Hindawi
Journal of Electrical and Computer Engineering
Volume 2019, Article ID 7343784, 14 pages
https://doi.org/10.1155/2019/7343784

*Research Article*

# Seismic Events Prediction Using Deep Temporal Convolution Networks

**Yue Geng**,[1] **Lingling Su**,[2] **Yunhong Jia**,[1] **and Ce Han**[1]

[1]*School of Mechanical Electronic & Information Engineering, China University of Mining and Technology, Beijing 10083, China*
[2]*Guanghe Xinzhi (Beijing) Technology Co. Ltd, Beijing 100015, China*

Correspondence should be addressed to Yue Geng; danielgy19890310@gmail.com

Received 26 October 2018; Revised 16 February 2019; Accepted 12 March 2019; Published 2 April 2019

Academic Editor: Ioannis Valavanis



Seismic events prediction is a crucial task for preventing coal mine rock burst hazards. Currently, this task attracts increasing research enthusiasms from many mining experts. Considering the temporal characteristics of monitoring data, seismic events prediction can be abstracted as a time series prediction task. This paper contributes to address the problem of long-term historical dependence on seismic time series prediction with deep temporal convolution neural networks (CNN). We propose a dilated causal temporal convolution network (DCTCNN) and a CNN long short-term memory hybrid model (CNN-LSTM) to forecast seismic events. In particular, DCTCNN is designed with dilated CNN kernels, causal strategy, and residual connections; CNN-LSTM is established in a hybrid modeling way by utilizing advantage of CNN and LSTM. Based on these manners, both of DCTCNN and CNN-LSTM can extract long-term historical features from the monitoring seismic data. The proposed models are experimentally tested on two real-life coal mine seismic datasets. Furthermore, they are also compared with one traditional time series prediction method, two classic machine learning algorithms, and two standard deep learning networks. Results show that DCTCNN and CNN-LSTM are superior than the other five algorithms, and they successfully complete the seismic prediction task.

## 1. Introduction

Underground coal mines are different from general permanent tunnel engineering; like subways, their stopes are constantly moving with the mining activities ongoing. Affected by mining disturbance, the force equilibrium status of coal is destroyed and the internal stress of coal and rock is redistributed. Due to the above situation, rock burst disasters occur frequently. Seismic events prediction can directly reflect the safety conditions of underground coal mine, and it helps preventing rock burst accidents and hazards effectively. Therefore, seismic forecasting is a crucial guarantee for coal mine safety production. Previously, conventional predictors are mostly based on classic geomechanics, applying unified indexes to evaluate all coal mine roofs [1, 2]. However, the mechanism of roof disaster has not been thoroughly studied; it is difficult to accurately establish geomechanical models for simulating the occurrence processes of roof disasters.

Several coal mine safety scholars have attempted to explore some data-driven works [3]. Considering the temporal characteristics of coal mine seismic data, they abstracted underground coal mine seismic prediction as a time series regression task. Time series regression or prediction is a top ten challenge in data mining [4]; it plays an extremely important role in many domains. Time series prediction has attracted increasing research enthusiasms from different communities over the past years. A lot of classic prediction methods have been proposed, like ARIMA [3]. Traditional time series prediction approaches have good mathematical and theoretical explanation, but some of those classic methods treat time series with linear characteristic hypothesis, while, seismic time series is typical nonlinear data. Thus, applying those conventional methods on coal mine seismic prediction is not appropriate.

Recently, artificial intelligence technologies like machine learning have been successfully applied in computer vision, audio synthesis, and natural language processing [5]. It

inspired some mining experts to utilize classic machine learning algorithms on mining safety areas, like support vector machines (SVM) [6–9] and random forests (RF) [10, 11]. These machine learning algorithms rely on high quality and hand-crafted statistic or domain feature engineering, which is the foundation to guarantee the forecasting effects. However, manually defining or extracting detailed features from the monitoring time series is generally time consuming and laborious. Moreover, hand-crafted domain features require the cooperation of mining experts, which might be greatly influenced by subjective factors. In addition, if the original monitoring data are directly fed to the abovementioned algorithms, the relevant models could output inaccurate prediction which will lead to false risk alarms.

In the realms of time series prediction, some efforts were spent on addressing the above issues with deep learning algorithms in the past few years. The most popular deep sequential models are recurrent neural networks (RNNs). Except receiving the signals from the previous layer, each layer of RNN can additionally learn its own historical information. Based on this mechanism, RNN is naturally suitable for predicting time series. However, the application of standard RNN is still limited by some training problems, like gradients vanishing [12, 13]. Long short-term memory network (LSTM) is a modified RNN model which was designed with gated mechanism to avoid these problems. With long-term memory ability, LSTM has been the preferred model for time series prediction in the field of deep learning [14].

While some recent studies have shown that convolutional networks can achieve similar results and even surpass LSTM on time series prediction tasks [15–17], standard convolution neural networks (CNNs) were designed for image processing. After modified with 1D convolution kernels, CNN can be used to forecast time series. The modified CNN can be called as temporal CNN, which could automatically learn time-translation invariant features from time series. In order to make temporal CNN extract long-term historical features, some scholars proposed casual strategy and dilated CNN kernels [15, 17]. Those improved networks are capable of learning time series data autoregressively and memorizing historical long-term information with larger convolutional receptive fields. Based on the modified temporal CNNs, some experts conducted a series of explorations on audio synthesis [17], financial index prediction [15], power load forecasting [18], mechanical fault diagnosis [19], and urban water level prediction [20].

