A Modified RNN-Based Deep Learning Method for Prediction of Atmospheric Visibility

Zengliang Zang 1, Xulun Bao 2*, Yi Li 1,Youming Qu 3, Dan Niu 4,* Ning Liu 1 and Xisong Chen 4

1 College of Meteorology and Oceanography, National University of Defense Technology, Changsha 410073, China
2 School of Software, Southeast University, Suzhou 215123, China
3 Hunan Meteorological Information Center, Changsha 410000, China
4 School of Automation, Southeast University, Nanjing 210096, China
* Correspondence: 101011786@seu.edu.cn

Abstract: Accurate atmospheric visibility prediction is of great significance to public transport safety. However, since it is affected by multiple factors, there still remains difficulties in predicting its heterogenous spatial distribution and rapid temporal variation. In this paper, a recursive neural network (RNN) prediction model modified with the frame-hopping transmission gate (FHTG), feature fusion module (FFM) and reverse scheduled sampling (RSS), named SwiftRNN, is developed. The new FHTG is used to accelerate training, the FFM is used for extraction and fusion of global and local features, and the RSS is employed to learn spatial details and improve prediction accuracy. Based on the ground-based monitoring data of atmospheric visibility from the China Meteorological Information Center during 1 January 2018 to 31 December 2020, the SwiftRNN model and two traditional ConvLSTM and PredRNN models are performed to predict hourly atmospheric visibility in central and eastern China at a lead of 12 h. The results show that the SwiftRNN model has better performance in the skill scores of visibility prediction than those of the ConvLSTM and PredRNN model. The averaged structural similarity (SSIM) of predictions at a lead up to 12 h is 0.444, 0.425 and 0.399 for the SwiftRNN, PredRNN and ConvLSTM model, respectively, and the averaged image perception similarity (LPIPS) is 0.289, 0.315 and 0.328, respectively. The averaged critical success index (CSI) of predictions over 1000 m fog area is 0.221, 0.205 and 0.194, respectively. Moreover, the training speed of the SwiftRNN model is 14.3% faster than the PredRNN model. It is also found that the prediction effect of the SwiftRNN model over 1000 m medium grade fog area is significantly improved along with lead times compared with the ConvLSTM and PredRNN model. All above results demonstrate the SwiftRNN model is a powerful tool in predicting atmospheric visibility.

Keywords: atmospheric visibility prediction; spatiotemporal sequence prediction; RNN; improve accuracy; feature fusion

1. Introduction

Atmospheric visibility is one of the important indicators of atmospheric transparency; it is of great importance in influencing weather and climate, such as the effects on precipitation and radiation mentioned by Yang et al. [1,2]. Snow, rain, fog, and haze could usually reduce the visibility. Low visibility may affect the safety of aviation, navigation, and transportation, and cause significant economic losses. The atmospheric visibility prediction is of great significance to support government decision-making, maintain production and life order and protect citizens’ health and safety. However, the accurate visibility prediction even for short lead times of less than 12 h is challenging [3–21]. The visibility is not only affected by temperature, humidity, and wind, but also closely related to aerosol concentration and chemical composition. Atmospheric visibility is one of the most difficult elements of meteorological prediction.
Since most numerical weather prediction (NWP) models do not directly predict visibility, atmospheric visibility is usually derived from the other NWP model variables, such as relative humidity, dew point, cloud water content, wind, and precipitation. Model Out Statistics (MOS) is a popular approach to derive atmospheric visibility by estimating a multivariate linear regression with NWP model variables [22,23]. However, the traditional MOS approach has several limitations for the atmospheric visibility prediction. First, the linear regression is usually unsuccessful for the small probability events, such as a sudden low visibility. Second, the effect of MOS regression is based on the persistent and stable NWP model forecast data. Once the NWP model changes, the effect of MOS approach will decrease. Third, in the 0-6 h after the NWP model is started, the cloud water content and rainwater content in the NWP model are usually very small due to the spin-up problem, which seriously affects the short-term forecast of visibility by MOS approach [24,25].

With the increase in computing power, machine learning algorithms have generated a great deal of interest in short-term weather prediction research. Sara et al. [26] discussed short-term prediction of hourly low-visibility events using different methods, some of which come from persistence analysis via Markov chain models and others based on machine learning (ML) techniques. The results showed that a hybrid expert approach involving persistence-based methods and machine learning techniques provides the best results on this prediction problem. Driss et al. [27] investigated the use of supervised machine learning regression techniques (tree-based ensembles, feedforward neural networks and generalized linear methods) to diagnose the visibility prediction effect of a large-area mesoscale model by hourly observations from 36 synoptic land stations in the northern region of Morocco. The tree-based ensemble approach showed significant improvements for the accuracy of visibility prediction compared to a practical visibility diagnostic scheme based on the Kunkel formula. A statistical model with a physical basis was constructed to predict visibility in the Arctic based on a dynamic Bayesian network, and test-ed daily visibility prediction over a 1° × 1° grid area. The results show that the mean relative error of the predicted visibility from the dynamic Bayesian network is approximately 14.6% compared with the inferred visibility from the artificial neural network [28]. It is concluded that the dynamic Bayesian network is useful for predicting visibility in the Arctic. In addition, the previous studies using machine learning or deep learning methods are mostly based on one-dimensional sequences and rarely extract spatial features. In recent years, two-dimensional time-series prediction models have been used for precipitation or weather radar images prediction with good results. These models can be regarded as a spatiotemporal sequence prediction problem and often adopt the sequence of past meteorological images as input to output the sequence of future meteorological images. In this paper, such models considering spatiotemporal correlation are migrated and optimized for atmospheric visibility prediction models.

