ABSTRACT Researchers are employing deep learning (DL) in many fields, and the scope of its application is expanding. However, because understanding the rationale and validity of DL decisions is difficult, a DL model is occasionally called a black-box model. Here, we focus on a DL-based explainable time-series prediction model. We propose a model based on long short-term memory (LSTM) followed by a convolutional neural network (CNN) with a residual connection, referred to as the LSTM-resCNN. In comparison to one-dimensional CNN, bidirectional LSTM, CNN-LSTM, LSTM-CNN, and MTEX-CNN models, the proposed LSTM-resCNN performs best on the three datasets of fine dust (PM2.5), bike-sharing, and bitcoin. Additionally, we tested with Grad-CAM, Integrated Gradients, and Gradients, three gradient-based approaches for the model explainability. These gradient-based techniques combined very well with the LSTM-resCNN model. Variables and time lags that considerably influence the explainable time-series prediction can be identified and visualized using gradients and integrated gradients.

INDEX TERMS Neural networks, recurrent neural networks, time series analysis.
In a recurrent neural network (RNN), past information is stored in the hidden layer. However, RNN models suffer from a long-term memory dependency problem in which long-term memory cannot be transferred to deep layers [13]. Long short-term memory (LSTM) model [14] can be used to modify the RNN structure to improve the learning of long-term dependence. Information from the previous stage is stored in a memory cell, enabling the transfer of long-term past information. Furthermore, bidirectional LSTM (BiLSTM) is an extension of the conventional LSTM model that additionally considers the reverse direction. Because two directions are considered in BiLSTM, the number of parameters is also twice that employed in unidirectional LSTM. Gated recurrent units (GRUs) [15] are used to simplify the LSTM structure and offer the advantage of faster learning speed and fewer parameters compared with LSTM. However, compared with LSTM, GRU-based BiLSTM does not always achieve better performance; hence, based on the requirements, an appropriate model selection must be performed.

Similar to RNN models, convolutional neural network (CNN) models can be used to handle time-series data. Typically, a one-directional (1D) CNN is used to extract features by moving along only the time axis, unlike two-dimensional (2D) CNNs, which extract features by moving along two directions using a square filter. The dilated causal convolution of WaveNet [16] involves a combination of causal and dilated convolutions. Using the dilated network, the model can achieve a wide acceptance range.

In a CNN followed by an RNN (CNN-RNN) model, the strengths of an RNN and a CNN are combined. CNN-RNN models have been employed widely because CNN simplifies the feature space, and the long-term dependency is modeled using LSTM or a GRU [17], [18], [19]. In an RNN followed by a CNN (RNN-CNN) model, an RNN and a CNN are combined in the opposite direction [20], [21], [22]. Deep learning models make use of a lot of variables and are challenging to analyze. It is also known as a “black box model,” and numerous experiments on interpretable AI are being done to add explanatory power. Integrated Gradient (IG) [23] synthesizes gradient values surrounding provided data to look for significant variables. Grad-CAM [24] also utilizes gradients to display class activation maps. MTEX-CNN [25] was suggested for time series analysis, and Grad-CAM was used on the 1D CNN-based MTEX-CNN model. However, the MTEX-CNN design solely uses 1D CNN layers, and 1D CNN layers cannot simulate long-term dependency.

We propose an LSTM-resCNN, LSTM-CNN with an additional residual connection on 1D CNN layers in this study. We included a residual connection to ensure that no critical information is lost in the process of three consecutive 1D convolution layers. To add explanatory power, we experimented with applying IG and Grad-CAM methods. The proposed LSTM-resCNN achieves better prediction performance than 1D CNN, BiLSTM, LSTM-CNN, and CNN-LSTM models on three datasets. Moreover, the application of Grad-CAM to LSTM-resCNN adds explanatory power as the Grad-CAM activation visualizes temporal features that significantly influence the prediction. The proposed LSTM-resCNN can achieve high predictive performance with additional interpretability on three datasets: fine dust (PM$_2.5$), bike-sharing, and Bitcoin. The key contributions of this study are summarized below.

- The proposed LSTM-resCNN architecture outperforms other popular DL models (1D-CNN, BiLSTM, CNN-LSTM, and LSTM-CNN) on three datasets.
- The Grad-CAM endows the proposed LSTM-resCNN with gradient-based explainability. We can identify time zones that play an essential role in prediction.
- To determine important variables in predictions, Gradients and Integrated Gradients are employed.

Related studies are introduced in Chapter 2. The model put out in this study is established in Chapter 3. In chapter 4, the proposed model was tested against three different datasets, and our suggested LSTM-resCNN was contrasted against other available techniques. Finally, the paper ends with discussion and conclusion section (chapters 5 and 6).

