Research in Astronomy and Astrophysics, 22:125003 (14pp), 2022 December https://doi.org/10.1088/1674-4527/ac977b

Data-driven Seeing Prediction for Optics Telescope: from Statistical Modeling, Machine Learning to Deep Learning Techniques

Wei-Jian Ni[1], Quan-Le Shen[1], Qing-Tian Zeng[1], Huai-Qing Wang[2], Xiang-Qun Cui[2], and Tong Liu[1]
[1] College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266510, China; qtzeng@sdust.edu.cn, liu_tongtong@foxmail.com
[2] Nanjing Institute of Astronomical Optics & Technology, National Astronomical Observatories, Nanjing 210042, China
Received 2022 January 10; revised 2022 July 23; accepted 2022 September 27; published 2022 November 11

Abstract

Predicting seeing of astronomical observations can provide hints of the quality of optical imaging in the near future, and facilitate flexible scheduling of observation tasks to maximize the use of astronomical observatories. Traditional approaches to seeing prediction mostly rely on regional weather models to capture the in-dome optical turbulence patterns. Thanks to the developing of data gathering and aggregation facilities of astronomical observatories in recent years, data-driven approaches are becoming increasingly feasible and attractive to predict astronomical seeing. This paper systematically investigates data-driven approaches to seeing prediction by leveraging various big data techniques, from traditional statistical modeling, machine learning to new emerging deep learning methods, on the monitoring data of the Large sky Area Multi-Object fiber Spectroscopic Telescope (LAMOST). The raw monitoring data are preprocessed to allow for big data modeling. Then we formulate the seeing prediction task under each type of modeling framework and develop seeing prediction models through using representative big data techniques, including ARIMA and Prophet for statistical modeling, MLP and XGBoost for machine learning, and LSTM, GRU and Transformer for deep learning. We perform empirical studies on the developed models with a variety of feature configurations, yielding notable insights into the applicability of big data techniques to the seeing prediction task.

Key words: methods: data analysis – methods: statistical – telescopes

1. Introduction

Observation quality has been a key issue for both the construction and daily operations of optical telescopes. Astronomers and observers are much concerned about improving observation quality such that high-quality observational data can be collected in an efficient way (García-Lorenzo et al. 2009; Qian et al. 2018). As the hardware of an observatory is relatively immobile, the observation quality is mostly determined by astronomical observing conditions such as temperature, humidity and atmospheric pressure (Zhang et al. 2015). In general, small-scale and irregular fluctuation of air temperature or moisture may result in atmospheric turbulence that can disturb the propagation of light through the atmosphere and thus degrade the resolution of ground-based telescopes (Roddier 1981; Brunner 1982).

Seeing is an important parameter of the quality of astronomical images. In essence, it quantifies the blurring of the image of an astronomical object due to turbulent airflows in the atmosphere of Earth (Chromey 2016). One common measure of seeing is the angular diameter of the long-exposure image of a star, which is calculated by the full width at half maximum of its optical intensity (Tokovinin 2002). Practically, Differential Image Motion Monitor (DIMM) (Vernin & Munoz-Tunon 1995; Sarazin 1997; Liu et al. 2010) has been widely used for assessing astronomical seeing for its high accuracy and ease of operation and maintenance.

Monitoring seeing is of paramount importance for most ground-based optical astronomy observatories (Roddier et al. 1990; Benkhaldoun et al. 2005; Xin et al. 2020). Seeing can be used as an indicator for site characterization (Bradley et al. 2006) and observation scheduling (Tsapras et al. 2009). As different types of observations require particular seeing conditions, observation plans should be adjusted to the changes of seeing to ensure that the most suitable observations are performed under given seeing conditions, thus maximizing the use of instruments. Furthermore, the knowledge of seeing can help design and optimize telescope instruments such as the ventilation system (Good et al. 2003) and the temperature control system (Miyashita et al. 2003).

More recently, seeing prediction has been attracting an increasing amount of research attention in the optical astronomy community. For example, the Maunakea observatories conducted quantitative analysis of the impacts of various factors on seeing including wind shear and atmospheric stability patterns (Lyman et al. 2020), and realized a forecast of the nightly average seeing with machine learning techniques (Cherubini et al. 2021); the

Paranal observatory built a short-term turbulence prediction system by using three machine learning algorithms, i.e., random forests, multilayer perceptron and long-short term memory (Milli et al. 2019); Kornilov approached forecasting seeing as an auto-regressive problem and the linear auto-regressive integrated moving average (ARIMA) model is used for forecasting (Kornilov 2016); Giordano et al. demonstrated the importance of the local specificity of a given site in predicting seeing (Giordano et al. 2021). Seeing prediction is considered as a substantial utility for building and upgrading modern optical telescopes as it is a key to enabling the so-called "flexible scheduling" of astronomical observation activities. Astronomy telescopes, as precious scientific facilities, are commonly receiving a large number of observation requests and even oversubscribed. Thus, the scheduling system is vital to ensure the most efficient use of observation time and maximize scientific yields. If some degree of predictions of seeing of a telescope is possible, more suitable observation activities, of which the required observation conditions will be just satisfied in the near future, can be pre-scheduled. This can result in a more robust and dynamic scheduling, in contrast to the traditional post-scheduler where observation activities have to be passively adapted to observation conditions.

In this paper, we aim to explore predictive analysis of seeing of optical telescopes. Motivated by recent success of big data techniques in a wide array of scientific domains such as molecular biology (López-Rubio & Ratti 2021) and climate science (Dueben & Bauer 2018), we focus on data-driven prediction of seeing by a systematic exploitation of statistical modeling, machine learning and deep learning techniques. This study makes use of the monitoring data of the Large sky Area Multi-Object fiber Spectroscopic Telescope (LAMOST), including the historical meteorological data, interior sensor data and seeing measurements. A series of seeing prediction models are constructed to equip LAMOST with a certain ability of predictive monitoring of seeing during night-time observations.

As a data-driven study, we first construct the data set that can be used for developing seeing prediction models with big data techniques. Through correlation analysis on the LAMOST's monitoring data, we select a number of data fields highly related to the target seeing as the feature variables for seeing prediction. The monitoring data are collected from different sources (e.g., all-sky cameras, in-dome temperature sensors and DIMM), heterogenous in data type, value range, sampling frequency, etc. We thus perform preprocessing on the raw monitoring data and generate normative data sets that can facilitate the application of various data-driven approaches.

Next, we use diverse types of big data techniques to develop seeing prediction models. Specifically, two traditional statistical models (i.e., ARIMA and Prophet), two traditional machine learning algorithms (i.e., multilayer perceptrons and XGBoost), and three emerging deep learning algorithms (i.e., LSTM, GRU and Transformer) are exploited in this study. We also explore the combination of different types of big data techniques to achieve better prediction performance. The seeing prediction task is formulated under each type of these big data frameworks and then the corresponding predictive model is developed.

Through extensive empirical study on the LAMOST's seeing monitoring data set, we systematically evaluate the effectiveness of data-driven seeing prediction approaches, as well as the importance of feature variables in seeing prediction. The hybrid approaches to integrating machine learning and deep learning achieve the best performance, while XGBoost, as an elaborate machine learning algorithm, is surprisingly competitive for the seeing prediction task.

The rest of the paper is arranged as follows. Section 2 describes the data sources and preprocessing of the data set. Section 3 presents the formulations and models of seeing prediction under various big data frameworks, while Section 4 reports the empirical results of each model. Finally, we summarize the findings with thoughts for future work in Section 5.

2. Dataset Construction

2.1. Data Sources

In this study, we use the monitoring data of the Large sky Area Multi-Object fiber Spectroscopic Telescope (LAMOST, also known as the Guo Shou Jing Telescope) (Cui et al. 2012) as the data source. LAMOST is installed with astronomical site monitoring systems and in-dome sensing systems to monitor indispensable information about observation conditions and the status of astronomical facilities. According to previous theoretical and empirical studies (Coulman 1985; Coulman et al. 1986), astronomical seeing is mostly determined by meteorological status and the atmosphere in the immediate locality of the telescope and its dome. Therefore, while being subject to the data privacy issue, we have access to use the following types of monitoring data in this study.