Recently, recursive neural network (RNN), including long-term and short-term memory (LSTM) based on encoding and decoding framework [29–31], are used in weather forecasting to capture sequence temporal correlation. Klein et al. [32] proposed a dynamic convolution method to extract spatiotemporal features to predict image sequence of rain and snow. Shi et al. [33] developed a convolution LSTM (ConvLSTM) model based on the traditional LSTM model. It could capture the dynamic features between image sequences. In addition, they further proposed the trajectory gated recursive unit (Trajgru) model [34], which is more flexible than the ConvLSTM model through the recursive connection structure of location variables and performs better in the task of shortening precipitation prediction. Wang et al. [35] proposed a predictive recurrent neural network (PredRNN) and used a unified memory pool to remember spatial appearance and temporal changes. The results revealed the PredRNN model obtained better performance than the previous Trajgru and ConvLSTM models. In addition, the hybrid method combining the effective information of observation data and NWP variables can further improve the prediction accuracy. A model based on the UNet [36] is proposed to improve the accuracy of high rainfall prediction by the fusion of rainfall radar image and wind speed generated by
the NWP model. A dual encoder network structure is also proposed to extract NWP-based and observation-based prediction features [37,38].

Though the deep learning algorithms were widely utilized for weather forecasting, there are some problems of slow prediction speed and fuzzy prediction image [39–41]. In this paper, a modified SwiftRNN model based on the PredRNN model is proposed to improve the forecast accuracy and preserve the spatial details of the forecasted atmospheric visibility images. We design a new frame-hopping transmission gate (FHTG) unit to obtain faster training speed, use feature fusion module (FFM) to extraction and fusion global and local features and use reverse scheduled sampling (RSS) to obtain higher accuracy and more spatial details in the SwiftRNN model. In the following sections, we use the atmospheric visibility data set in central and eastern China to evaluate the SwiftRNN model and compare the performance with the traditional ConvLSTM and PredRNN model.

2. Method and Data

The atmospheric visibility prediction method needs to predict the future visibility image sequences from the previous visibility image sequences. In this section, we first introduce the basic formulation of prediction problem, then the shortcomings of the traditional PredRNN model are analyzed and the structure of the proposed SwiftRNN model are presented. Finally, the implementation of two models is also described.

2.1. Formulation of Prediction Problem

The prediction of atmospheric visibility can be seen as a spatiotemporal sequence prediction problem solved by deep learning methods. The basic formulation is to input a period of historical consecutive images into a deep learning system and to output the next period of consecutive images. In this paper, the input and output images are two-dimensional atmospheric visibility field color speckle maps (RGB image). The inferred sequence of the color spot map is a prediction as to the future atmospheric visibility. The observed image at time \( t \) can be expressed as a tensor \( X_t \in \mathbb{R}^{(P \times M \times N)} \), where \( \mathbb{R} \) represents the observed feature domain, \( P, M \) and \( N \) represent the RGB channels, width and height of atmospheric visibility maps, respectively. From the perspective of time, if the observations are recorded periodically, it will generate a tensor sequence. Therefore, the deep learning prediction problem of spatiotemporal sequence is to predict the \( K \) sequences in the future by giving the previous \( T \) sequences (including the current observation value), as shown in Equation (1).

\[
\hat{X}_{T+1}, \ldots, \hat{X}_{T+K} = \arg\max_{X_{T+1}, \ldots, X_{T+K}} \rho(X_{T+1}, \ldots, X_{T+K} | X_t, \ldots, X_T)
\]  

(1)

In this paper, \( T \) and \( K \) are all set as 12. That is to say, we use the previous 12 h historical visibility field color spot map (12 consecutive images) to predict the future 12 h visibility field color spot map.

2.2. Network Structure

2.2.1. Challenges in the PredRNN Model

The PredRNN model is a kind of RNN neural network proposed based on the ConvLSTM model; it replaces the ConvLSTM model with a new spatiotemporal short-term memory (ST-LSTM) unit. This spatiotemporal storage unit is updated in a zigzag direction. The information is first transmitted upward across layers, and then forward with time to realize the efficient flow of spatial information. However, the gradient is easy to disappear in the transmitted process because the memory needs to flow a longer path between distant states. With the help of the ST-LSTM unit, the PredRNN model realizes the simultaneous extraction of standard temporal memory and recommended spatiotemporal memory. The PredRNN model structure is shown in Figure 1.
Figure 1. The architecture of the ST-LSTM unit (left) and the PredRNN model (right). The $X_t$, $H_t$, $M^l_t$ and $C^l_t$ in the left panel are the data image, the hidden state, the spatiotemporal memory cell and the standard temporal memory cell, respectively. $i$, $f$, $g$ and $o$ indicate the input gate, the forget gate, the input-modulation gate and the output gate. The orange arrows in the right panel denote the spatiotemporal memory flow, namely the transition path of the spatiotemporal memory cell $M^l_t$ in the left panel.