Despite the success of temporal convolutional networks in all these areas and time series prediction applications, there has not yet been an effort to apply deep temporal convolutional networks on the field of coal mine seismic events prediction. Zhou et al. [21] discussed different evaluation methods of mining rock bust; they found that deep learning techniques outperform shallow machine learning algorithms in almost all categories where data are plentiful. The motivation of this paper is to attempt this idea on such application. In particular, we aim to address the long-term historical dependence issue of underground coal mine seismic events prediction with deep temporal convolutional networks. The focus of this paper is to propose deep temporal convolutional models that can be successfully applied for underground coal mine seismic events prediction. Specially, the contribution and novelty of this paper are as follows:

(1) Abstract the underground coal mine seismic events prediction into time series forecasting task

(2) Two generative deep learning models are proposed to deal with the long-term memory problem: dilated causal temporal convolutional network (DCTCNN) and CNN-LSTM hybrid network

(3) Improve the standard deep learning network to address the long-term historical dependence in time series prediction

(4) Design the modified deep learning networks through expanding CNN local receptive field and hybrid modeling

(5) Test the proposed models on two real-life coal mine seismic datasets and compare them with some classic algorithms

Section 1 starts with an introduction. Brief reviews of related works are given in Section 2. Section 3 describes the proposed models. Section 4 presents the experimental setup. Section 5 provides the results and discussion. The conclusion is given in Section 6.

## 2. Related Works

Previously, some mining experts studied the hazard prediction methods based on time series regression. Wang et. al. [3] applied an autoregressive sliding integral seasonal product model (ARIMA) to predict coal mine water inflow. They smoothed underground water inflow data through standard differential method. However, their predicted values were linearly related to the input monitoring time series, which is not in accord with the real-life situation.

In order to overcome the problem that the above method cannot fit nonlinear monitoring time series, many researchers have applied classic machine learning algorithms. They proved that machine learning algorithms are more suitable for underground coal mine risk prediction. Hui et. al. [22] proposed a chaotic neural network for predicting underground coal mine rock burst hazards with monitoring seismic time series. Jian et. al. [6] applied SVM to predict long-term rock burst, and they combined SVM with particle swarm optimizing algorithm to forecast roof failures in a large scale underground goaf [9]. Similarly, Juisheng and Thedja [7] used intelligent firefly algorithm to optimize least squared SVM for coal mine disaster forecasting. Yahui et. al. [8] combined genetic algorithm with SVM to assess coal mine rock burst risk. Except SVM, Jian et. al. [10, 11] also used RF to forecast roof bolt support stability and rock burst hazards. These machine learning based algorithms can relieve the shortcomings of traditional temporal sequences prediction approaches, but there are still problems of overreliance on hand-crafted features and inability to process large scale data.

Due to the abilities of automatic feature extraction and large-scale data processing, deep learning has been a popular research field. The mostly used deep temporal models are recurrent networks, especially LSTMs. Theoretically, RNN is considered with the ability to memorize infinitely long sequences. However, standard RNN could not handle the problem of gradient vanishing [12, 13]. Gradients at a certain moment become too small to finish the backpropagation of RNN, which could lead to the termination of training process and the long-term historical information cannot be memorized. To address this issue, Hochreiter and Schmidhuber [23] designed LSTM with the gated mechanism. Laptev et. al. [24] and Lingxue and Nikolay [25] applied LSTM to forecast the waiting time of Uber passengers. Flunkert et. al. [26] proposed an autoregressive and probabilistic prediction network (DeepAR) to forecast retail volume and electrical and traffic load. Fernández-Navarro et. al. [27] applied an autoregressive strategy to improve the long-term memory ability of RNN.

The recurrent structures and long-term memory abilities of LSTMs are natural advantages on temporal sequence prediction. Many scholars of deep learning community regard these networks as the preferred methods for time series forecasting [5, 14]. However, some studies have shown that CNN based models can also achieve similar or better results [15–17]. They modified standard 2D CNNs into 1D temporal CNNs, which can be applied as a feature extractor to automatically learn time-translation invariance features. Haytham et. al. [20] used 1D CNN to predict urban water level; Alberto et. al. [28] and Yunxuan et. al. [29] applied those improved convolutional networks to forecast traffic load, wind speed, and financial time series, respectively.

Although the abovementioned 1D CNN can fit the data structure of time series, they did not take the long-term dependence issue into account. The modified convolutional networks cannot memorize long-term historical information. To deal with this problem, Google's Deep Mind team proposed a generative model called WaveNet for audio synthesis [17]. The main contributions of WaveNet are the causal strategy, dilated CNN kernels, and residual connection mechanism. Causal strategy autoregressively creates a recurrent structure on convolutional network, dilated kernels allow CNN obtaining the long-term historical memory, and skip connection avoids the omission of useful information. Based on the idea of WaveNet, Anastasia et. al. [15] proposed an improved dilated convolutional network for financial time series prediction; Bai et. al. [16] designed a generative network with causal and dilated CNN filters for music and voice temporal sequences generation.

Another idea of solving the long-term memory issue is to combine temporal CNN and RNN or LSTM and utilize the advantages of these two networks to complete time series prediction tasks. He [18] and Wu and Tan [30] proposed hybrid deep learning models, in which CNN and RNN were connected in a serial way for short-term power and traffic load forecasting. Rui et. al. [31] designed a convolutional bidirectional LSTM network for motor fault diagnosis. Lai et. al. [32] proposed a traffic load forecasting using diffusion convolutional recurrent network.