2.1.1. Weather Data

An automatic weather station is set up as one of the infrastructure facilities of LAMOST that monitors a number of atmospheric parameters including humidity, air temperature and pressure, wind speed and direction, cloud cover, etc. Table 1 shows the metadata of the weather data used in this study.

2.1.2. Interior Temperature Sensing Data

As the thermal monitoring system, a number of temperature sensors are deployed at various positions inside LAMOST, e.g., under No.1 sub-mirror of the primary mirror, and on the east of the focal surface. Because the temperature fluctuation is

Table 1
The Metadata of the Weather Dataset

	Field	Range	Unit	Example
1	Temperature	[−15,35]	°C	21.61
2	Relative humidity	[10,100]	⋯	59.72
3	Air pressure	[890,950]	mb	902.37
4	Wind speed	[0,20]	m s^{-}	2.01
5	Wind direction	[0, 205]	F	130.7
6	Dew point	[0, 15]	°C	14.84
7	Rain probability	[0,100]	%	11.21
8	Timestamp	⋯	⋯	190827232601

one important factor in air turbulence that can affect seeing conditions, we choose to use temperature sensing data as feature variables, of which the metadata is shown in Table 2.

2.1.3. Seeing Data

The collected seeing data of LAMOST comprises two elements: the site seeing and the total seeing. Specifically, the site seeing is mainly determined by the atmospheric conditions of the astronomical site of LAMOST (i.e., Xinglong Observatory), and monitored by using a DIMM telescope (Liu et al. 2010); the total seeing is measured according to the imaging qualities of four guiding cameras located on the focal surface of LAMOST (illustrated in Figure 1). In order to obtain the total seeing of observations of LAMOST, we adopt the FWHM (Full Width at Half Maximum) method Tokovinin (2002), of which the calculation pipeline is depicted in Figure 2. There are specifically five stages in the pipeline:

1. For each of the four guiding cameras, observers first select a certain number of unsaturated bright stars (i.e., those with relatively low magnitude, but not over-exposed) in the sky area where its assigned guiding star is located.
2. The images of selected unsaturated bright stars are captured by guiding cameras, usually with an exposure time between 20 and 30 s.
3. Based on the images of selected unsaturated bright stars, the FWHM of each representative star is calculated by using Source Extractor (SExtractor) software packages.
4. To reduce the bias of single observations, the FWHMs of all the unsaturated bright stars selected by the four guiding cameras are averaged.
5. The raw value of the mean FWHM (in pixel) is transformed into the value of seeing (in arc-second).

The metadata of the seeing data set are shown in Table 3. Here, the total seeing reflects the overall observation quality, which is determined by various factors including seeing conditions, optical system, etc. Thus, in general, Site seeing ⩽Total seeing. In this study, the site seeing is taken as one of

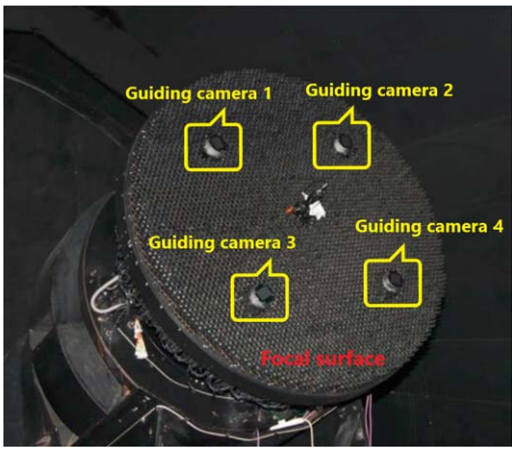


Figure 1. The four guiding cameras of LAMOST.[1]

Table 2
The Metadata of the Interior Temperature Dataset

	Field	Range	Unit	Example
1	Sensor Id	⋯	⋯	108108
2	Temperature	[−15, 35]	°C	4.75
3	Position	⋯	⋯	West of FP bracket
4	Timestamp	⋯	⋯	191115145005

the feature variables and the total seeing as the target variable; that is, we aim to predict total seeing by learning from historical monitoring data composed of the above feature variables.

2.2. Data Preprocessing

Before applying big data techniques, an indispensable step is to preprocess the collected monitoring data. In particular, we perform two main types of preprocessing, i.e., normalization and alignment, on the above data sets.

2.2.1. Normalization

As shown in Tables 1, 2 and 3, the fields of the data set are of different value ranges and units. For example, the values of seeing range in [0, 5], while that of air pressure range in [890, 950]. Because many machine learning algorithms are sensitive to data scales (Singh & Singh 2020), we apply min-max feature scaling on each field of the monitoring data set, which can be formulated as:

$$x' = \frac{x - \min_x}{\max_x - \min_x} \quad (1)$$

where x and x' are the values before and after normalization. $\min_x$ and $\max_x$ are the minimum and maximum values of the corresponding field of x.

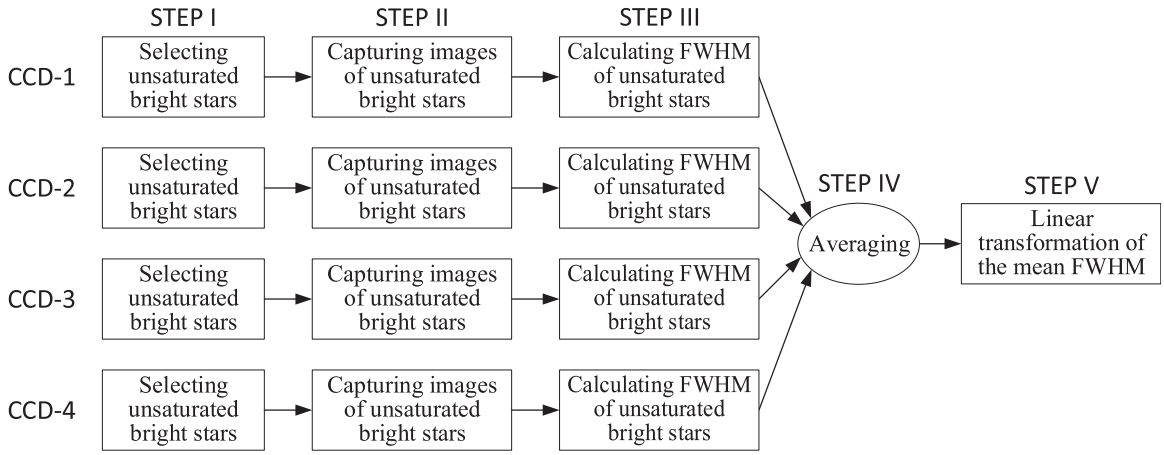


Figure 2. Total seeing calculation pipeline.

Table 3
The Metadata of the Seeing Dataset

	Field	Range	Unit	Example
1	Site seeing	[0, 5]	Arc second	1.17
2	Total seeing	[0, 5]	Arc second	2.48
3	Timestamp	···	···	191118201820

2.2.2. *Synchronization*

The weather station, temperature sensors and DIMM telescopes of LAMOST operate independently with different configurations (e.g., sampling rates), making the monitoring data collected in an asynchronous mode. One direct consequence is that most monitoring data seem to be missing if queried at a specified point of time. Figure 3 illustrates an example of asynchronous monitoring data from three sensors. As shown in Figure 3(a), only the data of sensor A are available at the query time 22:20:03, whereas the data of sensors A and B are missing due to different sampling rates. To facilitate further analysis and modeling, it is thus necessary to synchronize all the sensing samples. To this end, we first specify a fixed sampling rate (e.g., 10 s in Figure 3(b) depicted by red frame), then for each source of the monitoring data, calculate a representative value for each sampling period. Actually, there are three possible cases to be considered, which are illustrated by sensors A, B and C in Figure 3(b), respectively:

Case A Single sample in the given sampling period. As the simplest case, the value of the single sample will be selected as the representative value.