The equations of the ST-LSTM unit are shown as follows:

$$
\begin{align*}
g_t &= \tanh \left( W_{xg} \ast X_t + W_{bg} \ast H_{t-1}^l + b_g \right) \\
i_t &= \sigma \left( W_{xi} \ast X_t + W_{hi} \ast H_{t-1}^l + b_i \right) \\
f_t &= \sigma \left( W_{xf} \ast X_t + W_{hf} \ast H_{t-1}^l + b_f \right) \\
C_t^l &= f_t \circ C_{t-1}^l + i_t \circ g_t \\
g'_t &= \tanh \left( W_{xg} \ast X_t + W_{mg} \ast M_{t-1}^l + b_g' \right) \\
i'_t &= \sigma \left( W'_{xi} \ast X_t + W'_{mi} \ast M_{t-1}^l + b_i' \right) \\
f'_t &= \sigma \left( W'_{xf} \ast X_t + W'_{mf} \ast M_{t-1}^l + b_f' \right) \\
M_t^l &= f'_t \circ M_{t-1}^l + i'_t \circ g'_t \\
o_t &= \sigma \left( W_{xo} \ast X_t + W_{ho} \ast H_{t-1}^l + W_{co} \ast C_t^l + W_{mo} \ast M_t^l + b_o \right) \\
H_t^l &= o_t \circ \tanh \left( W_{l,x1} \ast \left[ C_{t}^l, M_t^l \right] \right)
\end{align*}
$$

(2)

In Equation (2), $W$ and $b$ are the learnable weight and bias parameters which should be optimized in the training process. The operator $\ast$ and $\circ$ donate the convolution operator and the Hadamard product. The $\tanh$ and $\sigma$ represent tanh and sigmoid activation function, respectively. Maintain two memory cells, e.g., the standard temporal memory cell $C^l_t$ and the spatiotemporal memory cell $M^l_t$. $C^l_t$ horizontally conveys the temporal evolution at the current time step $t$ from the previous time step $t-1$. $M^l_t$ vertically transits the spatiotemporal memory in the current layer $l$ from the previous layer $l-1$. These storage cells from different directions are connected, and then the convolution layer is applied to extract spatial information across different time steps. The final hidden state $H_t^l$ depends on the fused spatiotemporal memory cell $M^l_t$ and standard temporal cell $C^l_t$, so that the hidden state is the same as the dimension of the storage cells. Different from the simple LSTM unit, the ST-LSTM unit can effectively extract the shape deformation and motion trajectory in spatiotemporal image sequences. However, there are three challenges in the PredRNN model. Firstly, the ST-LSTM unit is often difficult in transmitting gradient during
back propagation process. Thus, the cyclic architecture of the ST-LSTM unit is unstable, especially for continuous images with periodic motion. Secondly, the PredRNN model follows a sequence-to-sequence architecture. In the training phase, it always takes the real images as the input of the encoding time step, which may affect the long-term dynamic learning process. Finally, the PredRNN does not capture the global and local features sufficiently, which leads to less than optimal final predictions of the model.

2.2.2. The SwiftRNN Model

To solve the above three challenges, we propose a new SwiftRNN model based on the PredRNN model. For the first challenge, we design a frame-hopping transmission gate (FHTG) to improve the difficulty of gradient transmission. In the new structure, the FHTG unit works seamlessly with the ST-LSTM unit to capture the dependence of long-term and short-term continuous images, respectively. By quickly updating the hidden state, the FHTG unit could carry out a fast route from any vertical steps of the first-time step to any vertical steps of the last time step (green line in Figure 2). It could realize the adaptive learning of the long-term and short-term image relationship.

![Figure 2. The architecture of the FHTG unit (left), FFM unit (left) and the SwiftRNN model (right). The orange arrows in SwiftRNN model denote the spatiotemporal memory flow.](image)

The internal calculation formula of the SwiftRNN model integrating the FHTG unit is as follows.

\[
Z_{l-1}^t = FHTG(H_{l-1}^t, Z_{l-1}^t)
\]

\[
H_{l}^t, C_{l}^t, M_{l}^t = STLSTM_{l}(Z_{l-1}^t, H_{l-1}^t, C_{l-1}^t, M_{l-1}^t)
\]

The internal calculation formula of the FHTG unit is:

\[
P_l = \tanh(W_{px} * X_t + W_{pz} * Z_{l-1})
\]

\[
S_l = \sigma(W_{px} * X_t + W_{pz} * Z_{l-1})
\]

\[
Z_l = S_l \circ P_l + (1 - S_l) \circ Z_{l-1}
\]

For the second challenge, inspired by the PredRNN-v2 model [42], we employ the new curriculum learning strategy of the PredRNN-v2 model, which consists of two parts, e.g., reverse scheduled sampling (RSS) and scheduled sampling (SS). Figure 3 shows two
possible strategies for combining RSS and SS strategies. The RSS strategy is employed at encoding timesteps to force the PredRNN model to learn long-term dynamics. The SS strategy is utilized to alleviate the inconsistency of data flow at forecasting timesteps between the training and inference process. The formula of encoder and forecaster is as follows:

\[
\hat{X}_{t+1} = \begin{cases} 
\text{SwiftRNN}(\hat{X}_t \xrightarrow{RSS} X_t, H_{t-1}, Q_{t-1}) & \text{if } t \leq T, \\
\text{SwiftRNN}(X_t \xrightarrow{SS} \hat{X}_t, H_{t-1}, Q_{t-1}) & \text{if } t > T,
\end{cases}
\]