The above works have provided valuable research inspiration for us. A dilated convolutional network (DCTCNN) and a CNN-LSTM hybrid network are proposed in this paper to address the long-term historical dependence issue of underground coal mine roof risk prediction tasks.

## 3. Deep Temporal CNN Generative Network

Given a time series $\{x\}$ of length $N$ and a prediction model with parameter $\theta$, this model would predict the next time step value $x(t+1)$ based on the previous $t$ steps of the input data. The likelihood function can be formulized as the following equation:

$$p(x \mid \theta) = \prod_{t=0}^{N-1} p(x(t+1) \mid x(0), \ldots, x(t), \theta). \tag{1}$$

For learning the likelihood $p(\cdot)$ of the prediction model, this paper establishes two deep temporal CNN-based generative networks in Sections 3.1 and 3.2. In particular, we apply multivariable time series input which means the input is $[\mathbf{x}(0), \ldots, \mathbf{x}(N-1)]$.

*3.1. Dilated Causal Temporal CNN.* Time series data often exhibit long-term dependence and correlation. In order to make temporal CNN capable of extracting long-term historical features, we utilize dilated convolution kernel mechanism which can be defined as

$$\left(\mathbf{w}^l_{h*d}\mathbf{f}^{l-1}\right)(i) = \sum_{j=-\infty}^{\infty} \sum_{m=1}^{M_{i-1}} \mathbf{w}^i_h(j,m)\mathbf{f}^{l-1}(i-j,m), \tag{2}$$

where $(\mathbf{w}^l_{h*d}\mathbf{f}^{l-1})(i)$ is the $i^{\text{th}}$ convolutional kernel feature mapping of the $l^{\text{th}}$ layer, $\mathbf{w}$ is the weight parameter, $\mathbf{f}^{l-1}$ denotes the activation function of the previous layer, and $h$ and $d$ represent the number of cells and the dilated coefficient respectively. When the current computed data are the $j^{\text{th}}$ time step value, the current channel is $m$ and $M_{i-1}$ is the numbers of all channels.

In the input time series of dilated kernel, convolutional operation is performed every $d$ time steps. Through this manner, temporal CNN can effectively learn the historical information which is far from the current time step. Suppose the number of dilated CNN layers is $l \in \{1, \ldots, L\}$. The dilated coefficient is increased by powers of constant two per layer, namely, $d \in \{2^0, 2^1, \ldots, 2^{L-1}\}$. The convolutional filter size is $1 \times k$. The local receptive field $r$ of each neuron is the set of changed elements in the previous output. For example, the receptive field of the $L^{\text{th}}$ layer is $2^{L-1} \times k$, which means it has been enlarged $2^{L-1}$ times of the original ones. A three-layer dilated CNN can be illustrated as Figure 1.

Through the dilated coefficient, convolutional filter size, and layer number, we can control the size of the enlarged local receptive field which would benefit from obtaining more historical information. In addition, we applied causal strategy to make sure that the local perceptive field only contains $x(0), \ldots, x(t)$ when predicting $x(t+1)$. Causal strategy can avoid the influence of future information on

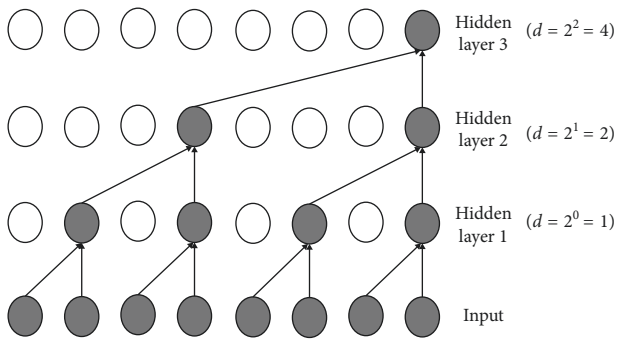


Figure 1: Three-layer dilated CNN.

convolutional layers. This is equivalent to filling the input time series with zero vector

$$[0, \ldots, 0, \mathbf{x}(0), \ldots, \mathbf{x}(N-1)] \in \mathbb{R}^{N+r}. \tag{3}$$

Output of the $L^{\text{th}}$ dilated CNN is

$$[\widehat{x}(1), \ldots, \widehat{x}(N)] \in \mathbb{R}^{N+1}, \tag{4}$$

where $\widehat{x}(N)$ is the predicted value at time step $N$.

During the testing phase, the predicted value of every single step is obtained by feeding the trained network with $[x(t+1-r), \ldots, x(t)]$, where $t+1 > r$, while multiple step predicted values are conducted in an autoregressive way. For example, when the model gives the prediction of $\widehat{x}(t+2)$, it is fed with $[x(t+2-r), \ldots, x(t), \widehat{x}(t+1)]$. According to the above description of DCTCNN, the conditional expectation of predicted value on each time step can be formulized as

$$\begin{aligned} E[x(t+1) \mid x(t), \ldots, x(t-r)] = {} & e_1(x(t-r)) + \ldots \\ & + e_i(x(t-r+i)) \\ & + e_r(x(t)), \end{aligned} \tag{5}$$

where $e_i (i = 1, .., r)$ is the previous expectation on the $i^{\text{th}}$ dilated neuron; it is learnable through the model.