II. RELATED WORKS

A. HYBRID MODEL FOR FORECASTING

Many studies have shown that CNN-LSTM, which comprises a combination of a CNN layer followed by an LSTM layer, shows better prediction performance than single CNN or LSTM models [17], [18], [19]. Using PM2.5 data for multi-step prediction experiments, CNN-LSTM showed better prediction performance than LSTM [17]. In gold price predictions, CNN-LSTM outperformed support vector regression (SVR) and LSTM in terms of prediction accuracy [18]. In predicting house energy consumption, CNN-LSTM achieved the optimal prediction accuracy among ML techniques of linear regression, DTR, random forest regression, and multilayer perceptron [19]. In the Covid 19 hotspot prediction, CNN-LSTM outperformed SVR, ARIMA, and LSTM. [26]. Compared with DL techniques such as LSTM, GRU, BiLSTM, and attention-LSTM, CNN-LSTM achieved the optimal prediction accuracy. In the prediction of solar power generation, LSTM-CNN model was used to extract temporal and spatial feature information from data. As a result of the comparison with LSTM, CNN, CNN-LSTM, and LSTM-CNN, LSTM-CNN had the best predictive performance and took less training time than CNN-LSTM [21]. Moreover, based on this finding, research on gold price prediction revealed that the LSTM-CNN model has an advantage over the CNN-LSTM model (in which a CNN layer is applied as the first layer) [20]. This is because LSTM, the first layer of the LSTM-CNN model, influences the recording of the order of gold prices and the CNN layer optimizes the original input data to extract local features. In the CNN-LSTM model, the CNN layer can confuse the order of gold prices to some extent when extracting the pattern and thus cannot provide
the advantage of LSTM when disorganized data are provided as inputs.

**B. TIME-SERIES INTERPRETABILITY WITH GRAD-CAM**

In MTEX-CNN, a two-stage model structure is used to explain both time dimensions and variables [25]. In stage 1, a 2D CNN is employed. To determine the influence of variables on each prediction, an asymmetric CNN is applied with a kernel size of $k \times 1$. In stage 2, a 1D CNN is used, which considers only the time axis. Thus, the interaction between time and predictor can be explained by focusing on the temporal aspect of the data. Hence, MTEX-CNN can provide spatiotemporal explanations for multivariate time-series data based on the combination of Grad-CAM outputs.

**III. METHODOLOGY**

In this section, the proposed LSTM-resCNN model is explained. Further, other DL models used for experimental evaluations and comparisons are discussed.

**A. OVERVIEW**

The LSTM-resCNN model is a time-series prediction model that is an integration of existing LSTM and a 1D CNN with a skip connection. It offers the advantage of the model’s explainability by applying Grad-CAM to the last layer of the CNN.

Fig. 1 presents the structure of the LSTM-resCNN model. The model’s input is a fixed-size time-series sequence, and the output is a future value (one-step prediction) or a set of future values (multi-step prediction). The temporal and spatial features are extracted from the LSTM module and two CNN blocks, respectively. The advantage of this architecture is that we can directly apply the Grad-CAM method to the last 1D CNN layer. We can identify time zones that are instrumental in predictions using the Grad-CAM method. Moreover, variables and time point that significantly impact predictions can be determined using the Gradients (G) or IG techniques.

**B. LSTM**

Existing RNNs show considerably degraded learning ability in long-range relationships due to a gradual reduction in gradients during backpropagation. Such a phenomenon is called the vanishing gradient problem, and LSTM is designed to overcome this problem [14]. LSTM can read unknown and
persistent patterns between important events in time-series data.

The LSTM module comprises a cell ($c_t$), an input gate ($i_t$), an output gate ($o_t$), and a forget gate ($f_t$). The mathematical expression and graphical explanation of the LSTM module are described in the following and Fig. 2, respectively.

$$
\begin{align*}
    f_t &= \sigma(W_f \cdot x_t + W_{hf} \cdot h_{t-1} + b_f), \\
    i_t &= \sigma(W_i \cdot x_t + W_{hi} \cdot h_{t-1} + b_i), \\
    o_t &= \sigma(W_o \cdot x_t + W_{ho} \cdot h_{t-1} + b_o), \\
    g_t &= \tanh(W_g \cdot x_t + W_{hg} \cdot h_{t-1} + b_g), \\
    c_t &= f_t \otimes c_{t-1} + i_t \otimes g_t, \\
    h_t &= o_t \otimes \tanh(c_t),
\end{align*}
$$

where, $x_t$ denotes an input vector, $h_t$ represents an output vector and $W_f$, $W_{hf}$, $W_i$, $W_{hi}$, $W_o$, $W_{ho}$, $W_g$, $W_{hg}$ are estimable parameters related to the LSTM module.

Cells ($c_t$) memorize values over arbitrary time intervals, and the three gates regulate information flow in and out of the cell. The forget gate is controlled by $f_t$ and determines the part of the long-term state of $c_t$ to be deleted. The input gate is controlled by $i_t$ and controls the part of $g_t$ to be added to the long-term state of $c_t$. The output gate $o_t$ determines the part of the long-term state of $c_t$ to be read and output to $h_t$ and $y_t$. The three gates—called gate controllers—based on the sigmoid activation function close the gate when the output is 0 and open the gate when the output is 1.

C. 1D-CNN

The most commonly known CNN is a 2D CNN mainly employed in image processing. Alternatively, a 1D CNN is used in natural language processing and time-series analysis and it extracts features by moving the kernel along the time axis [15]. The 1D CNN layer contributes to the extraction of features inherent in the data, and the number of filters determines the number of traits to be learned, which is consistent with the dimension of the output space.