where \(Q_{t-1} = \{C_{t-1}, M_{t-1}\}\) is a combination of memory cells at previous time step \(t - 1\). The main difference between encoding part \((t \leq T)\) and forecasting part \((t > T)\) is whether the observed image \(X_t\) or the previous prediction image \(\hat{X}_t\) is used. In the encoder, the model mainly learns observed images because \(X_t\) contains more accurate information than \(H_{t-1}\) and \(Q_{t-1}\). However, for the forecaster, due to the fact that there are no new observations in the future timestep, the model should learn long-term dynamics transmitted from \(H_{t-1}\) and \(Q_{t-1}\). Specifically, in encoder part, RSS gradually uses the observed image \(X_t\) instead of the previous prediction image \(\hat{X}_t\) with an increasing probability \(p_{RSS}\) during the training process. Conversely, for SS in the forecaster part, the previous prediction image \(\hat{X}_t\) will be used with an increasing probability \(p_{SS}\) instead of the observed image \(X_t\) during the training process. The increasing probability function \(p_{RSS}\) and \(p_{SS}\) used in this study are displayed in Equation (6). \(p_{RSS}\) is an increasing function of the number of training iterations \(k\), starting from \(\epsilon_s\) and increasing to \(\epsilon_c\), where \(\alpha_t, \beta_s > 0\) denotes the increasing factors and \(\beta_s > 0\) denotes the starting point of the sigmoid function.

\[
p_{RSS} = \epsilon_c + (\epsilon_c - \epsilon_s) \times \frac{1}{1 + \exp(\frac{\epsilon_c - \epsilon_s}{\beta_s})} \\
p_{SS} = \min(\epsilon_s + \alpha_t \times k, \epsilon_c)
\]

Figure 3. The RSS strategy in encoder and the SS strategy in forecaster. \(X_t\) is the observed image at timestep \(t\) and \(\hat{X}_{t+1}\) is the predicted image at the next timestep \(t + 1\).

If the conventional training method is used, the inconsistent training between encoder and forecaster may lead to invalid optimization and hinder the long-term dynamics of model learning. Additionally, the RSS training strategy will effectively relieve the above problem.

For the final challenge, we intend to improve the sensitivity of the model to low atmospheric visibility by using global features. We use a transformer-based model to extract global information. Therefore, we propose a novel feature fusion module (FFM) that includes a local feature extraction unit, a global feature extraction unit, and then we fuse these two different scales of features. When predicting atmospheric visibility, in addition to global features, local features are also important. Therefore, we designed a
local feature extraction unit, which can fully extract local features from the atmospheric visibility information. It is helpful for prediction of high and medium atmospheric visibility. Concretely, our local features extraction unit mainly consists of convolution block and attention mechanisms. The latter is composed of a convolutional block attention module (CBAM) [43].

The last step of this module is to fuse the extracted two parts of features as shown in Equation (7). It is similar to a gating mechanism. It empties what needs to be forgotten and keeps what needs to be remembered. By constraining the local and global features through the above gating mechanism, it greatly alleviates the numerical differences and avoids the problem that linear fusion does not work due to significant differences between them.

\[
X'_t = \sigma(global) \ast global + \sigma(local) \ast local
\]  

where local and global represent local features and global features, respectively.

2.3. Loss Function

The frequency of atmospheric visibility data with different levels is significantly unbalanced, so the weighted loss is used to alleviate this problem. Based on the principle that the lower visibility shares the greater importance, we define different weights for atmospheric visibility data with different levels by Equation (8):

\[
\omega(X) = \begin{cases} 
25, & X < 1000 \text{ m} \\
9, & 1000 \text{ m} \leq X < 4000 \text{ m} \\
4, & 4000 \text{ m} \leq X < 10000 \text{ m} \\
1, & 10000 \text{ m} \leq X 
\end{cases}
\]

where, \( X \) represents the atmospheric visibility image in this study. The weighted loss function we designed is shown in Equation (9):

\[
loss = \frac{1}{K} \sum_{t=1}^{K} MSE(X_t, \hat{X}_t) + \frac{1}{2}(1 - SSIM(X_t, \hat{X}_t) + LPIPS(X_t, \hat{X}_t))
\]

\[
MSE(X_t, \hat{X}_t) = \sum_{i,j} \omega_{t,i,j}(X_{t,i,j} - \hat{X}_{t,i,j})^2
\]

where \( K \) represents the number of all images in the predicted sequence, and \( \omega_{t,i,j} \) represents the weight of the \((i,j)\)th pixel at the \(t\)-th image. \( X_{t,i,j} \) and \( \hat{X}_{t,i,j} \) are the value of \((i,j)\)th of ground observed visibility field map \( X_t \) and predicted visibility field map \( \hat{X}_t \) at the time step \( t \). In the calculation of mean square error (MSE), more weight is given to the lower atmospheric visibility to improve the prediction performance of low visibility. By this way, the lack of low visibility samples can be compensated to some extent. The main purpose of training this network is to ensure that the predicted visibility field map is as close to the observed image as possible. The combination of structural similarity (SSIM) [44] and image perception similarity index (LPIPS) [45] in the loss function can enhance the details of the predicted image.