Case B No sample in the given sampling period. The average of the values of nearby samples will be used as the representative value. To be more specific, the two most proximate samples before and after the given sampling period will be selected for calculation. Note that the nearby sample will be discarded if its proximity to the given sampling period exceeds a specified threshold.

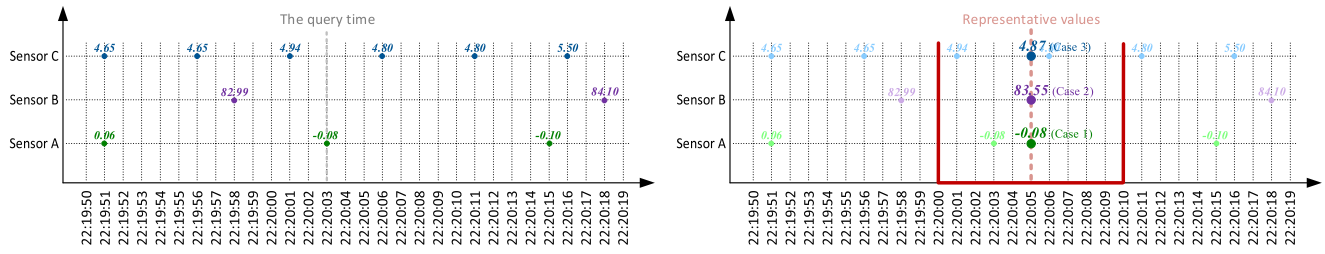
Case C Multiple samples in the given sampling period. The average of the values of internal samples will be used as the representative value.

After calculating representative values, we can generate a normative data set in the sense that each field of the monitoring data mentioned in Section 2.1 synchronously takes one specific value in each sampling period.

2.3. *Temperature Difference Feature Generation*

Air of a stable temperature, whether cold or hot, does not necessarily lead to turbulence. Exactly, small-scale and irregular fluctuation of temperature, rather than the level of temperature, affects seeing conditions. In order to directly introduce the temperature fluctuation information into big data modeling, we calculate the difference in temperature between every pairs of temperature sensors so as to generate the temperature difference feature variables, in addition to the temperature feature variables shown in Table 2. Actually, there are more than two hundred temperature sensors deployed inside LAMOST, resulting in a huge number of sensor pairs (∼50,000) to be considered when calculating temperature difference features. This will inevitably bring a great computational burden when applying machine learning or deep learning algorithms, and more seriously, causes the so-called 'the curse of dimensionality' (Verleysen & François 2005) that can result in poor prediction performance.

[1] The guiding cameras of LAMOST adopt the full-frame CCD architecture. The total number of pixels is 2092[H] × 2093[V], and the size of each pixel is 24μm[H] × 24μm[V].



(a) The raw data (b) The aligned data (with a sampling frequency of 10s)

Figure 3. An example of asynchronous monitoring data.

In this study, we perform feature selection on the massive temperature difference features to retain the most relevant ones for seeing prediction. Specifically, we calculate the Pearson correlation coefficient between either temperature features or temperature difference features and the total seeing on the LAMOST's monitoring data set, which can be formulated as:

$$r_{x,y} = \frac{\sum_t (x_t - \bar{x})(y_t - \bar{y})}{\sqrt{\sum_t (x_t - \bar{x})^2}\sqrt{\sum_t (y_t - \bar{y})^2}} \quad (2)$$

where x_t is a value of a temperature/temperature difference feature at timestamp t, and y_t is the value of total seeing at the same time. $\bar{x}$ and $\bar{y}$ are the average value of the feature and seeing, respectively.

Then, we select the N temperature difference features with the highest Pearson correlation coefficients for the subsequent seeing prediction modeling.

To verify the usefulness of temperature difference features, we depict the Pearson correlation coefficients of all the temperature features and the top-500 temperature difference features on the LAMOST monitoring data set, as shown in Figure 4. Both of these two types of features are sorted according to the Pearson correlation coefficients. It can be seen that there is a significant margin between the two curves of temperature difference features and temperature features. In fact, any of the top-500 temperature difference features are much more relevant w.r.t. seeing than all temperature features. This provides empirical evidence for the rationale of generating temperature difference features.

Furthermore, we investigate the redundancy of temperature-related features. The Pearson correlation coefficients of feature pairs within either temperature features and temperature difference features are calculated and shown in Figures 5(a) and (b), respectively. It is apparent that the correlations among temperature difference features are much lower than that of temperature features.

3. Seeing Prediction Modeling

Based on the LAMOST's monitoring data set, we explore the use of a series of big data techniques to construct seeing prediction models, in which the weather conditions, site seeing, and interior temperature information (including temperature values and their differences) are treated as feature variables, and the total seeing as the target variable. Following the development path of big data techniques, we choose to employ three types of big data techniques, i.e., statistical model, machine learning and deep learning, in this study. Furthermore, we also develop hybrid models by combining heterogeneous techniques for the seeing prediction task.

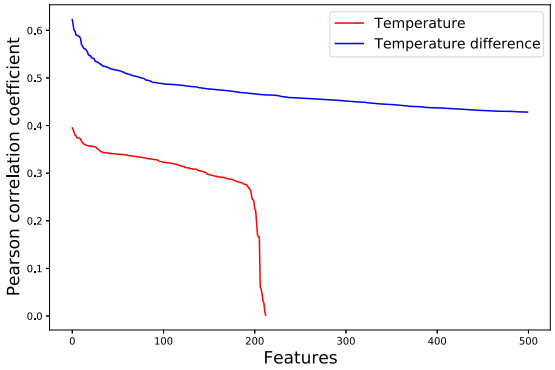
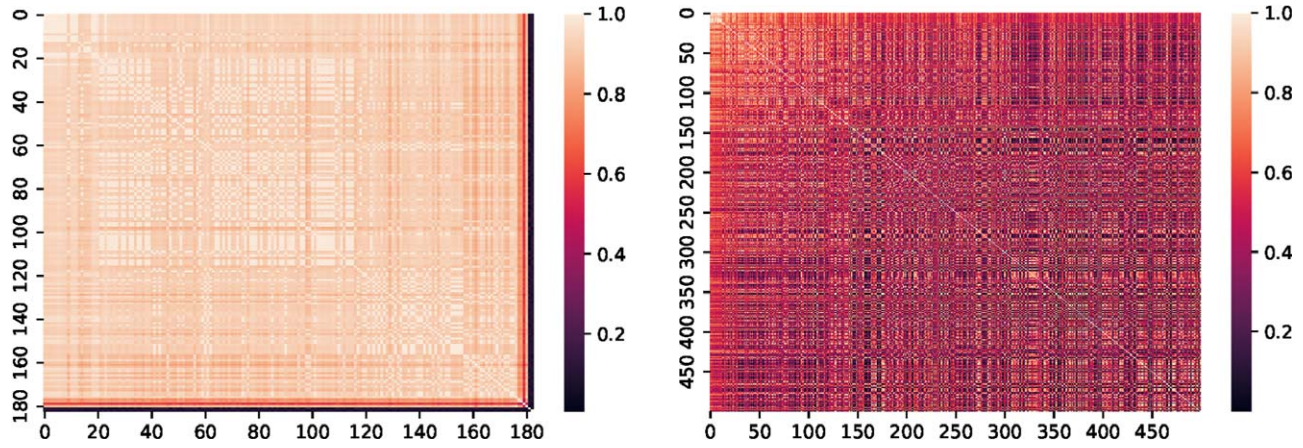


Figure 4. The relevance between temperature-related features and seeing.