D. PROPOSED LSTM-resCNN WITH GRADIENT-BASED APPROACHES

In this paper, we suggest an LSTM-resCNN, which is an LSTM-CNN with an additional residual connection on 1D CNN layers. The reason for taking the order of LSTM-CNN rather than CNN-LSTM is to place the convolution layer to which Grad-CAM can be applied in the latter part to represent more complex features. We included a residual connection to avoid losing important information while applying three successive 1D convolution layers [27]. Also, residual networks demonstrate that they can be viewed as an assortment of numerous paths of different lengths, and these paths exhibit ensemble-like behavior [28].

1) MODEL DESIGN

The propose LSTM-resCNN model described in Table 1 and Fig. 1. Each CNN block comprises three 1D CNN layers with a filter size of 64 each and kernel sizes of 3, 2, and 1. The filter size of 64 was selected among 32, 64, and 128 based on the hyperparameter search. A residual connection is added to each CNN block.

After applying the two CNN blocks, a flatten layer is applied to convert the features into one-dimensional vectors. Next, a dropout layer is used to prevent overfitting. Finally, the result is transferred to the dense layer. The filter sizes are 1 and 24 for one-step and multi-step prediction models, respectively (predictions of 1 and 24 h, respectively).

2) GRAD-CAM APPLICATION TO LSTM-resCNN MODEL

Another advantage of using the RNN-CNN structure instead of the CNN-RNN structure is the application of Grad-CAM. The Grad-CAM technique has been proposed to visualize the part of the input feature that can be distinguished using the CNN model [24]. Thus, we use the activation gradient of the last CNN layer, which models abstract and complex high-level features [29]. Structurally, the CNN layer in the RNN-CNN structure is deeper than that in the CNN-RNN
structure. This implies that we can visualize the parts related to important high-level features for time-series predictions.

The proposed LSTM-resCNN model is defined as \( f(X) \), where \( X \) denotes the time-series matrix with \( T \times J \) inputs (\( T \) and \( J \) represent the time window size and number of features, respectively). The result of the model \( y = f(X) \) is the value predicted using the input time-series matrix \( X \). To apply the Grad-CAM on a specific CNN layer, we separate the LSTM-resCNN model \( f(\cdot) \) with \( h(\cdot) \) and \( g(\cdot) \). \( f(X) = h(tg(X)) \), where \( A = g(X) \) is the 2D activation matrix of a specific convolution layer. The activation \( A \) shows temporal and feature dimensions, respectively. The result of the model \( \alpha \) can be used to quantify the importance of the given position. Each filter can capture complex time-series data, and the gradient amount of \( \partial y/\partial A \) can be used to quantify the importance of the given position. To obtain the overall filter importance weights \( q_k \) for the \( k \)-th filter:

\[
q_k = \frac{1}{T} \sum_i \frac{\partial y}{\partial A_i^k}.
\]

By taking \( q_k \) as weights for the \( k \)-th channel activation matrix \( A^k \), we can estimate the Grad-CAM as follows:

\[
L_{1D-Grad-CAM} = ReLU\left( \sum_k q_k A^k \right),
\]

where ReLU denotes a rectified linear unit and \( A^k \) represents the \( k \)-th channel activation matrix from the activation tensor \( A \). \( L_{1D-Grad-CAM} \) denote the weighted average activation of the activated parts from each filter. Thus, the plot of \( L_{1D-Grad-CAM} \) for the \( i \)-th temporal component and \( k \)-th filter. The gradient amount of \( \partial y/\partial A \) can be used to quantify the importance of the given position. Each filter can capture complex time-series data, and the gradient amount of \( y \) with respect to \( A \) can be used to quantify the importance of the given position. To obtain the overall filter importance weights \( q_k \) for the \( k \)-th filter:

\[
q_k = \frac{1}{T} \sum_i \frac{\partial y}{\partial A_i^k}.
\]

By taking \( q_k \) as weights for the \( k \)-th channel activation matrix \( A^k \), we can estimate the Grad-CAM as follows:

\[
L_{1D-Grad-CAM} = ReLU\left( \sum_k q_k A^k \right),
\]

where ReLU denotes a rectified linear unit and \( A^k \) represents the \( k \)-th channel activation matrix from the activation tensor \( A \). \( L_{1D-Grad-CAM} \) denote the weighted average activation of the activated parts from each filter. Thus, the plot of \( L_{1D-Grad-CAM} \) as a function of the input time-series \( x \) can yield the important parts for the prediction.

Herein, the specific CNN layer is the activated last 1D CNN layer for the LSTM-resCNN, LSTM-CNN, and CNN-LSTM models. Note that the Grad-CAM is applied to a deeper layer in the LSTM-resCNN and LSTM-CNN models, which embed more complex features.