Due to the sparse connection and parameter sharing mechanism of CNN, the above-described DCTCNN could automatically learn the translation-invariant features from input time series. Furthermore, it can also reduce the amount of model parameters. The objective function is formulized as equation (6). The weights of DCTCNN are trained to minimize the mean squared error (MSE). To avoid overfitting, L2 regularization is applied. Regularization can make the network get the balance between data fitting and model complexity during the training phase; it can prevent the excessive weight matrix. Thus, L2 regularization could avoid overfitting and improve the generalization of DCTCNN.

$$E(\mathbf{w}) = \frac{1}{N} \sum_{t=0}^{N-1} (\widehat{x}(t+1) - x(t+1))^2 + \frac{\lambda}{2} \sum_{l=0}^{L} \sum_{h=1}^{M_{l+1}} \left(\mathbf{w}_h^l\right)^2, \tag{6}$$

where $\lambda$ is the regularization term, $\mathbf{w}_h^l$ denotes the weight parameter, and $\widehat{x}(t+1)$ is the predicted value of $x(t+1)$ by using $x(0), \ldots, x(t)$.

From the perspective of Bayesian theory, the above-mentioned minimized function is equivalent to maximizing the posterior probability. Furthermore, it is subject to a Laplace distribution with the center of $\widehat{x}(t+1)$ and the scale of 0.5.

$$p(x(t+1) \mid x(0), \ldots, x(t), \theta) \sim \text{Laplace}\left(x(t+1), \beta = \frac{1}{2}\right). \tag{7}$$

The purpose of model training is to find weight parameters that minimize the loss function. Deep learning networks are generally optimized with gradient descent methods and so does DCTCNN. The parameters of DCTCNN would be updated as follows:

$$\mathbf{w}_h^l(\tau+1) = \mathbf{w}_h^l(\tau) - \eta \nabla E(\mathbf{w}(\tau)), \quad \tau = 1, \ldots, T, \tag{8}$$

where $T$ and $\eta$ represent the number of training iterations and the learning rate, respectively. Each iteration $\tau$ contains the computation of the forward predicted time series $\widehat{x}$ and its corresponding error $E(\mathbf{w}(\tau))$ and the backpropagation gradient $\nabla E(\mathbf{w}(\tau))$. The derivative of each relevant weight parameter and its gradient are calculated by using the chain rule in the backpropagation process (see equation (9)). The number of training iteration is set to ensure the network convergence, which is 500 in this paper. DCTCNN is trained with Adam [33]; this optimization method can adaptively update the learning rate parameter.

At the final layer, gradient is computed as follows:

$$\frac{\partial E(\mathbf{w}(\tau))}{\partial \mathbf{w}_h^l(j,m)} = \sum_{i=1}^{N_l} \frac{\partial E(\mathbf{w}(\tau))}{\partial \mathbf{f}^l(i,h)} \frac{\partial \mathbf{f}^l(i,h)}{\partial a^l(i,h)} \frac{\partial a^l(i,h)}{\partial \mathbf{w}_h^l(j,m)}, \tag{9}$$

where $a(\cdot)$ and $\mathbf{f}$ are the activation function and output feature mapping of previous layer, respectively. Using a nonlinear activation function in each layer enables the model to learn nonlinear representations from the input monitoring time series. In this paper, rectifier linear unit (ReLU) is applied as equation (10) and the output of $l^{\text{th}}$ layer is formulized as follows:

$$\text{ReLU}(x) = \max(x, 0). \tag{10}$$

$$\begin{aligned} \mathbf{f}^l = \Big[ & \text{ReLU}\left(\mathbf{w}_{1*d}^l \mathbf{f}^{l-1} + \mathbf{b}\right), \ldots, \\ & \text{ReLU}\left(\mathbf{w}_{M_l*d}^l \mathbf{f}^{l-1} + \mathbf{b}\right) \Big], \end{aligned} \tag{11}$$

where $\mathbf{b} \in \mathbb{R}$ is the bias, $*d$ denotes the convolutional operation with dilated coefficient, and $\mathbf{f}^l \in \mathbb{R}^{1 \times N_l \times M_{l+1}}$ represents the convolution output with $\mathbf{w}_l^h$, $h = 1, \ldots, M_l$.

When the network is too "deep," the standard backpropagation becomes unstable and the training error would be increasing. This phenomenon is called performance degradation. To address this problem, residual and skip connection is added in each dilated module of DCTCNN. Connecting the input and output of each convolutional module can force the network to approximately learn residual mapping instead of the original output. Then, the residual term is passed as input to the next layer through the residual connection, ensuring not missing any useful information.

Above all, the final prediction is computed through the forward pass on the optimized DCTCNN. Considering the scale sensitivity of MAE, input monitoring time series needs to be standardized. The proposed dilated CNN module is demonstrated as follows.

Based on the module in Figure 2, the proposed DCTCNN can be established as Figure 3. In the proposed network, a fully connected dense layer and a linear activation are applied for outputting time series predictions. Dropout operation with 50% is added to avoid overfitting; it only works during the training phase. For prediction task, the inputs of DCTCNN are multivariable time series and six hyperparameters. The training and testing phases of DCTCNN are summarized in Algorithm 1.

*3.2. CNN-LSTM Hybrid Model.* The proposed generative model DCTCNN in Section 3.1 takes advantage of dilated and causal convolution mechanism to address the historical long-term dependence problem. It is based on the idea of expanding convolutional kernel receptive field. While this section combines one-dimensional CNN with LSTM to solve the long-term memory issue of time series prediction, specifically, temporal CNN works as an automatic feature extractor to obtain temporal translation-invariance features from input time series; the long-term memory capability of LSTM is applied to express the historical long-term dependence of the convolutional output. We name this generative network as CNN-LSTM hybrid model.