3.1. Statistical Modeling

As most advanced big data techniques derive from statistical theories and models Franke et al. (2016), we start our exploration on seeing prediction from the traditional statistical modeling perspective. In essence, the seeing data of optical telescopes can be viewed as a type of time series data because the values of seeing evolve over time and exhibit temporal correlations. Thus we employ a classical statistical modeling technique—the autoregressive model. Under the autoregressive framework, the prediction model outputs seeing at a future timestamp according to historical seeing. Formally, seeing prediction can be written as:

$$\hat{y}_t = f_{\mathrm{sm}}(y_{t-1}, \cdots, y_{t-\delta}) \quad (3)$$



(a) Temperature features (b) Temperature difference features

Figure 5. Correlation analysis of temperature-related features.

where t is the target timestamp, and $\{y_{t-1}, \cdots, y_{t-\delta}\}$ are the seeing observations in a recent window before t with the size of δ.

We specifically used two types of autoregressive methods—ARIMA (auto-regressive integrated moving average) and Prophet—to construct seeing prediction models.

3.1.1. ARIMA

The ARIMA model is a traditional statistical analysis model for time series data (Box et al. 2015), and has wide applications in predicting time series from social, economic and engineering areas, due to its flexible modeling capability. The ARIMA model is essentially a combination of the differenced autoregressive model (AM) with the moving average model (MA), which can be formulated as:

$$\nabla^d y_t = \sum_{i=1}^{p} \alpha_{t-i} \cdot \nabla^d y_{t-i} + \epsilon_t + \sum_{i=1}^{q} \beta_{t-i} \cdot \epsilon_{t-i} \quad (4)$$

where $\{y_{t-1}, \cdots, y_{t-p}\}$ are the p lagged observations called "autoregressive" terms, and $\{\epsilon_{t-1}, \cdots, \epsilon_{t-q}\}$ are the q random errors of the respective lags called "moving average" terms. $\{\alpha_{t-1}, \cdots, \alpha_{t-p}\}$ and $\{\beta_{t-1}, \cdots, \beta_{t-q}\}$ are the corresponding coefficients to be estimated, respectively. $\nabla^d y_t$ denotes the d-th order differences of y_t that can be written recursively:

$$\nabla^d y_t = \nabla^{d-1} y_t - \nabla^{d-1} y_{t-1} \quad (5)$$

In particular, $\nabla^1 y_t = y_t - y_{t-1}$, and $\nabla^0 y_t = y_t$.

3.1.2. Prophet

Prophet, proposed by Facebook in 2017, is a popular time series prediction model that is powerful in handling daily periodicity with large outliers and shifts in trends, as well as multiple periods of seasonality (Taylor & Letham 2018). The Prophet model elaborately approaches time series prediction by a combination of three functions of time plus an error term:

$$\hat{y}_t = g(y_1, \cdots, y_{t-1}) + s(y_1, \cdots, y_{t-1}) + h(y_1, \cdots, y_{t-1}) + \epsilon_t \quad (6)$$

The growth function $g(\cdot)$ models the non-periodic trend of the data, which has further three options: linear growth, logistic growth, and flat (no growth over time). The seasonality function $s(\cdot)$ approximates arbitrary smooth seasonal effects by using a Fourier series. The holiday function $h(\cdot)$ is used to adjust predictions when a holiday or major event may affect the prediction. The Prophet model is estimated in a fully Bayesian manner to allow for automatically identifying model characteristics such as the selection of changepoints.

3.2. Machine Learning

Traditional autoregression models like ARIMA and Prophet make seeing predictions by using observations of seeing from previous time steps; however, they are insufficient in terms of exploiting the feature observations (i.e., meteorological status and temperature sensings) of time series data, besides the target seeing. To address this deficiency, we leverage machine learning approaches to incorporate various feature observations related to the target seeing. With similar notations in Equation (3), seeing prediction under the machine learning framework can be formulated as:

$$\hat{y}_t = f_{\mathrm{ml}}(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}, y_{t-1}, \cdots, \boldsymbol{x}_{t-\delta}, y_{t-\delta}) \quad (7)$$

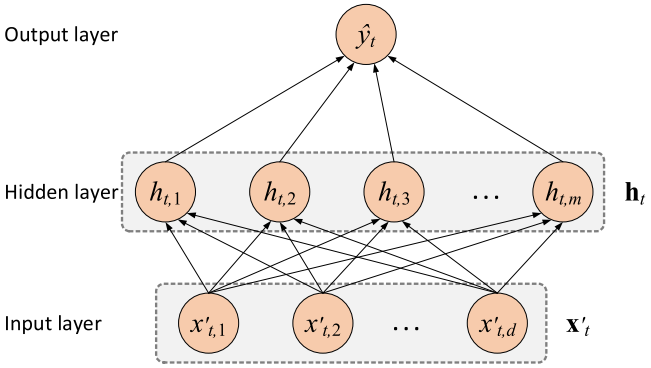


Figure 6. The structure of 3-layer MLP.

where $\boldsymbol{x}_t \in \mathbb{R}^d$ is the feature vector at the target timestamp, and d is the number of features. Here, to incorporate historical information, the feature and seeing observations, denoted as $\boldsymbol{x}_{t-i}$ and $y_{t-i}\,(1 \leqslant i \leqslant \delta)$ are also used as input variables for machine learning models. To ease notations in the rest of this paper, let $\boldsymbol{x}'_{t\sim\delta} = (\boldsymbol{x}_t, \boldsymbol{x}_{t-1}, y_{t-1}, \cdots, \boldsymbol{x}_{t-\delta}, y_{t-\delta})$.

We specifically used two types of machine learning methods —MLP (multilayer perceptron) and XGBoost (extreme gradient boosting)—to construct seeing prediction models.

3.2.1. MLP

MLP is a typical class of feed-forward neural networks, which is also the building block of many advanced deep neural networks. An MLP model is comprised of three types of layers, i.e., one input layer, one or several hidden layers, and one output layer. Figure 6 depicts the structure of the 3-layer MLP adopted in this study. The prediction of MLP is obtained by a nonlinear mapping from an input feature vector to a corresponding output seeing, which can be formally written as:

$$\begin{aligned} \hat{y}_t &= W^{(2)} \cdot \boldsymbol{h}_t + \boldsymbol{b}^{(2)} \\ \boldsymbol{h}_t &= g(W^{(1)} \cdot \boldsymbol{x}'_{t\sim\delta} + \boldsymbol{b}^{(1)}) \end{aligned} \tag{8}$$

where $W^{(1)}$ and $W^{(2)}$ are the weight matrix parameters connecting neurons between layers, and $\boldsymbol{b}^{(1)}$ and $\boldsymbol{b}^{(2)}$ are the corresponding bias vector parameters. $g(\cdot)$ is the differentiable nonlinear activation function, e.g., sigmoid and ReLU.

3.2.2. XGBoost

XGBoost is a highly scalable boosting-tree-based machine learning model (Chen & Guestrin 2016). Its advantages have been widely recognized in a variety of machine learning tasks, as it has been a popular approach or indispensable component of winning solutions in many data science competitions. XGBoost is essentially a tree ensemble model in which a set of additive regression trees (CART tree in general) are used as base learners. The sum of all the base learners' outputs is taken as the final prediction. Formally, the prediction of XGBoost can be written as:

$$\hat{y}_t = \sum_{k=1}^{K} g_k(\boldsymbol{x}'_{t\sim\delta}) \tag{9}$$

where g_k is a regression tree model that is learned in an additive manner and can perform prediction independently. K is the number of regression trees assembled in the XGBoost model.

3.3. Deep Learning

In recent years, deep learning has started to revolutionize a broad range of data-driven research and applications, e.g., natural language processing, computer vision and e-commerce. Compared with traditional machine learning, deep learning is advantageous in modeling complex patterns inherent in vast amounts of unstructured data. As mentioned above, the seeing data are essentially a type of time series data. However, the machine learning formulation to seeing prediction, as shown in Equation (7), relies on hard feature engineering, i.e., manually specifying a fixed window size δ, to model the sequential patterns. In other words, the model capability is restricted by this parameter.