3) IG AND G FOR LSTM-resCNN MODEL

A disadvantage of Grad-CAM \( (L_{1D-Grad-CAM}) \) is that it can only visualize the temporal importance. To achieve the feature-level importance, a naive approach will involve the estimation of \( \partial y/\partial X \). Because the input matrix \( X \) include both temporal and feature dimensions, we can quantify both the temporal- and feature-level importance by visualizing \( G \).

\[
G = \frac{\partial y}{\partial X}.
\]

A large \( |\partial y/\partial X| \gg 0 \) indicates a high importance. However, this naive gradient break sensitivity [23]. Sundararajan et al. [23] proposed IG to overcome the drawbacks of the G method. We employed the integrated gradient technique for visually explaining the feature-level importance of the time-series models such as CNN-LSTM, LSTM-CNN, and LSTM-resCNN. Let \( X \) be the time-series matrix with \( T \times J \) inputs and \( X' \) be the baseline input matrix. The IG for time-series data can be defined as

\[
IG = (X - X') \times \int_0^1 \frac{\partial f(X' + \alpha \cdot (X - X'))}{\partial X} d\alpha.
\]

Computationally, it can be approximated as:

\[
IG = (X - X') \times \sum_{k=1}^m \frac{\partial f(X' + \frac{1}{m} \cdot (X - X'))}{\partial X} \times \frac{1}{m},
\]

where \( m \) denotes the number of steps in the Riemann approximation of the integral. Note that the resulting IG matrix is a \( T \times J \) matrix, similar to the input time-series matrix. We can approximately estimate the feature- and temporal-level importance using the IG matrix.

IV. EXPERIMENTS

A. DATASET

1) PM2.5

The PM2.5 dataset [30] is used herein to predict the fine dust concentration in Beijing using seven variables: dew point, temperature, pressure, wind direction, wind speed, snow cover, and precipitation. The variables were recorded every hour from January 1, 2010 to December 31, 2014. The total data length is 43,800, in which missing values are replaced by zero. The training, validation, and test data are divided at a ratio of 8:1:1 [31]. The test data are used only for model performance evaluations.

2) BIKE-SHARING

A bike-sharing dataset comprising data collected from Seoul, Korea [32], [33] is employed to predict the bike-sharing demand in Seoul. Since 2017, bike sharing has been increasing in Seoul because of convenient accessibility and near-zero waiting time. For a stable bike-sharing supply, the supply requirement information must be predicted for each hour. The hourly usage data are recorded from December 1, 2017 to November 30, 2018, and the total data length is 8760. The temperature, humidity, wind speed, visibility, dew point temperature, solar radiation, rainfall, and snowfall are used as explanatory variables. The missing values are replaced with the data of the same time a week ago, and the demand for bike sharing is considered as a dependent variable. The training, validation, and test data are divided at a ratio of 8:1:1.

3) BITCOIN

The Bitcoin dataset [34], [35] is used to predict its closing price. Nowadays, Bitcoin has been garnering considerable attention and the number of people interested in Bitcoin investment is increasing. Consequently, attempts to predict Bitcoin prices are increasing. The Bitcoin dataset comprises hourly data from December 12, 2019 to February 11, 2021, and the total data length is 10005. The Bitcoin closing price is deemed as a dependent variable. The opening, high, and low prices as well as volume-to and volume-from are explanatory variables. Volume-to refers to the volume of currency being
traded, and volume-from indicates the volume of base currency that things are traded into. The training, validation, and test data are divided at an 8:1:1 ratio.

B. PREPROCESSING
In DL, if the difference in the data scale of features are large, the model yields poor performance. Herein, the min–max normalization of 1 is performed on the training data.

\[
x' = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}},
\]

Thereafter, the data shape is modified by cutting it based on the length of time to be predicted. In the one-step prediction, in which the time after which the data of the same day and time can be included in the training data is predicted at least once, the prediction of the next hour is performed using the data of the previous 24 h (Fig. 6(a)). In the multistep prediction, in which multiple time points are simultaneously predicted using the previous data, the prediction of the next day is performed using the data of the previous 168 h [36], i.e., one-week data (Fig. 6(b)).

C. MODELS
We compare the 1D CNN, BiLSTM, LSTM-CNN, CNN-LSTM, and MTEX-CNN models with the proposed LSTM-resCNN model.

1) 1D-CNN
Fig. 3(a) presents the architecture of the studied 1D CNN model. The input sequence is transferred to one LSTM layer with a filter size of 64 and six sequential 1D CNN layers with a filter size of 64 and kernel sizes of 3, 2, 1, 3, 2, and 1. Then, 1D maxpooling with a size of two is applied and a flatten layer is applied. One output dense layer is applied for the one-step (with a filter size of 1) or the multi-step (with a filter size of 24) predictions.

2) BiLSTM
In BiLSTM, an additional LSTM layer is introduced that performs processing in a direction opposite to that of the existing LSTM layer. The final hidden state outputs a vector concatenating the hidden states of the two LSTM layers. Moreover, the number of parameters required is twice that required in unidirectional LSTM. Fig. 3(b) presents the BiLSTM architecture used herein. The input feature transfers one BiLSTM with a filter size of 64. Then, the additional output dense layer is applied for the one-step (filter size of 1) or multistep (filter size of 24) predictions.