2.4. Implementation

All models are optimized using the Adam optimizer with a learning rate equal to \(10^{-4}\). The parameters in Adam optimizer are set \(\beta_1 = 0.9, \beta_2 = 0.999\). The formula for the Adam optimizer is shown in Equation (10), where \(m_t\) represents past gradients, \(v_t\) represents square of past gradient, \(m_t\) and \(d_t\) are the deviation correction of \(m_t\) and \(v_t\), \(\theta_{t+1}\) represents the new gradient. The implementation details of the SwiftRNN model are shown in Table 1. The training batch size in the SwiftRNN model is set to 4. The kernel sizes of all convolutional layers except that in the output operations (the kernel size is \(1 \times 1\)) are set as \(5 \times 5\). Before the convolution operation, the boundary of the visibility field map is zero-filled to keep the size of the feature map unchanged. We implemented the SwiftRNN
model using the Pytorch framework and trained this model with NVIDIA RTX 3090 GPU and CUDA 11.0.

\[
\begin{align*}
m_t &= \beta_1 m_{t-1} + (1 - \beta_1) \hat{g}_t \\
v_t &= \beta_2 v_{t-1} + (1 - \beta_2) \hat{g}_t^2 \\
m_t &= \frac{m_t}{1 - \beta_1} \\
v_t &= \frac{v_t}{1 - \beta_2} \\
\theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{v_t + \epsilon}} m_t
\end{align*}
\]

(10)

Table 1. Implementation details of the proposed method.

<table>
<thead>
<tr>
<th>Hyper-Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST-LSTM layers</td>
<td>4</td>
</tr>
<tr>
<td>Kernel sizes of convolutional layers</td>
<td>5 × 5</td>
</tr>
<tr>
<td>Channels of convolutional layers</td>
<td>128</td>
</tr>
<tr>
<td>Optimizer</td>
<td>Adam ((\beta_1 = 0.9, \beta_2 = 0.999))</td>
</tr>
<tr>
<td>Batch size</td>
<td>4</td>
</tr>
<tr>
<td>Learning rate</td>
<td>(1 \times 10^{-4})</td>
</tr>
<tr>
<td>Framework</td>
<td>Pytorch 1.7</td>
</tr>
<tr>
<td>GPU</td>
<td>NVIDIA RTX 3090</td>
</tr>
</tbody>
</table>

2.5. Visibility Field Map Dataset

2.5.1. Introduction to the Dataset

Using the deep learning method to realize atmospheric visibility prediction requires a large amount of data. In this paper, the atmospheric visibility data is obtained from the meteorological station observed by China Meteorological Information Center. We collected the hourly visibility data from 1st January 2018 to 31st December 2020. The regional geographical scope covers the central and eastern regions of China, with longitude ranging from 100° to 120° E and latitude ranging from 20° to 40° N. A total of 2428 observation stations are selected in the area. Figure 4 shows the geographical area involved in the data set.

![Figure 4. Geographic area involved in atmospheric visibility data set. The black line represents the provincial line, and the red points indicate the meteorological station from the China Meteorological Information Center.](image)

2.5.2. Pre-Process of the Visibility Field Map Dataset

Because the original observation data has outliers and defect values, the damaged data should be cleaned before training the model, including outliers’ deletion and missing data filling. For the data of outliers, e.g., 999,999, are directly deleted. Additionally, then two methods are used to fill missing data:

1. If the number of continuous missing data does not exceed 1, the previous and behind two non-missing data are used for mean filling.
2. If the number of continuous missing data exceeds 1, the previous and behind non-missing data are used for linear interpolation for filling.

The data set required for spatiotemporal sequence prediction is a set of continuous two-dimensional images. Therefore, this paper makes a user-defined color mapping on the longitude and latitude of the hourly site data, and finally generates the image in RGB picture. Because the number of ground observation stations is limited and cannot cover all grid points, linear interpolation is used to fill the values of grid points without station. We have used the bilinear interpolation method, which is a linear interpolation extension of the interpolation function with two variables, the core idea of which is to perform one linear interpolation in each of the two directions.

When sorting the pictures, it is found that some pictures are all-blue or all-green (indicating high visibility), but the pictures before and after them are not all-blue or all-green and have obvious low visibility. Therefore, it is speculated that these all-blue or all-green pictures are caused by the lack of collected station data. Therefore, delete these all-blue or all-green pictures, and then copy the adjacent pictures to the default place. This can avoid a negative impact on the training process. Considering the time and memory cost of deep learning training, compress the pictures to 128 × 128 pixels and 96 DPI.

In the time segment extraction function, we select the time segments with continuous time, and take corresponding pictures during the time segments as modelling samples. If there are any missing pictures in the time segment, this segment will not be considered. Among them, the modelling samples in 2018 and 2019 are used as training sets and validation sets, with a ratio of 8:2. Modelling samples in 2020 is used as a test set. Details of the visibility field map dataset is shown in Table 2.

Table 2. Details of the visibility field map dataset.

<table>
<thead>
<tr>
<th>Year</th>
<th>Number of Pictures</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018</td>
<td>8760</td>
</tr>
<tr>
<td>2019</td>
<td>8760</td>
</tr>
<tr>
<td>2020</td>
<td>8784</td>
</tr>
</tbody>
</table>

3. Experiments

3.1. Evaluation

In this paper, we use a variety of metrics to quantify the prediction effect of three models, including SSIM (structural similarity), LPIPS (image perception similarity index) and CSI (critical success index). The SSIM and LPIPS metrics are used to quantify the image similarity based on the perspective of iconology, and the CSI metric is commonly used in the field of meteorology, which represents the success rate of prediction.

