Empirical comparisons. *International Journal of Forecasting* 8, 1 (1992), 69–80.

[6] Devon K Barrow, Sven F Crone, and Nikolaos Kourentzes. 2010. An evaluation of neural network ensembles and model selection for time series prediction. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*. IEEE, IEEE, Barcelona, Spain, 1–8.

[7] Christoph Bergmeir and José M Benítez. 2012. On the use of cross-validation for time series predictor evaluation. *Information Sciences* 191 (2012), 192–213.

[8] Markus Bögl, Wolfgang Aigner, Peter Filzmoser, Tim Lammarsch, Silvia Miksch, and Alexander Rind. 2013. Visual analytics for model selection in time series analysis. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2237–2246.

[9] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. 2015. *Time series analysis: Forecasting and control*. John Wiley & Sons, Hoboken, NJ, USA.

[10] Leo Breiman and Jerome H Friedman. 1997. Predicting multivariate responses in multiple linear regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 59, 1 (1997), 3–54.

[11] Lee Byron and Martin Wattenberg. 2008. Stacked graphs–geometry & aesthetics. *IEEE Transactions on Visualization and Computer Graphics* 14, 6 (2008), 1245–1252.

[12] Francesco Camastra and Maurizio Filippone. 2009. A comparative evaluation of nonlinear dynamics methods for time series prediction. *Neural Computing and Applications* 18, 8 (2009), 1021.

[13] Patrick S Carmack, William R Schucany, Jeffrey S Spence, Richard F Gunst, Qihua Lin, and Robert W Haley. 2009. Far casting cross-validation. *Journal of Computational and Graphical Statistics* 18, 4 (2009), 879–893.

[14] Michael P Clements, Philip Hans Franses, and Norman R Swanson. 2004. Forecasting economic and financial time-series with non-linear models. *International Journal of Forecasting* 20, 2 (2004), 169–183.

[15] Gustavo Deco, Ralph Neuneier, and Bernd Schümann. 1997. Non-parametric data selection for neural learning in non-stationary time series. *Neural Networks* 10, 3 (1997), 401–407.

[16] Naser Ezzati-Jivan and Michel R Dagenais. 2017. Multi-scale navigation of large trace data: A survey. *Concurrency and Computation: Practice and Experience* 29, 10 (2017), e4068.

[17] Tingting Fang and Risto Lahdelma. 2016. Evaluation of a multiple linear regression model and SARIMA model in forecasting heat demand for district heating system. *Applied Energy* 179 (2016), 544–552.

[18] Johannes Fuchs, Fabian Fischer, Florian Mansmann, Enrico Bertini, and Petra Isenberg. 2013. Evaluation of alternative glyph designs for time series data in a small multiple setting. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Paris, France, 3237–3246.

[19] Steve R Gunn et al. 1998. Support vector machines for classification and regression. *ISIS Technical Report* 14, 1 (1998), 5–16.

[20] David Gunning. 2017. Explainable artificial intelligence (XAI). *Defense Advanced Research Projects Agency (DARPA), nd Web* 2 (2017), 1–8.

[21] Jeffrey Heer, Nicholas Kong, and Maneesh Agrawala. 2009. Sizing the horizon: The effects of chart size and layering on the graphical perception of time series visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Boston, MA, USA, 1303–1312.

[22] Jeffrey Heer, Fernanda B Viégas, and Martin Wattenberg. 2007. Voyagers and voyeurs: Supporting asynchronous collaborative information visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, San Jose, CA, USA, 1029–1038.

[23] Youssef Hmamouche, Piotr Marian Przymus, Hana Alouaoui, Alain Casali, and Lotfi Lakhal. 2019. Large Multivariate Time Series Forecasting: Survey on Methods and Scalability. In *Utilizing Big Data Paradigms for Business Intelligence*. IGI Global, Hershey, PA, USA, 170–197.