3) MTEX-CNN
The MTEX-CNN model is taken from Assaf et al. [25] and shown in Fig. 3(c). The input sequence uses three sequential 2D CNN layers with filter size 16, 32,1 and kernel size (8,1), (4,1), 1. After that, a 1D CNN with a filter size of 64 and a kernel size of 3 is applied, and a flatten layer is applied. We used one output dense layer for one-step (filter size of 1) or multi-step (filter size of 24) predictions.

4) CNN-LSTM
Fig. 4(d) shows the CNN-LSTM architecture used herein. The input sequence is transferred to six sequential 1D CNN
layers with a filter size of 64 and kernel sizes of 3, 2, 1, 3, 2, and 1. Then, 1D maxpooling with a size of two is applied and one LSTM layer with a filter size of 64 transferred. One output dense layer is applied for the one-step (filter size of 1) or multistep (filter size of 24) predictions.

5) LSTM-CNN
Fig. 4(e) shows the LSTM-CNN architecture used herein. The input sequence is transferred to one LSTM layer with a filter size of 64 and six sequential 1D CNN layers with a filter size of 64 and kernel sizes of 3, 2, 1, 3, 2, and 1. Then, the flatten layer and one dense layer are added to achieve the results.

6) LSTM-resCNN
Fig. 5(f) presents the architecture of the proposed LSTM-resCNN. In this architecture, two residual connections are introduced in the six sequential 1D CNN layers of the LSTM-CNN model.

D. METRIC
The metrics of the root-mean-square error (RMSE), mean absolute error (MAE) and mean absolute percentage error (MAPE) are used for evaluating prediction performance; these metrics are commonly used analyzing the difference between predicted and actual values. When the values of the three metrics are small, the difference between the predicted and actual values is small, indicating enhanced prediction performance.

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

\[ MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \]

\[ MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \]

E. EXPERIMENTAL SETUP
All models are trained using TensorFlow version 2.3 with NVIDIA RTX 2080TI GPU & INTEL i9-9900K CPU. A dropout with rate parameter 0.5 is applied before the last dense layer in all models. We train 100 and 30 epochs for the one-step and multistep predictions. The Adam optimizer [37] is used.

F. EXPERIMENTAL RESULTS
1) ONE-STEP PREDICTION
A commonly known prediction technique is one-step prediction [36], in which the next one-time point is predicted. To quantitatively evaluate the performance of the proposed LSTM-resCNN model, its prediction performance was compared with those of the BiLSTM, 1D CNN, CNN-LSTM, LSTM-CNN, and MTEX-CNN models. Furthermore, the
same number of parameters is used in each model for a fair performance comparison based on the number of parameters in the models. Table 2 presents the training time, evaluation time, and the number of parameters used in each model. The difference between the actual and predicted values for each model is measured on the testing data, and the results are shown in Table 3. The best-performing values are indicated in bold. On the bike-sharing dataset, LSTM-resCNN achieved MAE, RMSE, and MAPE values of 0.0226, 0.0339, and 0.3125, respectively. LSTM-resCNN and CNN-LSTM achieved the best performance in terms of the MAE, RMSE, and MAPE values. No statistically significant differences were observed in the values obtained using LSTM-resCNN and CNN-LSTM (two-sample t-test with Bonferroni adjustment; all p values > 0.1). However, LSTM-resCNN outperformed the third-best LSTM-CNN model in terms of the MAE, RMSE, and MAPE values. No statistically significant differences were detected in the values obtained using LSTM-resCNN, LSTM-CNN, and CNN-LSTM (two-sample t-test with Bonferroni adjustment; all p values < 0.1). However, LSTM-resCNN outperformed the 1D CNN and BiLSTM models in terms of the MAE, RMSE, and MAPE values. On the Bitcoin dataset, LSTM-resCNN showed the best performance in terms of the MAE and RMSE, achieving values of 0.0138 and 0.0182, respectively (two-sample t-test with Bonferroni adjustment; all p values < 0.001). MTEX-CNN achieved top performance in terms of the MAPE value but the result was not statistically significantly different from the MAPE value of LSTM-resCNN (two-sample t-test with Bonferroni adjustment; all p values > 0.01).

2) MULTI-STEP PREDICTION
Multistep prediction is useful because it can be used to establish countermeasures based on predictions for the next day, month, and year from now. Among the various methods
available for multistep predictions, recursive, direct, and multi-input multi-output (MIMO) methods [17], [36], [38] are most commonly used. In the recursive method, the first time point prediction in the training data is included and retrained.
In the direct method, multiple time points are predicted using independent models for each step. MIMO is a single model that takes multiple inputs and predicts multiple outputs using a single model. The MIMO method is adopted in this study. It offers the advantage of preventing the accumulation of prediction errors.

Fig. 7 shows a comparison of the actual and predicted values for 24-h data using the proposed LSTM-resCNN model. The black line represents the input data for predicting the next 24-h data. Red and blue points indicate the actual and predicted values, respectively. The bike-sharing dataset shows periodic weekly patterns. CNN-LSTM failed to record such picks. However, LSTM-CNN and LSTM-resCNN could capture such picks, with the LSTM-resCNN model outperforming the LSTM-CNN model. The PM2.5 and Bitcoin datasets do not show periodic patterns. Consequently, LSTM-resCNN shows better performance than CNN-LSTM and LSTM-CNN.