Suppose the $i^{\text{th}}$ input time series is

$$\mathbf{x}_i = \left[x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(t)}, \ldots, x_i^{(l)}\right], \quad \mathbf{x}_i \in \mathbb{R}^d, \tag{12}$$

where $l$ is the length of sequence, $x_i^{(t)}$ denotes the value of time step $t$, and $d$ represents the input dimension (number of input variables).

The first layer of CNN-LSTM is temporal convolution layer, whose purpose is to automatically extract short-term temporal features from the input multivariable time series. The convolutional operation can be formulized as follows:

$$c_k = \text{ReLU}\left(\mathbf{w}_k * \mathbf{x} + \mathbf{b}\right), \tag{13}$$

where $c_k$ is the output feature mapping of the $k^{\text{th}}$ convolutional filter and the corresponding weight and bias parameters are $\mathbf{w}_k$ and $\mathbf{b}$. These kernels slide from the start time step to the end for completing the convolutional operations on the entire input sequences.

The abstracted feature output is

$$c_i = \left[c_1, c_2, \ldots, c_{l-m+1}\right], \tag{14}$$

where $m$ is the convolutional filter size.

The first layer contains several kernels, which can be shown as $\{\mathbf{x}_{1:m}, \mathbf{x}_{2:m-1}, \ldots, \mathbf{x}_{l-m+1:l}\}$. After the convolutional layer, a max pooling layer is used to subsample the extracted feature mapping:

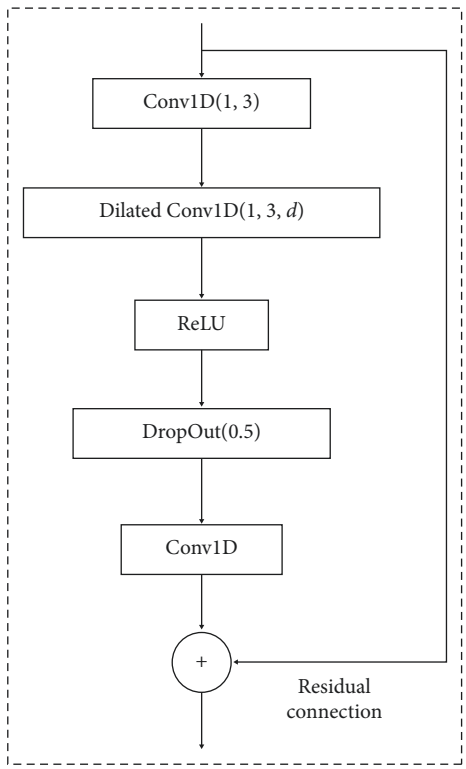


Figure 2: Dilated causal convolutional module.

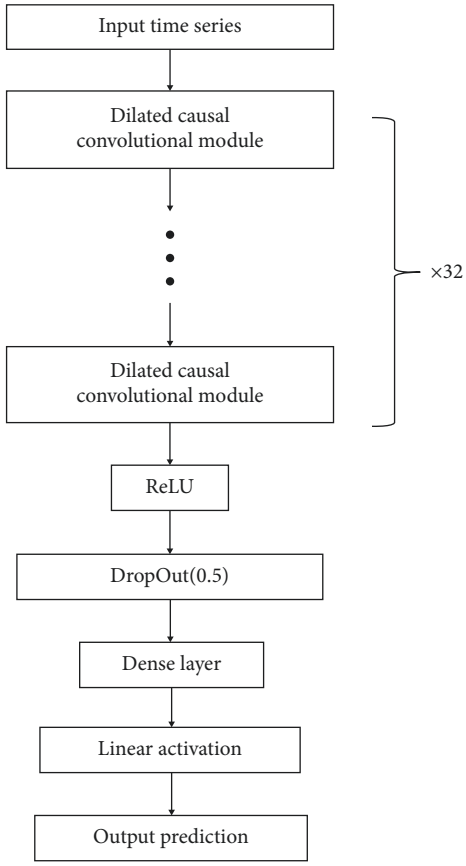


Figure 3: Deep dilated causal temporal CNN.

**Input:** Multivariable time series $[\mathbf{x}(0), \ldots, \mathbf{x}(N-1)]$ (training set and testing set)
**Output:** Predicted values $\hat{\mathbf{y}}$, trained network weights: $\mathbf{w}^*$
**Parameters:**
Kernel size: $k$
Training iterations: $M$
Training batch size: $B$
Dilated coefficient: $\mathbf{d}$
Learning rate: $\eta$
Number of stacks: $s$
Load training set and testing set from $[\mathbf{x}(0), \ldots, \mathbf{x}(N-1)]$
Randomly initialize weight $\mathbf{w}$
Begin Training
**for** [1, $M$] **do**
  Forward passing as equations (2)–(4)
  Calculate loss as equation (6)
  Calculate gradients of weights *as* equation (9)
  Backpropagation and update $\mathbf{w}$
End Training
Begin Testing: calculate predicted values $\hat{\mathbf{y}}$ with testing set
**return predicted values** $\hat{\mathbf{y}}$, $\mathbf{w}^*$

Algorithm 1: Pseudo code of DCTCNN.

$$\text{pooling} = \Big[\text{pooling}_1, \text{pooling}_2, \ldots, \text{pooling}_j, \ldots \text{pooling}_{(l-m/s)+1}\Big], \tag{15}$$

$$\text{pooling}_j = \max\left(c_{(j-1)s}, c_{(j-1)s+1}, \ldots, c_{js-1}\right), \tag{16}$$

where $s$ is the pooling size.