[24] Sui-Lau Ho, Min Xie, and Thong Ngee Goh. 2002. A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction. *Computers & Industrial Engineering* 42, 2-4 (2002), 371–375.

[25] Tin Kam Ho. 1995. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, Vol. 1. IEEE, Montreal, Quebec, Canada, 278–282.

[26] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[27] Rob J Hyndman and Anne B Koehler. 2006. Another look at measures of forecast accuracy. *International Journal of Forecasting* 22, 4 (2006), 679–688.

[28] Tobias Isenberg, Petra Isenberg, Jian Chen, Michael Sedlmair, and Torsten Möller. 2013. A systematic review on the practice of evaluating visualization. *IEEE Transactions on Visualization and Computer Graphics* 19, 12 (2013), 2818–2827.

[29] Søren Johansen. 1995. *Likelihood-based inference in cointegrated vector autoregressive models*. Oxford University Press on Demand, Oxford, England.

[30] Michael J Kane, Natalie Price, Matthew Scotch, and Peter Rabinowitz. 2014. Comparison of ARIMA and Random Forest time series models for prediction of avian influenza H5N1 outbreaks. *BMC Bioinformatics* 15, 1 (2014), 276.

[31] Few,Stephen. 2008. Time on the Horizon Visual. Last accessed on 12/31/2020 at https://www.perceptualedge.com/articles/visual_business_intelligence/time_on_the_horizon.pdf.

[32] Hyndman, Rob J, Anne B Koehler, J Keith Ord, and Ralph D Snyder. 2008. *Forecasting with exponential smoothing: The state space approach*. Springer Science & Business Media, Berlin, Germany.

[33] Munroe, Randall. 2009. Xkcd# 657: Movie narrative charts.

[34] Ummul Khair, Hasanul Fahmi, Sarudin Al Hakim, and Robbi Rahim. 2017. Forecasting error calculation with mean absolute deviation and mean absolute percentage error. In *Journal of Physics: Conference Series*, Vol. 930. IOP Publishing, PA, USA, 012002.

[35] Josua Krause, Adam Perer, and Enrico Bertini. 2014. INFUSE: Interactive feature selection for predictive modeling of high dimensional data. *IEEE Transactions on Visualization and Computer Graphics* 20, 12 (2014), 1614–1623.

[36] Josua Krause, Adam Perer, and Kenney Ng. 2016. Interacting with predictions: Visual inspection of black-box machine learning models. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, San Jose, CA, USA., 5686–5697.

[37] Bum Chul Kwon, Min-Je Choi, Joanne Taery Kim, Edward Choi, Young Bin Kim, Soonwook Kwon, Jimeng Sun, and Jaegul Choo. 2018. RetainVis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2018), 299–309.

[38] Heidi Lam, Enrico Bertini, Petra Isenberg, Catherine Plaisant, and Sheelagh Carpendale. 2011. Seven guiding scenarios for information visualization evaluation. *HAL* 2011, 992 (2011), 04.

[39] Philip A Legg, David HS Chung, Matthew L Parry, Mark W Jones, Rhys Long, Iwan W Griffiths, and Min Chen. 2012. MatchPad: Interactive glyph-based visualization for real-time sports performance analysis. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, Vienna, Austria, 1255–1264.

[40] Gordon Leitch and J Ernest Tanner. 1991. Economic forecast evaluation: Profits versus the conventional error measures. *The American Economic Review* 81 (1991), 580–590.

[41] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*. NIPS, Long Beach, CA, USA, 4765–4774.

[42] Helmut Lütkepohl. 2013. Vector autoregressive models. In *Handbook of Research Methods and Applications in Empirical Macroeconomics*. Edward Elgar Publishing, Cheltenham, UK.

[43] Ganapathy Mahalakshmi and S.Sridevi and Rajaram, Shyamsundar. 2016. A survey on forecasting of time series data. In *Proceedings of the International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE)*. IEEE, Kovilpatti, India, 1–8.