SSIM is an index to measure the similarity of two images. Given two images \(x\) and \(y\), the structural similarity of the two images can be calculated according to Equation (11), where \(\mu_x\) is the average of \(x\), \(\mu_y\) is the average of \(y\), \(\sigma^2_x\) is the variance of \(x\), \(\sigma^2_y\) is the variance of \(y\), and \(\sigma_{xy}\) is the covariance of \(x\) and \(y\). \(c_1 = (k_1L)^2\), \(c_2 = (k_2L)^2\) is a constant used to maintain stability, \(L\) is the dynamic range of pixel value, \(k_1 = 0.01\), \(k_2 = 0.03\).

\[
SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu^2_x + \mu^2_y + c_1)(\sigma^2_x + \sigma^2_y + c_2)}
\]  

(11)

LPIPS learns to generate the reverse mapping from the image to the ground truth, forces the generator to learn to reconstruct the reverse mapping of the real image from the false image, and gives priority to processing the perceptual similarity between them, which is more in line with human perception. The lower the value of LPIPS, the more similar the two images are, where \(d(x, x_0)\) represents the distance of \(x\) and \(x_0\) (\(d\) represents LPIPS), as shown in Equation (12), where \(H_l\) and \(W_l\) represent the height and width of the image, respectively in convolutional layer \(l\), \(\hat{g}^l_{raw}\) and \(\hat{g}^l_{0raw}\) are the result of normalizing the output
of each convolutional layer after activation, vector $w_l$ is used to deflate the number of active channels, and $\circ$ represents the multiplication of $w_l$ and $(\hat{y}_{hw}^l - \hat{y}_{0hw}^l)$.

$$d(x, x_0) = \sum_l \frac{1}{H_lW_l} \sum_{h,w} \| w_l \circ (\hat{y}_{hw}^l - \hat{y}_{0hw}^l) \|^2_2$$

(12)

For the construction of the CSI metric in this study, since the prediction task is completed at the pixel level, they are projected back to the atmospheric visibility, and the atmospheric visibility of each pixel is calculated. This metric is like the classification metric and mainly focuses on whether the prediction of the location point is hit within a certain threshold range. For example, if the threshold is 1000 m, after binarization, 999 m will be converted to 0, 1001 m will be converted to 1. After converting each pixel value between the predicted value and the real value to 0/1, the hits (true positive, predicted value = 1, real value = 1), misses (false positive, predicted value = 1, real value = 0) and false alarms (false negative, predicted value = 0, true value = 0) are calculated. To fully evaluate the performance of the algorithm, this paper calculates this metric under three thresholds, e.g., 1000 m, 4000 m and 10,000 m, corresponding to atmospheric visibility with different levels. The CSI metric is shown in Equation (13).

$$CSI = \frac{\text{hits}}{\text{hits} + \text{misses} + \text{false alarms}}$$

(13)

3.2. Results

In this section, we compare the SwiftRNN model with the classical ConvLSTM and PredRNN models based on the SSIM, LPIPS and CSI metrics.

To intuitively analyze the prediction effect, a group of individual cases are selected from the test set for analysis. Figure 5 shows the spatial distribution of atmospheric visibility images predicted by the ConvLSTM, PredRNN and SwiftRNN models at 12:00–17:00 on 31st October 2020; those results are also compared with the observed atmospheric visibility images. It can be seen that the spatial distribution of atmospheric visibility images predicted by the ConvLSTM, PredRNN and SwiftRNN models is in good agreement with the observed images. However, since the predictions of the ConvLSTM model are relatively unsatisfactory compared with the other two models, we only analyze the results of the proposed SwiftRNN model and the PredRNN model. Generally, the SwiftRNN and PredRNN model can well predict two large low-visibility areas in north and central China. Additionally, in some fine local characteristics, the low visibility in Wuhan, Tianjin, Shijiazhuang and other areas can also be well grasped. However, the overall intensity of the predicted images is slightly larger than the observed images, and the low visibility in Henan is underestimated in the next few hours. Compared with the PredRNN model, the visibility distribution predicted by the SwiftRNN model is closer to the observed images. For the example in Figure 5, the PredRNN model forecasts lower visibility in Shanxi, Shaanxi and the Yangtze River Delta, while the predicted results by the SwiftRNN model are more accurate and do not differ much from the observed images. In addition, the advantage of the SwiftRNN model is more obvious in the last 6 h. Figure 6 shows the comparison of the predicted results from 18:00 to 23:00 on 31st October 2020, the predicted results from the SwiftRNN model have a larger area of low visibility in central China and Guizhou, and a smaller area of low visibility in northern Jiangsu, northern Anhui, and Henan, which is closer to the observed images. However, the PredRNN model predicts heavier visibility over more areas, and the spatial details of the predictions are not as good as those of the SwiftRNN model. Overall, the PredRNN model is less accurate compared with the SwiftRNN model.
heavier visibility over more areas, and the spatial details of the predictions are not as good as those of the SwiftRNN model. Overall, the PredRNN model is less accurate compared with the SwiftRNN model.

Figure 5. Observed and predicted images on 31st October 2020 (12:00–17:00). (a) Observed; (b) ConvLSTM; (c) PredRNN; (d) SwiftRNN.