With the above description, the convolutional layer and the max pooling layer constitute a temporal CNN module. As shown in Figure 4, the input size is $n \times l \times d$ in which $d$ is the number of input samples. The output feature mapping size is $n \times [(l-m)/(s+1)] \times k$, where $k$ is the number of kernels. The input multivariable time series are abstracted and compressed from size $l$ to $[(l-m)/(s+1)]$. Temporal CNN module essentially acts as an automatic feature extractor.

After several modules are shown in Figure 4, the output feature mapping is sent to long short-term memory cells. The unfold LSTM on time axis shows a chain structure, which is proposed for addressing the gradient vanishing of standard RNN training. A memory state cell is added in LSTM for storing long-term historical information. In addition, three gates are designed for controlling the data-flow. Specially, forget, input, and output gates are applied to determine the insignificance information, the useful information, and the output fused information. Temporary saved information is fused with the previous memory state; the long-term historical information and the current memory are fused with above manners. Therefore, LSTM could deal with the problems of long-term historical dependence and gradient vanishing. The long short-term memory module is illustrated in Figure 5.

After LSTM, the proposed hybrid network applies a linear regression layer to compute the final prediction.

$$\hat{y}_t = \mathbf{W}_r h_t + \mathbf{b}_r, \tag{17}$$

where $\hat{y}_t$ is the prediction at the current time step, $\mathbf{W}_r \in \mathbb{R}^{k \times z}$ and $\mathbf{b}_r \in \mathbb{R}^z$ are the weight and bias parameters of linear regression, and $z$ denotes the output dimension.

In the training phase, MSE is applied to be the objective function of CNN-LSTM.

$$E(\mathbf{W}, \mathbf{b}) = \frac{1}{n} \sum_{t=1}^{n} \left(\hat{y}_t - y_t\right)^2. \tag{18}$$

Similar with Section 3.1, CNN-LSTM is also trained with Adam, and its structure diagram is shown as Figure 6. The training and testing phases of DCTCNN-LSTM are summarized in Algorithm 2.

## 4. Experimental Setup

The proposed generative temporal convolutional networks were tested on two real-life underground coal mine seismic dataset, which are shown in Table 1. UCI seismic bumps dataset [34] describes the problem of high-energy seismic bumps forecasting in a Polish coal mine Wesola. It concludes 2584 samples and 18 columns. We split this dataset into 60%, 10%, and 30% as training set, validation set, and test set, respectively. The mounted risk monitoring system in Wesola would give a seismic hazard alarm if the accumulated seismic energy is higher than $5 \times 10^4$ J. The other seismic dataset was from AAIA′16 Data Mining Challenge (Predicting Dangerous Seismic Events in Active Coal Mines [35]). The organizer Knowledge Pit offered a large volume dataset to predict increased coal mine seismic activities that endanger coal workers working underground. AAIA′16 seismic dataset was prepared by the organizer as the training set (133151 samples) and test set (3860 samples), and we split the 10% of the training set for validation.

In the experiments, our goal is to predict the total seismic energies. Therefore, we assigned the column 17 of UCI

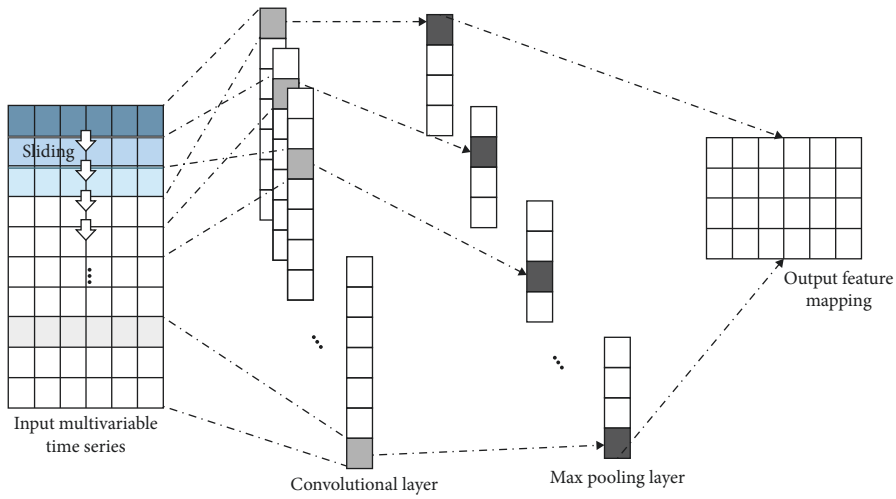


Figure 4: Temporal CNN module.

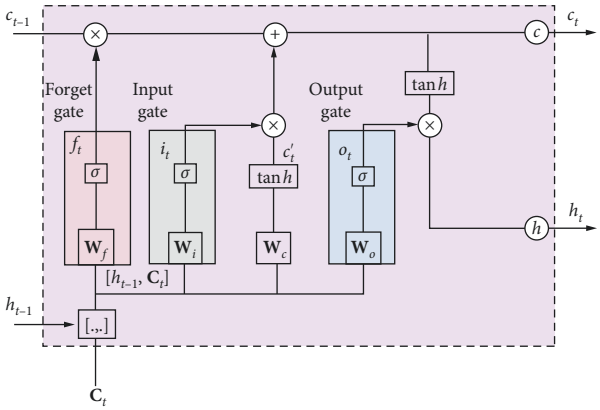


Figure 5: Long short-term memory (LSTM) module.

seismic bumps and column 5 of AAIA′16 as the label columns. Note that all these datasets indicated in Table 1 were well prepared and cleaned of malformed and erroneous values, without missing attributes. Therefore, we test DCTCNN and CNN-LSTM on these real-life datasets to perform the coal mine seismic events prediction.