In contrast, deep learning allows for more natural modeling of the sequential seeing data. Formally, let $[\boldsymbol{x}_1, y_1; \cdots; \boldsymbol{x}_T, y_T]$ be an observation sequence across T time step, where $\boldsymbol{x}_t$ and y_t $(1 < t \leqslant T)$ are the feature observations and seeing observation at the t-th time step, respectively. Under the deep learning framework, seeing prediction at a given time step t can be formally written as:

$$\hat{y}_t = f_{\mathrm{dl}}(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}, y_{t-1}, \cdots, \boldsymbol{x}_1, y_1) \tag{10}$$

Note that in Equation (10), there is no need to specify the additional window size parameter δ as in Equation (7). Thus, deep learning has more potential than traditional machine learning to handle the arbitrarily long dependencies inherent in the seeing observations.

We specifically used three types of deep learning methods—LSTM (long short-term memory), GRU (gated recurrent unit) and Transformer—to construct seeing prediction models.

3.3.1. LSTM

One of the most traditional deep learning approaches to handling sequential data is recurrent neural networks, among which LSTM has the advantage of alleviating the vanishing gradient problem that can be encountered when training traditional recurrent neural networks (Hochreiter & Schmidhuber 1997). As a special type of recurrent neural networks, LSTM maintains a recurrent hidden state vector $\boldsymbol{h}_t$ at each time step t, and updates it in a chain-like manner:

$$\boldsymbol{h}_t = \mathrm{LSTM}(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t) \tag{11}$$

where $\boldsymbol{x}_t$ is the input feature vector at time step t.

To be more specific, the hidden state is updated by using one "**c**ell" and three "gates", namely the **i**nput gate, the **f**orget gate and the **o**utput gate, which can be formulated as:

$$\begin{aligned}
\boldsymbol{i}_t &= \sigma(W_1\boldsymbol{x}_t + V_1\boldsymbol{h}_{t-1} + b_1)\\
\boldsymbol{f}_t &= \sigma(W_2\boldsymbol{x}_t + V_2\boldsymbol{h}_{t-1} + b_2)\\
\boldsymbol{o}_t &= \sigma(W_3\boldsymbol{x}_t + V_3\boldsymbol{h}_{t-1} + b_3)\\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \sigma(W_4\boldsymbol{x}_t + U_4\boldsymbol{h}_{t-1} + b_4)\\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \phi(\boldsymbol{c}_t)
\end{aligned} \tag{12}$$

where $\sigma(\cdot)$ and $\phi(\cdot)$ denote the sigmoid and the hyperbolic tangent activation functions, respectively. $\odot$ denotes element-wise multiplication between vectors. The hidden state vector $\boldsymbol{h}_t$ is then fed into a MLP module (as shown in Equation (8)) to yield the seeing prediction at the time step t.

3.3.2. GRU

Similar to LSTM, GRU is another popular type of recurrent neural networks. It can be viewed as a simplified variant of LSTM by coupling the forget gate and the input gate into a single update gate and mixing the cell state and the hidden state as one state (Cho et al. 2014). In total, only two gates, namely the **u**pdate gate and the **r**eset gate, are used in GRU. The model formulation can be written as:

$$\begin{aligned}
\boldsymbol{u}_t &= \sigma(W_1\boldsymbol{x}_t + V_1\boldsymbol{h}_{t-1} + b_1)\\
\boldsymbol{r}_t &= \sigma(W_2\boldsymbol{x}_t + V_2\boldsymbol{h}_{t-1} + b_2)\\
\hat{\boldsymbol{h}}_t &= \phi(W_3\boldsymbol{x}_t + V_3(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + b_3)\\
\boldsymbol{h}_t &= (1 - \boldsymbol{u}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{u}_t \odot \hat{\boldsymbol{h}}_t
\end{aligned} \tag{13}$$

Existing empirical studies show the comparable performance of GRU to LSTM on many sequential data modeling tasks while offering improved computational efficiency, which motivates us to exploit GRU in the seeing prediction task.

3.3.3. Transformer

Transformer is a powerful sequence modeling approach that has lately attracted extensive interest. Unlike recurrent neural networks, Transformer capitalizes on multi-headed self-attention mechanism under the encoder-decoder architecture to learn the global sequential dependency without explicit recurrence mechanism (Vaswani et al. 2017). Transformer is now recognized to take the place of recurrent neural networks in sequence modeling applications due to its promising properties.

When applying Transformer on the seeing prediction task, the observation sequence is organized as a matrix $X = [\boldsymbol{x}_1;\cdots,\boldsymbol{x}_t] \in \mathbb{R}^{d\times t}$ (with sequence length t, and feature number d), and linearly projected into a query matrix, a key matrix and a value matrix, respectively:

$$Q,\ K,\ V = W_1 \cdot X,\ W_2 \cdot X,\ W_3 \cdot X \tag{14}$$

Then, a matrix representation of the input observation sequence is calculated by scaled dot-product operation:

$$\mathrm{head}(Q, K, V) = \mathrm{softmax}\left(\frac{Q \cdot K^{\top}}{\sqrt{d}}\right) \cdot V \tag{15}$$

where d is the feature dimensionality after linear projection.

Commonly, k parallel scaled dot-product operations, each called a "head", are performed. The independent outputs are concatenated and linearly transformed to yield the "multi-head" representation of the input observation sequence:

$$\mathrm{multi-head}(Q, K, V) = [\mathrm{head}_1,\cdots,\mathrm{head}_k] \cdot W_4 \tag{16}$$

Similar to recurrent neural networks, the seeing prediction result is output by an MLP module fed with the "multi-head" representation.

3.4. Hybrid Models

In many big data applications, combining multiple machine learning and deep learning approaches has been demonstrated empirically to provide much more accurate solutions. This motivates us to explore the use of hybrid models for the seeing prediction task, in addition to employing single techniques as mentioned above. As will be seen in Section 4, XGBoost and deep learning approaches perform competitively on the seeing prediction task. Therefore, we develop hybrid models by combining XGBoost with each of the deep learning models for further performance gains.

We specifically adopt two combination strategies. The first one is the result-level combination. That is, the prediction results of XGBoost and deep learning models are averaged to yield the final results:

$$\hat{y}_t = \frac{f_{\mathrm{xgb}}(\boldsymbol{x'}_{t\sim\delta}) + f_{\mathrm{dl}}(\boldsymbol{x'}_{t\sim(t-1)})}{2} \tag{17}$$

Here, the deep learning models f_{dl} here can in particular be either LSTM, GRU or Transformer described above.

To take further benefits of different models, the second strategy accomplishes model combination at the feature level. As shown in Equation (9), XGBoost is essentially an ensemble of regression trees $\{g_1, \cdots, g_k\}$, each of which can be considered as a compositional feature extracted from the original feature space. These features, as an abstraction of the observation sequence, are beneficial to learn downstream models. To this end, we take the outputs of every regression trees in the XGBoost model as additional features for deep learning models:

$$\hat{y}_t = f_{\mathrm{dl}}(\boldsymbol{x'}_{t\sim(t-1)}, g_1(\boldsymbol{x'}_{t\sim\delta}),\cdots,g_k(\boldsymbol{x'}_{t\sim\delta})) \tag{18}$$

Similarly as in Equation (17), f_{dl} in Equation (18) denotes deep learning models that can be either LSTM, GRU or Transformer.