Figure 6. Observed and predicted images on 31st October 2020 (18:00–23:00). (a) Observed; (b) ConvLSTM; (c) PredRNN; (d) SwiftRNN.
Tables 3 and 4 show the SSIM metrics and LPIPS metrics of the 12 h predictions by the ConvLSTM, PredRNN and SwiftRNN model. It can be seen from these tables that the performance of the SwiftRNN model is better than the PredRNN model in four seasons. Thus, the proposed SwiftRNN model is conducive to predict clearer and more detailed atmospheric visibility images.

Table 3. The SSIM metric of visibility field map prediction for 4 months.

<table>
<thead>
<tr>
<th>MODEL</th>
<th>January</th>
<th>April</th>
<th>July</th>
<th>October</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvLSTM</td>
<td>0.381</td>
<td>0.398</td>
<td>0.431</td>
<td>0.389</td>
</tr>
<tr>
<td>PredRNN</td>
<td>0.394</td>
<td>0.431</td>
<td>0.461</td>
<td>0.415</td>
</tr>
<tr>
<td>SwiftRNN</td>
<td>0.415 (5.33%↑)</td>
<td>0.452 (4.87%↑)</td>
<td>0.472 (2.39%↑)</td>
<td>0.435 (4.82%↑)</td>
</tr>
</tbody>
</table>

**↑** means the improvement rate of the SSIM metric of SwiftRNN model in comparison to that of the PredRNN model.

Table 4. Same with Table 3, but for LPIPS metric.

<table>
<thead>
<tr>
<th>MODEL</th>
<th>January</th>
<th>April</th>
<th>July</th>
<th>October</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvLSTM</td>
<td>0.328</td>
<td>0.325</td>
<td>0.337</td>
<td>0.321</td>
</tr>
<tr>
<td>PredRNN</td>
<td>0.308</td>
<td>0.321</td>
<td>0.320</td>
<td>0.312</td>
</tr>
<tr>
<td>SwiftRNN</td>
<td>0.286 (7.14%↓)</td>
<td>0.278 (13.39%↓)</td>
<td>0.314 (1.86%↓)</td>
<td>0.277 (11.22%↓)</td>
</tr>
</tbody>
</table>

**↓** means the reduction rate of the LPIPS metric of SwiftRNN model in comparison to that of the PredRNN model.

Figure 7 compares image similarity metrics in four seasons of the SwiftRNN and PredRNN model; we can see in winter, spring and autumn, the SSIM metrics of the predicted atmospheric visibility images by the SwiftRNN model are lower than those by the PredRNN model in the first four hours, but the SSIM metrics by the SwiftRNN model are higher than those by the PredRNN model in the following hours, which indicates that the SwiftRNN model can capture the characteristics of a longer time. Compared the SSIM metric in summer, the SwiftRNN model performs better than the PredRNN model in all hours. In terms of the LPIPS metric, the LPIPS metrics of the SwiftRNN model are lower than those of the PredRNN model. Therefore, in general, the images predicted by the SwiftRNN model are more detailed and closer to the observed images than those predicted by the PredRNN model.

![Image similarity metrics](image.png)

Figure 7. Image similarity metrics for visibility field map prediction in (a) January, (b) April, (c) July, and (d) October.

Table 5–7 show the CSI metrics of the 12 h predictions by the ConvLSTM, PredRNN and SwiftRNN model. To fairly compare and fully evaluate the performance of the algorithm, we calculate the CSI metric under three atmospheric visibility thresholds in different seasons, including 1000 m, 4000 m and 10,000 m.
Table 5. CSI-1000 of visibility field map prediction for 4 months.

<table>
<thead>
<tr>
<th>MODEL</th>
<th>January</th>
<th>April</th>
<th>July</th>
<th>October</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvLSTM</td>
<td>0.163</td>
<td>0.221</td>
<td>0.206</td>
<td>0.189</td>
</tr>
<tr>
<td>PredRNN</td>
<td>0.179</td>
<td>0.229</td>
<td>0.215</td>
<td>0.197</td>
</tr>
<tr>
<td>SwiftRNN</td>
<td>0.193 (7.82%↑)</td>
<td>0.243 (6.11%↑)</td>
<td>0.232 (7.91%↑)</td>
<td>0.213 (8.12%↑)</td>
</tr>
</tbody>
</table>

↑↑ means the improvement rate of the CSI-1000 metric of SwiftRNN model to that of PredRNN model.

Table 6. Same with Table 5, but for CSI-4000 metric.

<table>
<thead>
<tr>
<th>MODEL</th>
<th>January</th>
<th>April</th>
<th>July</th>
<th>October</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvLSTM</td>
<td>0.417</td>
<td>0.465</td>
<td>0.501</td>
<td>0.458</td>
</tr>
<tr>
<td>PredRNN</td>
<td>0.433</td>
<td>0.483</td>
<td>0.520</td>
<td>0.477</td>
</tr>
<tr>
<td>SwiftRNN</td>
<td>0.460 (6.24%↑)</td>
<td>0.510 (5.29%↑)</td>
<td>0.552 (6.15%↑)</td>
<td>0.506 (6.08%↑)</td>
</tr>
</tbody>
</table>

↑↑ means the improvement rate of the CSI-4000 metric of SwiftRNN model to that of PredRNN model.

Table 7. Same with Table 5, but for CSI-10000 metric.