In order to make the experiment more objective and fair, DCTCNN and CNN-LSTM were tested against ARIMA, support machine regression (SVR), random forest regression (RF), temporal CNN, and standard LSTM. We have split the 10% of experimental training sets for validation. For comprehensive evaluation, this work adopted 5-fold cross validation. Rooted MSE (RMSE) and mean absolute error (MAE) were used to test the performance of those methods before inverse standardization. Except standardization, the proposed networks did not commit any manual feature engineering operations. In particular, the predicted results were inversely standardized after the testing phase. The tested networks were trained with Adam in which the learning rate is 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 1e-8$. The batch size and the training iteration were fixed as 64 and 500, respectively. All the deep neural networks were implemented in the deep learning framework TensorFlow. Experiments were executed on a PC with an Intel i7-6700K 4.0 GHx processor, 32 GB RAM, and a GTX1080 GPU accelerator.

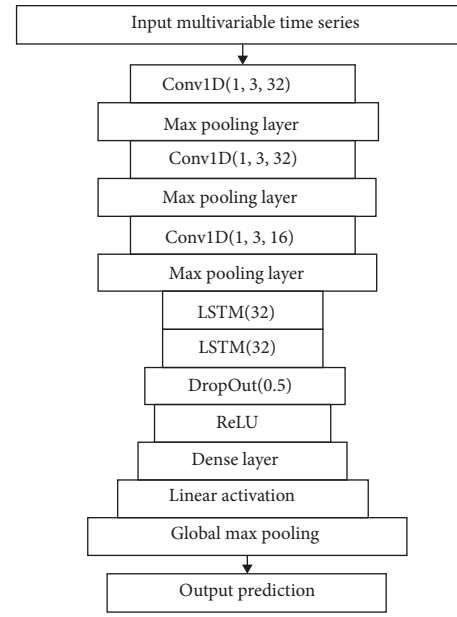


Figure 6: CNN-LSTM hybrid network.

## 5. Results and Discussion

The proposed networks were compared with the above-mentioned five methods. Their performance on underground coal mine seismic events prediction is shown in Tables 2 and 3.

Among the seven compared temporal sequence prediction methods, traditional time series prediction method ARIMA performed worst on both error metrics. For the classic machine learning methods, the RMSE scores of support vector regression and random forest regression were (0.213, 0.698) and (0.108, 0.455), respectively. Their MAE scores were (0.076, 0.586) and (0.048, 0.045). SVR and RF worked better and ran faster than the traditional time series prediction algorithm. However, compared with the deep learning algorithms, conventional machine learning algorithms still performed inferior. The results of LSTM were (0.010, 0.366) RMSE and (0.026, 0.284) MAE; the corresponding score of normal

**Input:** Multivariable time series $[\mathbf{x}(0), \ldots, \mathbf{x}(N-1)]$ (training set and testing set)
**Output:** Predicted values $\hat{\mathbf{y}}$, trained network weights: $\mathbf{w}^*$
**Parameters:**
Kernel size: $k$
Training iterations: $M$
Training batch size: $B$
Learning rate: $\eta$
1: Load training set and testing set from $[\mathbf{x}(0), \ldots, \mathbf{x}(N-1)]$
2: Randomly initialize weight $\mathbf{w}$
3: Begin Training
4: **for** [1, $M$] **do**
5: Forward passing as equations (12)–(17)
6: Calculate loss as equation (18)
7: Calculate gradients of weights
8: Backpropagation and update $\mathbf{w}$
9: End Training
10: Begin Testing: calculate predicted values $\hat{\mathbf{y}}$ with testing set
11: **return predicted values** $\hat{\mathbf{y}}$, $\mathbf{w}^*$

Algorithm 2: Pseudo code of CNN-LSTM.

Table 1: Experimental datasets.

| Datasets | Total columns | Label column | Training samples | Validation samples | Testing samples |
|---|---|---|---|---|---|
| UCI | 18 | 17 | 1629 | 180 | 775 |
| AAIA′16 | 541 | 4 | 119836 | 13315 | 3860 |

Table 2: Seven compared methods performance on UCI seismic bumps dataset.

| Category | Methods | RMSE | MAE | Running time (s) |
|---|---|---|---|---|
| Traditional time series prediction | ARIMA | 0.918 | 0.419 | 172.2 |
| Classic machine learning method | SVR | 0.213 | 0.076 | 37.2 |
| | RF | 0.108 | 0.048 | 17.8 |
| Normal recurrent network | LSTM | 0.010 | 0.026 | 46.3 |
| Normal temporal CNN network | TCNN | 0.006 | 0.009 | 26.1 |
| The proposed method | DCTCNN | $6.673 \times 10^{-4}$ | $1.031 \times 10^{-3}$ | 79.2 |
| | CNN-LSTM | $4.950 \times 10^{-3}$ | $3.667 \times 10^{-3}$ | 60.5 |

Table 3: Seven compared methods performance on AAIA′16 seismic dataset.