Table 4
Configurations of Seeing Prediction Models

	Model	Hyperparameter Candidates	Developing Framework
1	ARIMA	p: {0, 1, 2, 3, 4}, d: {0, 1}, q: {0, 1, 2, 3, 4}	Pmdarima 1.8.4[a]
2	Prophet	Growth: {*linear*, *logistic*}, Seasonality: *True*	Prophet in Python 0.7.1[b]
3	XGBoost	Max tree depth: {3, 6}, Tree number: {50, 100, 200}	dmlc XGBoost 1.3.1[c]
4	MLP	Learning rate: {0, 01, 0.001, 0.0001},	
5	LSTM	Hidden dim: {32, 64, 128, 256}, Epoch: {100},	PyTorch 1.1.0[d]
6	GRU	Batch size: {8, 16}, Optimizer: Adam	
7	Transformer		

Notes.
[a] https://pypi.org/project/pmdarima/.
[b] https://facebook.github.io/prophet/.
[c] https://github.com/dmlc/xgboost.
[d] https://pytorch.org/.

4. Experimental Study

4.1. Settings

Data set. In this study, we collected the monitoring data of LAMOST ranging from 9/1/2020 to 10/31/2020. The data is processed as described in Section 2. In particular, the sample rate for synchronization is set to 30 s. The final data set used in the experiment contains a total of 23,472 synchronized observations, involving 160 separated observation sequences. In order to train and evaluate seeing prediction models, the data set is randomly split into a training set and a testing set containing 128 and 32 observation sequences, respectively.

Model configurations. We developed a variety of seeing prediction models based on open-source frameworks. Grid search over pre-specified hyperparameter candidates is applied to find optimal ones for all models. The configuration details of each model are given in Table 4.

Evaluation metric. To assess the accuracy of seeing prediction models, we used the well-known error metric for regression, i.e., Root Mean Square Error (RMSE). Formally, given an seeing observation sequence $Y = \{y_1, \cdots, y_n\}$ and its corresponding prediction results $\hat{Y} = \{\hat{y}_1, \cdots, \hat{y}_n\}$, the RMSE is calculated as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{19}$$

In general, lower RMSE indicates a more accurate seeing prediction model.

4.2. Results

We performed extensive experiments on the developed seeing prediction models with the following three aims:

1. Evaluating how well each big data technique can be applied for the seeing prediction task;
2. Evaluating the helpfulness of feature engineering of the monitoring data on boosting the seeing prediction accuracy;
3. Evaluating the training efficiency of seeing prediction models.

In what follows, we present the details of the three experiments conducted for each of the above three aims.

4.2.1. Experiment 1: Evaluation of Big Data Techniques

In our experimental study, we first developed seeing prediction models on the basic feature variables which are described in Section 2.1 Table 5 shows the performance of these seeing prediction models in terms of RMSE. The best performance of each type of model is indicated with black boldface. Note that as shown in Equation (7), machine learning models rely on a historical windows hyperparameter δ, thus we conduct experimental analysis with varying δ to make a fair comparison among different types of models. Here, $\delta = 0$ means that no historical monitoring information is considered for developing seeing prediction models, whereas $\delta =$ seq_len means that the whole monitoring sequence is used.

From Table 5, we can obtain the following observations:

1. Machine learning and deep learning models outperform autoregressive models by a substantial margin, which indicates the indispensability of incorporating multiple monitoring variables, e.g., the weather information and the temperature sensing information, in predicting total seeings. In fact, univariate autoregressive like ARIMA and Prophet may yield trivial solutions for the seeing prediction task.

Table 5
Performance Comparison among Seeing Prediction Models Developed on Basic Features

	Model	$\delta = 0$	$\delta = 1$	$\delta = 3$	$\delta = 7$	$\delta =$ seq_len
Autoregressive models	ARIMA	⋯	⋯	⋯	⋯	**0.1792**
	Prophet	⋯	⋯	⋯	⋯	0.2420
Machine learning models	MLP	0.2294	0.1710	0.1133	0.1243	⋯
	XGBoost	0.1586	0.1223	0.1119	**0.1057**	⋯
Deep learning models	LSTM	⋯	0.1266	0.1139	0.1176	0.1079
	GRU	⋯	0.1192	0.1150	0.1118	0.1099
	Transformer	⋯	0.1242	0.1188	0.1169	**0.0998**
Hybrid models (result-level)	Hybrid_res(XGB+LSTM)	⋯	0.1183	0.1085	0.1075	**0.1021**
	Hybrid_res(XGB+GRU)	⋯	0.1212	0.1101	0.1134	0.1035
	Hybrid_res(XGB+Transf)	⋯	0.1251	0.1172	0.1190	0.1028
Hybrid models (feature-level)	Hybrid_fea(XGB+LSTM)	⋯	0.1133	0.0962	0.0960	0.0907
	Hybrid_fea(XGB+GRU)	⋯	0.1130	0.0945	0.0947	0.0904
	Hybrid_fea(XGB+Transf)	⋯	0.1168	0.0979	0.0950	**0.0893**

2. Among all the single machine learning and deep learning models, XGBoost is very competitive when limited lengths of observation sequences (i.e., $\delta = 1$, 3, 7) are taken into account, even outperforming deep learning models in most cases; however, deep learning models outperform machine learning models if the whole observation sequence is used to construct seeing prediction models (i.e., $\delta =$ seq_len), suggesting the benefit of tackling seeing prediction by capture patterns of the whole sequences. If going into more details, we can find that Transformer exhibits superior performance among all the single models, which verifies the competitiveness of Transformer in sequence modeling.
3. The feature-level hybrid models consistently outperform the result-level ones and achieve the best performance among all the compared models. This result suggests that one best practice for seeing prediction would be to combine different types of models by using internal results of machine learning models as features for building deep learning models. In contrast, the result-level hybrid models perform quite poorly, even degrading the performance of the constituted models at times, giving evidence that simply assembling results of single models does not necessarily lead to improved performance of seeing prediction.

4.2.2. Experiment 2: Evaluation of Feature Variables

In our next experiment, we further incorporated temperature difference features when learning seeing prediction models. Figure 7 depicts the performance variations of the models with and without temperature difference features. Recall that the feature number is potentially huge, we developed seeing prediction models with three configurations of temperature difference features: the first one is the 100 randomly selected features, and the second and the third ones are the top-100 and 500 features with the highest Pearson correlation with the target seeing, respectively.

Temperature difference features are intuitively helpful in boosting seeing prediction accuracy; however, it is surprising to see from Figure 7 that the performance of seeing prediction models vary in a different manner. In particular, the performance of machine learning models is consistently improved after incorporating temperature difference features. The MLP model achieves the lowest RMSE at the top-100 feature configuration, whereas the XGBoost model at the top-500 feature configuration. In fact, the best result is achieved by XGBoost with window size hyperparameter $\delta = 7$ and top-500 feature configuration (RMSE=0.1023). Conversely, the performance variations of deep learning models are mostly negative. We only see performance improvements of the LSTM model with the top-100 feature configuration when $\delta = 7$, and the GRU model with the rand-100 feature configuration when $\delta =$ seq_len. For the Transformer model, its performance degrades significantly with every feature configuration. One reason could be that after incorporating temperature difference features, the input dimension of deep learning models increases greatly, especially in the case of the top-500 feature configuration, requiring more computational effort and extensive hyperparameter tuning for training highly accurate seeing prediction models. To make a fair comparison in our experiment, we use fixed hyperparameter ranges shown in Table 4 for training models with every feature setting, which would lead to suboptimal seeing prediction models. We defer a more systematic evaluation on the hyperparameters of deep learning models (e.g., the number of layers of encoder, and head of self-attention) to future work.