For the quantitative evaluation of the multi-step prediction model, 15 non-overlapping random time points are selected. Table 4 shows the average MAE, RMSE, MAPE values with standard deviations for each model. The highest-performing values are indicated in bold. The proposed LSTM-resCNN model showed the best performance in all cases, except for the MAPE value on the bike-sharing dataset. However, no statistically significant differences were observed between the proposed and second-best-performing models (two-sample t-test with Bonferroni adjustment; all p values > 0.1).

**G. INTERPRETABILITY USING GRAD-CAM**

1) GRAD-CAM

To evaluate the performance of Grad-CAM with LSTM-resCNN, we considered four one-step prediction models: MTEX-CNN, CNN-LSTM, LSTM-CNN, and LSTM-resCNN. Here, Grad-CAM is applied to the last CNN layer of the MTEX-CNN, CNN-LSTM, LSTM-CNN, and LSTM-resCNN models. We extracted the Grad-CAM activation features and normalized them between values of 0 and 1. Fig. 8 presents the Grad-CAM activation of randomly selected periods on the bike-sharing, PM2.5, and Bitcoin datasets using the MTEX-CNN, CNN-LSTM, LSTM-CNN, and LSTM-resCNN models. The activation value closer to 1

| Models   | Bike-sharing MAE | Bike-sharing RMSE | Bike-sharing MAPE | PM2.5 MAE     | PM2.5 RMSE  | PM2.5 MAPE   | Bitcoin MAE  | Bitcoin RMSE | Bitcoin MAPE |
|----------|------------------|-------------------|-------------------|---------------|-------------|--------------|--------------|--------------|--------------|
| MTEX-CNN | 0.079 ± 0.061    | 0.53              |                   | 0.043 ± 0.037 | 0.353       |              | 0.045 ± 0.039 | 0.052        |
| CNN-LSTM| 0.083 ± 0.056    | 0.387             |                   | 0.044 ± 0.038 | 0.34        |              | 0.048 ± 0.045 | 0.058        |
| LSTM-CNN| ± 0.018 ± 0.015  | ± 0.426           | ± 0.262           | ± 0.033       | ± 0.29      | ± 0.246      | ± 0.029       | ± 0.028      | ± 0.034      |
| LSTM-resCNN | ± 0.03 ± 0.024  | ± 0.298           | ± 0.043           | ± 0.036       | ± 0.033     | ± 0.188      | 0.044 ± 0.041 | 0.053        |
| LSTM-resCNN | 0.063 ± 0.052  | 0.426             | ± 0.036           | ± 0.035       | ± 0.303     |              | 0.035 ± 0.03  | 0.041        |

### TABLE 4. The MAE, RMSE, MAPE performance on three datasets for each model using multi-step prediction rule (mean ± standard deviation).
FIGURE 9. Feature importance.

(a) Gradient importance

(b) Integrated Gradient importance
is represented by light colors, and it implies more influence on the prediction. The results obtained using MTEX-CNN, CNN-LSTM, LSTM-CNN, and LSTM-resCNN are presented in the first, second, third, and fourth columns, respectively. The results for the bike-sharing, PM$_{2.5}$, and Bitcoin datasets are presented in the first, second, and third rows, respectively. For the bike-sharing dataset, note that the model predicts the 24-h data points (predictions for a day) using weekly data. In the bike-sharing data, the first five similar pick patterns are obtained for Monday–Friday and the following two patterns are for Saturday and Sunday. The data for Monday–Friday exhibited similar patterns than those for Saturday and Sunday. Two picks from the weekends were estimated to be for bike commuting, going to work, and going back home. The Grad-CAM visualization of the CNN-LSTM model achieves two highly activated positions. However, elucidating the reason for these activations is difficult. Alternatively, the LSTM-CNN and LSTM-resCNN models focus more on weekday information than on weekend information, which is more reasonable because the patterns for the next Monday are being predicted. In particular, the LSTM-resCNN model focuses more on Monday data compared to MTEX-CNN, CNN-LSTM and LSTM-CNN models. It would be reasonable to concentrate on Monday’s information more for future Monday predictions. On the PM$_{2.5}$ dataset, CNN-LSTM focuses on past pick information, which may not be reasonable. The LSTM-CNN and LSTM-resCNN models focus on plateau areas. In particular, the LSTM-resCNN model focuses on the large plateau areas for the next multistep prediction. LSTM-resCNN also focuses on the front areas. There must be a reason, but the model does not explicitly explain it, thus sometimes guesses are required. On the Bitcoin dataset, all models highly focus on the most recent periods for the next one-step prediction. This is reasonable because the Bitcoin price data may not show any striking pattern. The most recent data would have more information for predicting the future values.