[44] Daniel J McDonald, Cosma Shalizi, and Mark Schervish. 2012. Time series forecasting: model evaluation and selection using nonparametric risk bounds. *CoRR* abs (2012), 1212.0463.

[45] José Maria P Menezes Jr and Guilherme A Barreto. 2008. Long-term time series prediction with the NARX network: An empirical evaluation. *Neurocomputing* 71, 16-18 (2008), 3335–3343.

[46] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*. IEEE, Florence, Italy, 33–38.

[47] Tarun K Mukherjee and Atsuyuki Naka. 1995. Dynamic relations between macroeconomic variables and the Japanese stock market: An application of a vector error correction model. *Journal of Financial Research* 18, 2 (1995), 223–237.

[48] Wolfgang Müller and Heidrun Schumann. 2003. Visualization for modeling and simulation: Visualization methods for time-dependent data-an overview. In *Proceedings of the 35th Winter Simulation Conference: Driving Innovation*. Winter Simulation Conference, New Orleans, Louisiana, USA, 737–745.

[49] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. (2016). arXiv:1609.03499

[50] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should I trust you" Explaining the predictions of any classifier. In *Proceedings of the 22nd SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, San Francisco, CA, USA, 1135–1144.

[51] Takafumi Saito, Hiroko Nakamura Miyamura, Mitsuyoshi Yamamoto, Hiroki Saito, Yuka Hoshiya, and Takumi Kaseda. 2005. Two-tone pseudo coloring: Compact visualization for one-dimensional data. In *Symposium on Information Visualization*. IEEE, Minneapolis, MN, USA], 173–180.

[52] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. 2017. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. (2017). arXiv:1708.08296

[53] Ramesh Sharda and Rajendra B Patil. 1992. Connectionist approach to time series prediction: An empirical test. *Journal of Intelligent Manufacturing* 3, 5 (1992), 317–323.

[54] Qiaomu Shen, Yanhong Wu, Yuzhe Jiang, Wei Zeng, KH Alexis, Anna Vianova, and Huamin Qu. 2020. Visual Interpretation of Recurrent Neural Network on Multi-dimensional Time-series Forecast. In *2020 IEEE Pacific Visualization Symposium (PacificVis)*. IEEE, Tianjin, China, 61–70.

[55] Ben Shneiderman. 2003. The eyes have it: A task by data type taxonomy for information visualizations. In *The Craft of Information Visualization*. Elsevier, San Francisco, CA, USA, 364–371.

[56] Tom AB Snijders. 1988. On cross-validation for predictor evaluation in time series. In *On Model Uncertainty and Its Statistical Implications*. Springer, Berlin, Heidelberg, 56–69.

[57] Eric Stellwagen, Len Tashman, et al. 2013. ARIMA: The models of Box and Jenkins. *Foresight: The International Journal of Applied Forecasting* 30 (2013), 28–33.

[58] Dong Sun, Zezheng Feng, Yuanzhe Chen, Yong Wang, Jia Zeng, Mingxuan Yuan, Ting-Chuen Pong, and Huamin Qu. 2020. DFSeer: A Visual Analytics Approach to Facilitate Model Selection for Demand Forecasting. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Honolulu, HI, USA, 1–13.

[59] E Toth, A Brath, and A Montanari. 2000. Comparison of short-term rainfall prediction models for real-time flood forecasting. *Journal of hydrology* 239, 1-4 (2000), 132–147.

[60] Michail Vlachos, Philip Yu, and Vittorio Castelli. 2005. On periodicity detection and structural periodic similarity. In *Proceedings of the SIAM International Conference on Data Mining*. SIAM, Newport Beach, CA, USA,, 449–460.

[61] Martin Wattenberg and Jesse Kriss. 2006. Designing for social data analysis. *IEEE Transactions on Visualization and Computer Graphics* 12, 4 (2006), 549–557.

[62] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc, Montreal, Quebec, Canada, 802–810.