<table>
<thead>
<tr>
<th>MODEL</th>
<th>January</th>
<th>April</th>
<th>July</th>
<th>October</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvLSTM</td>
<td>0.506</td>
<td>0.555</td>
<td>0.608</td>
<td>0.558</td>
</tr>
<tr>
<td>PredRNN</td>
<td>0.527</td>
<td>0.577</td>
<td>0.632</td>
<td>0.579</td>
</tr>
<tr>
<td>SwiftRNN</td>
<td>0.553 (4.93%↑)</td>
<td>0.603 (4.51%↑)</td>
<td>0.664 (5.06%↑)</td>
<td>0.609 (5.18%↑)</td>
</tr>
</tbody>
</table>

↑↑ means the improvement rate of the CSI-10000 metric of SwiftRNN model to that of PredRNN model.

It can be seen that in the deep learning method, the nonlinear and convolutional structure of the network can learn some complex spatiotemporal patterns in the data set. Obviously, our proposed SwiftRNN model better achieves the CSI metric than the PredRNN model. At the threshold of 1000 m, the CSI metrics of the SwiftRNN model in winter, spring, summer, and autumn are increased by about 7.82%, 6.11%, 7.91% and 8.12%, respectively, compared with those of the PredRNN model. At the threshold of 4000 m, the CSI metrics of the SwiftRNN model in winter, spring, summer, and autumn are increased by about 6.24%, 5.59%, 6.15% and 6.08%, respectively compared with those of the PredRNN model. At the threshold of 10,000 m, the CSI metrics of the SwiftRNN model in winter, spring, summer, and autumn are increased by about 4.93%, 4.51%, 5.06% and 5.18%, respectively compared with those of the PredRNN model. It can be concluded that although the SwiftRNN model obtains the promising performance at the threshold of 4000 m and 10,000 m, there is a maximum percentage increase at the threshold of 1000 m, which means that the proposed model has better prediction performance for low visibility. In addition, the improvement effect of atmospheric visibility prediction in winter, summer and autumn is relatively similar, and the improvement effect of atmospheric visibility prediction in spring is not obvious.

Figure 8 shows the CSI metrics of three models in four seasons. Comparing (a) January, (b) April, (c) July, and (d) October horizontally, it can be seen that the two models obtain the best predicted effect in summer and the worst predicted effect in winter. Vertical comparison of (a) January, (b) April, (c) July, and (d) October shows that the predicted effect of 10,000 m threshold is the best, the predicted effect of a 4000 m threshold is the second, and the predicted effect of 1000 m is the worst. Additionally, the SwiftRNN model performs better than the PredRNN model in all three thresholds.

Table 8 indicates the training speed of three models. Based on 40,000 epochs, where the PredRNN model is about 6.132 s per epoch and the SwiftRNN model is about 5.255 s per epoch, the improvement effect is 14.3%.
Figure 8. Prediction evaluation metric for visibility field map prediction in (a) January, (b) April, (c) July, and (d) October.

Table 8. Model training speed.

<table>
<thead>
<tr>
<th>MODEL</th>
<th>Seconds/Epoch (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvLSTM</td>
<td>6.774</td>
</tr>
<tr>
<td>PredRNN</td>
<td>6.132</td>
</tr>
<tr>
<td>SwiftRNN</td>
<td>5.255 (14.3%)</td>
</tr>
</tbody>
</table>

↓ means the reduction rate of the seconds/epoch metric of SwiftRNN model to that of PredRNN model.

4. Conclusions

In this paper, a SwiftRNN model for atmospheric visibility prediction based on the PredRNN model is proposed, and the atmospheric visibility in central and eastern China is predicted based on the station observation data of China Meteorological data network. This proposed model adds a multi-layer FHPG unit in the PredRNN model and uses the RSS strategy in the training process. This paper realizes the simultaneous spatiotemporal prediction of atmospheric visibility and improves the spatial details and accuracy of the predicted atmospheric visibility images. Compared with the PredRNN model, the SwiftRNN model has a more obvious improvement effect on the prediction effect of medium and low visibility along with the lead times. The training speed of the SwiftRNN model is also saved.

At present, the application of deep learning methods in the field of meteorology is still in the exploratory stage, and there is still much room for improvement. For example, the images predicted by the PredRNN and SwiftRNN model are relatively blurred, which will have a certain negative impact on the final test score. How to make the model produce finer prediction images as much as possible without losing the model test score is the focus of the next research. In addition, the accuracy deterioration of prediction effect along with the lead times is still an unsolved problem. In future research, the Attention Augmented TransUNet (AA-TransUNet) [46] model and multi-attention mechanism can be used to output finer prediction results and improve the prediction effect over time. In terms of atmospheric visibility prediction, how to effectively fuse radar, ground, satellite, and other multi-source data for prediction, and constantly incorporate the newly generated observation data into the neural network model training, to continuously improve the prediction effect with the passage of time still need more attention of researchers. In addition, how to organically combine the physical conceptual model into the deep learning algorithm is also worthy of in-depth research.
Author Contributions: Conceptualization, X.B.; Data curation, Z.Z.; Funding acquisition, Z.Z.; Methodology, X.B. and D.N.; Project administration, Z.Z. and D.N.; Resources, Y.L. and Y.Q.; Software, X.B. and D.N.; Validation, X.C.; Visualization, X.B.; Writing—original draft, X.B.; Writing—review and editing, Z.Z., D.N., N.L. and X.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China, Grant Nos. 41975167.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank Yichao Cao for helpful discussions and suggestions.

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

References


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.