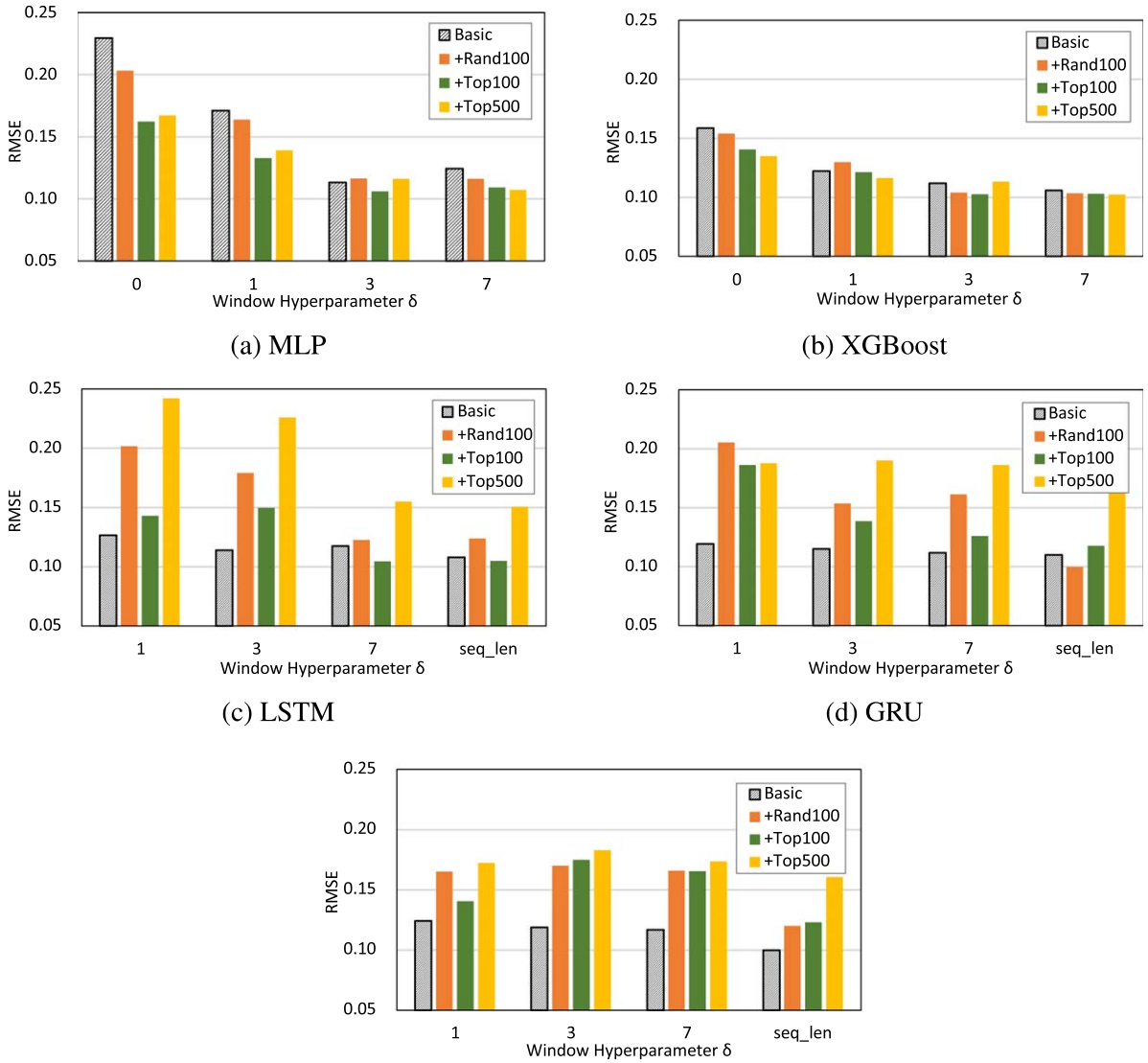


Figure 7. Performance comparison of seeing prediction models w/o temperature difference features.

It should be noted that despite the improvements by incorporating temperature difference features, the prediction accuracy of machine learning models is still lower than that of the best-performed deep learning models (i.e., 0.1023 by XGBoost versus 0.0998 by Transformer). This gives further evidence of the superiority of Transformer in the seeing prediction task.

4.2.3. Experiment 3: Evaluation of Learning Efficiency

Our efficiency experiments were centered around the comparison of the training time of different seeing prediction models. In particular, each type of seeing prediction model was trained with three feature configurations: (i) the basic features,

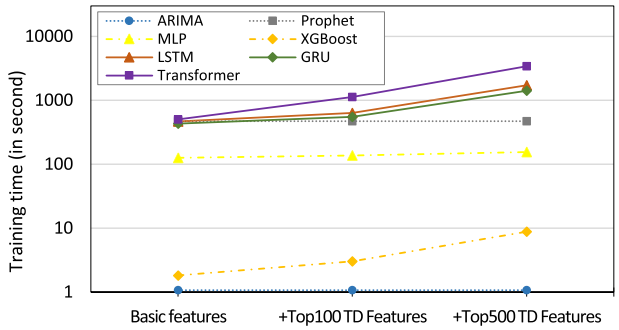
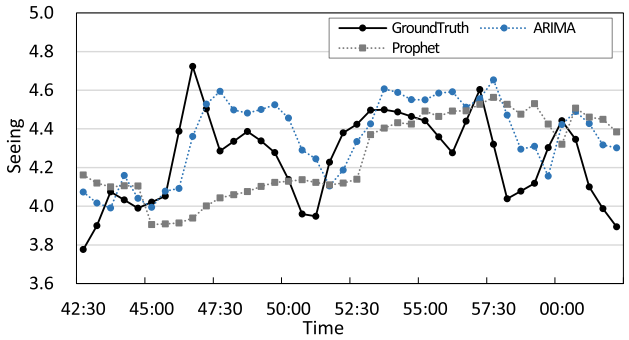
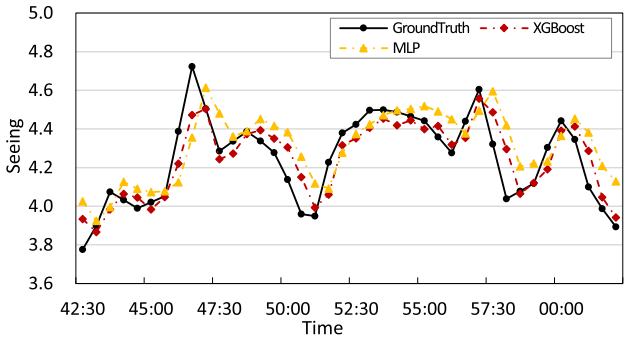


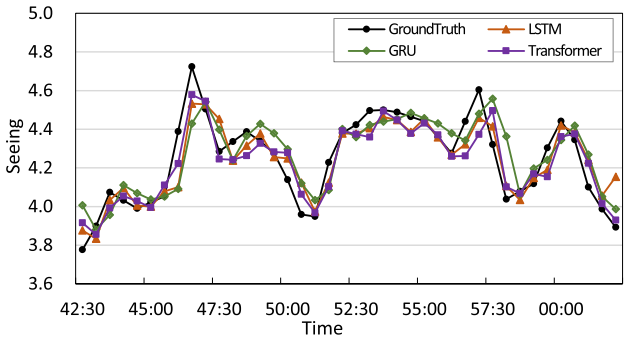
Figure 8. Training time of seeing prediction models.



(a) Predictions of statistical models



(b) Predictions of machine learning models



(c) Predictions of deep learning models

Figure 9. An example of seeing sequence with its predictions.

(ii) incorporating top-100 temperature difference (TD) features, and (iii) incorporating top-500 temperature difference features. Figure 8 depicts the training time of every seeing prediction model. Note that the training time is log-scaled to better portray the variation tendency among different configurations.

From Figure 8, we can observe that as the number of features increases the time required for training machine learning and deep learning models increases notably. Despite this, training machine learning models is still much more efficient than deep learning models. In fact, training the best-performed machine learning model (i.e., XGBoost by incorporating top-500 temperature difference features) can be completed in 10 s, whereas training the best performed deep learning model (i.e., Transformer with basic features) needs about 500 s. Recall the very competitive prediction accuracy of the XGBoost model, we suggest that the XGBoost method with elaborately selected features is a practical solution to the seeing prediction task.