2) VISUALIZING FEATURE-LEVEL IMPORTANCE

One disadvantage of Grad-CAM is that the result can only visualize the temporal-level importance. The feature-level importance can be visualized using the G and IG. Figs. 9(a) and (b) show the G and IG activations of previously selected periods from Fig. 8 from the bike-sharing, Figs. 9(a) and (b) show the G and IG activations of previously selected periods from Fig. 8 from the bike-sharing, PM$_{2.5}$, and Bitcoin datasets by employing the CNN-LSTM, LSTM-CNN, and LSTM-resCNN models. The activation value closer to 1 is indicated by darker colors than that closer to 0, implying more influence on the prediction. The results obtained using CNN-LSTM, LSTM-CNN, and LSTM-resCNN are depicted in the first, second, and third columns, respectively. The results for the bike-sharing, PM$_{2.5}$, and Bitcoin datasets are presented in the first, second, and third rows, respectively. Nine variables of the bike-sharing data are used: rented bike count, temperature, humidity, wind speed, visibility, dew point temperature, solar radiation, rainfall, and snowfall. The most crucial variable found using all the three models based on the G and IG for predicting the rented bike count variable from the most recent past, which is logically. The CNN-LSTM model was the most activated on Sunday, whereas the LSTM-CNN and LSTM-resCNN models were activated all week. On Monday, the LSTM-resCNN model with the G achieved more activation than CNN-LSTM and LSTM-CNN models. This is logical because it is a model for predicting the data points of the following Monday. Eight variables of the PM$_{2.5}$ data are employed: PM$_{2.5}$, dew point, temperature, pressure, wind direction, wind speed, snow cover, and precipitation. For the prediction of PM$_{2.5}$, the most important variable found using all the three models with the G and IG was the PM$_{2.5}$ value from the most recent past. Six variables of the Bitcoin data are employed: closing price, opening price, high price, low price, volume-to, and volume-from. On the Bitcoin dataset, the most important variables found using the three models with the G and IG are the closing, opening, high, and low prices from the most recent past. This is logical because the target variable (close price) is highly correlated with the previous day’s closing, opening, high, and low prices.

V. DISCUSSION

In this study, Explainability is introduced in the model using the principle of Grad-CAM. Based on a previous study, Grad-CAM is also applied to the time-series prediction model using the MTEX-CNN [25]. However, because the Grad-CAM can be applied to only CNN layers, it comprises only 1D CNN layers. The long-term dependencies are poorly modeled because models based on the Grad-CAM solely use CNN layers. We attempted to reflect the long-term dependency of LSTM by applying Grad-CAM to the LSTM-CNN architecture. Furthermore, we performed an experiment on the addition of a residual connection to 1D CNN layers. The LSTM-resCNN model achieved slightly better performance and visualization results than the LSTM-CNN model on the three datasets.

VI. CONCLUSION

Deep learning is widely used in time series problems. However, studies on explainable DL methods in the field of time series are lacking. In this paper, we propose an explanatory time-series prediction DL model, LSTM-resCNN, with a residual connection added to the LSTM-CNN model. The proposed model showed better performance than one-dimensional CNN, bidirectional LSTM, CNN-LSTM, LSTM-CNN, and MTEX-CNN models using the three different datasets of fine dust (PM$_{2.5}$), bike-sharing, and bitcoin. LSTM-resCNN model achieved the best performance on six of nine evaluation criteria for one-step prediction scenarios, and the LSTM-resCNN model achieved the best performance on eight of nine evaluation criteria for multi-step prediction scenarios. Furthermore, we experimented with adding explanatory power to the proposed LSTM-resCNN model by applying gradient-based techniques: Grad-CAM, Integrated
Gradients, and Gradients. The proposed LSTM-resCNN model worked well with these gradient-based approaches using three experimental datasets. As a future study, an interpretable AI model incorporating Transformer [39] will be interesting. Due to its strong performance, the Transformer model is garnering interest in a number of domains, including language [40], speech [41], and vision [42]. In time series studies, attempts to include a Transformer are also growing [43]. Conformer [44] is a model with 1D convolution added before the multi-head attention layer. Experiments such as applying Grad-CAM to the 1D CNN layer or placing the 1D CNN layer further back will be an interesting future study.

REFERENCES

[1] T. Hong, P. Pinson, Y. Wang, R. Weron, D. Yang, and H. Zareipour, “Energy forecasting: A review and outlook,” IEEE Open Access J. Power. Energy, vol. 7, pp. 376–388, 2020.

[2] C. Paoli, C. Voyant, M. Muselli, and M.-L. Nivet, “Forecasting of preprocessed daily solar radiation time series using neural networks,” Solar. Energy, vol. 84, no. 12, pp. 2146–2160, Dec. 2010.

[3] H. Rezaei, H. Fadlouj, and G. Mansourfar, “Stock price prediction using deep learning and frequency decomposition,” Exp. Syst. Appl., vol. 169, May 2021, Art. no. 114532.

[4] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, “Stock price prediction using the ARIMA model,” in Proc. UKSim-AMSS 16th Int. Conf. Comput. Model. Simul., Mar. 2014, pp. 106–112.

[5] P. K. Nalajam and R. Varadarajan, “A hybrid deep learning model for layer-wise melt pool temperature forecasting in wire-arc additive manufacturing using neural networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 30, I. Guyon, U. V. Luxburg, S. Bengio, Eds. Berlin, Germany: Springer, 2017, pp. 62–77, doi: 10.1007/978-3-642-33748-7_6.

[6] A. Zeroual, F. Harrou, A. Dairi, and Y. Sun, “Deep learning methods for forecasting COVID-19 time-series data: A comparative study,” Chaos, Solitons Fractals, vol. 140, Nov. 2020, Art. no. 110121.

[7] V. K. R. Chimmula and L. Zhang, “Time series forecasting of COVID-19 transmission in Canada using LSTM networks,” Chaos, Solitons Fractals, vol. 135, Jun. 2020, Art. no. 109864.

[8] J. Contreras, R. Espinola, F. J. Nogales, and A. J. Conejo, “ARIMA models to predict next-day electricity prices,” IEEE Trans. Power. Syst., vol. 18, no. 3, pp. 1014–1020, Aug. 2003.

[9] C. S. Bojer and J. P. Meldgaard, “Kaggle forecasting competitions: An overlooked learning opportunity,” Int. J. Forecasting, vol. 37, no. 2, pp. 587–603, Apr. 2021.

[10] S. Balli, “Data analysis of COVID-19 pandemic and short-term cumulative case forecasting using machine learning time series methods,” Chaos, Solitons Fractals, vol. 142, Jan. 2021, Art. no. 110512.

[11] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning Springer Series in Statistics. New York, NY, USA: Springer, 2001.

[12] S. Jin and H. Cho, “Data analysis of COVID-19 pandemic and short-term cumulative case forecasting using machine learning time series methods,” J. Korean Data Anal. Soc., vol. 22, no. 5, pp. 1779–1791, 2020.

[13] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent,” IEEE Trans. Neural. Netw., vol. 5, no. 2, pp. 157–166, Mar. 1994.

[14] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural. Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[15] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” in Proc. NIPS Workshop Deep Learn., Dec. 2014, pp. 1–9.

[16] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, “WaveNet: A generative model for raw audio,” in Proc. 9th ISCA Speech Synth. Workshop, Sunnyvale, CA, USA, Sep. 2016, p. 125.

[17] T. Li, M. Hua, and X. Wu, “A hybrid CNN-LSTM model for forecasting particulate matter (PM2.5),” IEEE Access, vol. 8, pp. 26933–26940, 2020.

[18] I. E. Livieris, E. Pintelas, and P. Pintelas, “A CNN-LSTM model for gold price time-series forecasting,” Neural. Comput. Appl., vol. 32, no. 23, pp. 17351–17360, 2020.
[41] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, “wav2vec2.0: A framework for self-supervised learning of speech representations,” in Proc. Adv. Neural Inf. Process. Syst., vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds. Red Hook, NY, USA: Curran Associates, 2020, pp. 12449–12460.

[42] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16 × 16 words: Transformers for image recognition at scale,” 2020, arXiv:2010.11929.

[43] H. Wu, J. Xu, J. Wang, and M. Long, “Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting,” in Proc. Adv. Neural Inf. Process. Syst., vol. 34, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds. Red Hook, NY, USA: Curran Associates, 2021, pp. 22419–22430.

[44] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, “Conformer: Convolution-augmented transformer for speech recognition,” in Proc. Interspeech, Oct. 2020, pp. 5036–5040.

HYOJUNG CHOI received the B.S. degree in statistics from Duksun Women’s University, Seoul, South Korea, in 2019, and the M.S. degree in statistics from Chung-Ang University, Seoul, in August 2021. Since 2021, she has been working at CJ Logistics, TES Logistics Technology Research Center. Her research interests include sequential data analysis such as time-series and audio data.

CHANHWI JUNG received the B.S. degree in statistics from Yonsei University, Wonju, South Korea, in 2021. He is currently pursuing the M.S. degree in statistics with Chung-Ang University. His research interests include machine learning and deep learning models with time-series data.

TAEIN KANG was born in Busan, South Korea, in 1996. He received the B.S. degree in applied statistics from Chung-Ang University, Seoul, South Korea, in 2021, where he is currently pursuing the M.S. degree in statistics. His research interest includes deep learning models with audio data.

HYUNWOO J. KIM received the Ph.D. degree in computer science (minor is statistics) from the University of Wisconsin-Madison, in 2017. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Korea University. Prior to the position, he worked at Amazon Lab126, Sunnyvale, CA, USA. His research interests include geometric deep learning for graphs and manifolds, manifold statistics, machine learning, computer vision, and medical imaging.

IL-YOUP KWAK (Associate Member, IEEE) was born in Seoul, South Korea, in 1981. He received the B.S. degree in mathematics and the M.S. degree in statistics from Korea University, and the Ph.D. degree in statistics from the University of Wisconsin-Madison, Wisconsin, USA, in 2014. He was a Postdoctoral Associate at the University of Minnesota-Twin Cities, from 2014 to 2017. He worked at Samsung Research, from 2017 to 2019. Since 2019, he has been an Assistant Professor at the Applied Statistics Department, Chung-Ang University. His research interests include deep learning applications and statistical genetics.

***