4.3. Case Study

In order to give a more intuitive understanding of the behaviors of the developed seeing prediction models, we selected a seeing sequence from the LAMOST's monitoring data as an illustrative example. As shown in Figure 9, the illustrative example is a continuous observation sequence lasting 20 minutes. The total seeing monitored by guiding cameras is denoted as "GroundTruth", and the predictions of statistical models, machine learning models and deep learning models are depicted in Figures 9(a), (b) and (c), respectively.

It can be seen that the predictions of statistical models (i.e., ARIMA and Prophet) deviate from the ground truth of seeing substantially. Among machine learning models, XGBoost performs much better than MLP. There is particularly an apparent time lag between the predictions of the MLP models and the ground truth, partly accounting for its inferior prediction performance. The predictions of deep learning models are shown to better fit the ground truth, among which the Transformer model exhibits superiority to the other two deep learning models.

In order to demonstrate the effectiveness of each type of seeing prediction models in a more precious way, we further quantitatively evaluate the performance of the models in this case study. Figure 10 depicts the RMSE of every models as well as the standard derivations of prediction bias. It can be seen that among all the models, Transformer achieves the best performance, i.e., both the lowest RMSE and prediction bias, followed by LSTM and XGBoost; in contrast, Prophet and

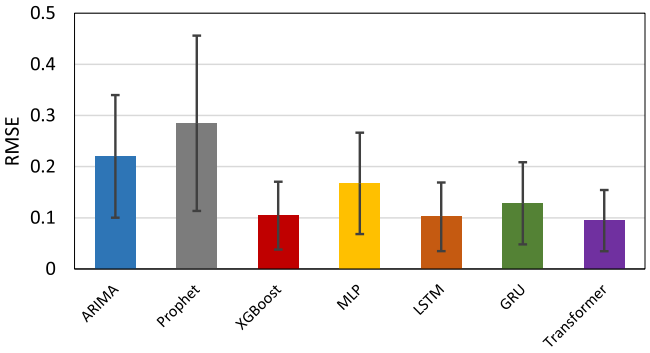


Figure 10. Quantitative performance (RMSE $\pm$ STD) of seeing prediction models in this case study.

ARIMA are the two models with the worst performance in terms of both RMSE and prediction bias. These performance differences of seeing prediction models, which largely correspond to that in experiment 1, can be intuitively reflected in Figure 9.

5. Conclusion and Future Work

In this paper, we systematically investigate big data techniques in predicting the seeing of astronomical observations. A variety of seeing prediction models are developed by leveraging representative statistical modeling, machine learning and deep learning methods. Model combination strategies are further proposed to derive more accurate seeing prediction models. By evaluating the developed models on LAMOST's monitoring data with different feature configurations, we can arrive at the following findings:

1. Incorporating internal outputs of XGBoost into deep learning models is the best practice for the seeing prediction task if the prediction accuracy is the main goal to be pursued.
2. Among all the simplex methods, Transformer achieves the highest prediction accuracy by only using the basic observation sequences, not relying on any additional feature engineering.
3. XGBoost exhibits competitive prediction accuracy if incorporated with the proposed temperature difference features, meanwhile being very efficient in terms of the training time. Thus the XGBoost model can be viewed as the most balanced one when both prediction accuracy and training efficiency are taken into account.

Data-driven approaches to seeing prediction, as a newly emerging research topic in the optical astronomy community, are far from being well addressed. There is particularly much room for improvement of deep-learning-based seeing prediction models. As shown in this work, deep learning does not show dominant advantages over traditional methods, which have been done in many domains like natural language processing and computer vision. One reason is that training deep learning models is computationally intensive, whereas the computational resources in our empirical study are relatively limited. We thus plan to conduct more extensive empirical studies on deep learning models for seeing prediction, including experimenting with deeper architectures, exploring larger hyperparameter spaces, etc. Furthermore, we only employed vanilla deep learning models in this work, which will inevitably lead to suboptimal results. Thus another line of future work is to design more elaborate deep learning architectures that can address defining challenges posed by the seeing prediction task.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (U1931207, 61602278 and 61702306), Sci. & Tech. Development Fund of Shandong Province of China (2016ZDJS02A11, ZR2017BF015 and ZR2017MF027), the Humanities and Social Science Research Project of the Ministry of Education (18YJAZH017), the Taishan Scholar Program of Shandong Province, and the Science and Technology Support Plan of Youth Innovation Team of Shandong Higher School (2019KJN024).

References

Benkhaldoun, Z., Abahamid, A., El Azhari, Y., & Lazrek, M. 2005, A&A, 441, 839
Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. 2015, Time Series Analysis: Forecasting and Control (New York: Wiley)
Bradley, E. S., Roberts, L. C., Jr, Bradford, L. W., et al. 2006, PASP, 118, 172
Brunner, F. 1982, BGeod, 56, 341
Chen, T., & Guestrin, C. 2016, in Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 785
Cherubini, T., Lyman, R., & Businger, S. 2021, MNRAS, 509, 232
Cho, K., van Merriënboer, B., Gulcehre, C., et al. 2014, in Proc. 2014 Conf. on Empirical Methods in Natural Language Processing, 1724
Chromey, F. R. 2016, To Measure the Sky: An Introduction to Observational Astronomy (Cambridge: Cambridge Univ. Press)
Coulman, C. 1985, ARA&A, 23, 19
Coulman, C., Andre, J.-C., Lacarrere, P., & Gillingham, P. 1986, PASP, 98, 376
Cui, X.-Q., Zhao, Y.-H., Chu, Y.-Q., et al. 2012, RAA, 12, 1197
Dueben, P. D., & Bauer, P. 2018, GMD, 11, 3999
Franke, B., Plante, J.-F., Roscher, R., et al. 2016, International Statistical Review, 84, 371
García-Lorenzo, B., Eff-Darwich, A., Fuensalida, J., & Castro-Almazán, J. 2009, MNRAS, 397, 1633
Giordano, C., Rafalimanana, A., Ziad, A., et al. 2021, MNRAS, 504, 1927
Good, J. M., Kelton, P. W., Booth, J. A., & Barker, E. S. 2003, Proc. SPIE, 4837, 237
Hochreiter, S., & Schmidhuber, J. 1997, Neural Comput., 9, 1735
Kornilov, M. V. 2016, ExA, 41, 223
Liu, L.-Y., Yao, Y.-Q., Wang, Y.-P., et al. 2010, RAA, 10, 1061
López-Rubio, E., & Ratti, E. 2021, Synthese, 198, 3131
Lyman, R., Cherubini, T., & Businger, S. 2020, MNRAS, 496, 4734
Milli, J., Gonzalez, R., Fluxa, P., et al. 2019, arXiv:1910.13767
Miyashita, A., Ogasawara, R., Macaraya, G., & Itoh, N. 2003, PNAOJ, 7, 25
Qian, X., Yao, Y., Wang, H., et al. 2018, PASP, 130, 125002
Roddier, F. 1981, PrOpt, 19, 281

Roddier, F. J., Cowie, L. L., Graves, J. E., et al. 1990, Proc. SPIE, 1236, 485
Sarazin, M. S. 1997, Proc. SPIE, 3125, 366
Singh, D., & Singh, B. 2020, Appl. Soft Comput., 97, 105524
Taylor, S. J., & Letham, B. 2018, The American Statistician, 72, 37
Tokovinin, A. 2002, PASP, 114, 1156
Tsapras, Y., Street, R., Horne, K., et al. 2009, AN, 330, 4
Vaswani, A., Shazeer, N., Parmar, N., et al. 2017, Advances in Neural Information Processing Systems, Vol. 5998 (Long Beach, CA: MIT Press)
Verleysen, M., & François, D. 2005, in Int. Work Conf. on Artificial Neural Networks (Berlin: Springer), 758
Vernin, J., & Munoz-Tunon, C. 1995, PASP, 107, 265
Xin, Y.-X., Bai, J.-M., Lun, B.-L., et al. 2020, RAA, 20, 149
Zhang, J.-C., Ge, L., Lu, X.-M., et al. 2015, PASP, 127, 1292