| Category | Methods | RMSE | MAE | Running time (s) |
|---|---|---|---|---|
| Traditional time series prediction | ARIMA | 2.038 | 1.602 | 800.3 |
| Classic machine learning method | SVR | 0.698 | 0.586 | 315.6 |
| | RF | 0.455 | 0.405 | 105.5 |
| Normal recurrent network | LSTM | 0.366 | 0.284 | 1496.2 |
| Normal temporal CNN network | TCNN | 0.198 | 0.159 | 1347.7 |
| The proposed method | DCTCNN | $6.080 \times 10^{-3}$ | **0.070** | 4729.3 |
| | CNN-LSTM | **0.011** | **0.109** | 2302.4 |

temporal recurrent network was (0.006, 0.198) and (0.009, 0.159). LSTM ran 46.3 s on UCI seismic bumps dataset and 1496.2 s on AAIA′16 seismic for 500 iterations. TCN ran 26.1 s on UCI seismic bumps dataset and 1347.7 s on AAIA′16 seismic for 500 iterations. The proposed generative networks performed better than the above algorithms. DCTCNN worked best with ($6.673 \times 10^{-4}$, $6.080 \times 10^{-3}$) RMSE and ($1.031 \times 10^{-3}$, 0.070) MAE on two datasets; its running time was 79.2 s and 4729.3 s. CNN-LSTM hybrid network got ($4.950 \times 10^{-3}$, 0.011) RMSE and ($3.667 \times 10^{-3}$, 0.109) MAE; its running time was 60.5 s and 2302.4 s. Although DCTCNN and CNN-LSTM were not the fastest models, the experimental results of the proposed models in this work were better than the above algorithms in 2~3 orders magnitudes on UCI seismic bumps and 1~2 orders magnitudes on AAIA′16 seismic. It illustrated that they are more appropriate on underground coal mine seismic events prediction tasks.

The performance of the above seven approaches are exhibited in Figures 7–13.

From the above figures, the energies of many coal mine seismic activities are higher than $5 \times 10^4$ J. These seismic events are extremely dangerous, and they need to be

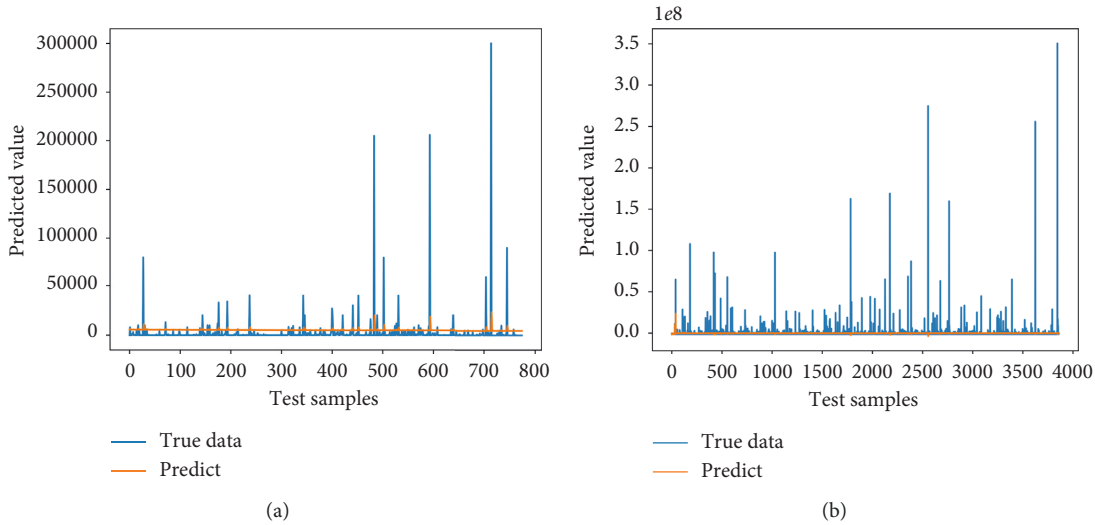


Figure 7: Testing performance of ARIMA on (a) UCI seismic bumps and (b) AAIA′16 seismic.

accurately predicted in advance. Observed from Figure 7, the traditional time series prediction method totally failed. ARIMA just fitted the data with the default hypothesis of linear characteristics. However, underground coal mine seismic prediction is a nonlinear task; it is not appropriate to predict the rock burst hazards in a linear way. Figures 8 and 9 showed that machine learning-based algorithms performed much better than ARIMA, and they can forecast several mild dangerous events which are below the alarm threshold ($5 \times 10^4$ J). For the activities which are beyond the alarm threshold, RF did better than SVR. Support vector machine is a generalized linear model which can be nonlinearized with kernel functions like radial basis function (RBF). But it might be overfitting on the training set. While random forest is a tree-based ensemble algorithm, it randomly applies a bagging of several decision trees during the training phase. Thus, RF could be trained more generalized than SVR. This is why RF could predict several hazards in Figure 9. Nonetheless, both RF and SVR did not perform well enough on coal mine seismic prediction.

Unlike classic machine learning algorithms, deep learning networks do not rely on hand-crafted feature engineering, they can commit end-to-end learning, and the high-level abstractions can be extracted through their hierarchical structures. As shown in Figures 10 and 11, LSTM and temporal CNN performed much better than RF and SVR. Under the alarm threshold level, their predictions almost cover all the mild dangerous events. Surprisingly, TCNN also forecasted several risk events and worked better than sequential algorithm LSTM. This situation endorsed the description in Section 1 and also some previous works [15–17]. TCNN could automatically extract the time-translation invariance features with one-dimensional CNN kernels. However, it did not catch the risk events whose seismic energy hit the alarm threshold. This can be explained by the lack of long-term historical memory in TCNN, which is the core issue of underground coal mine seismic prediction.

For addressing the problem of long-term historical dependence in risk prediction, this work proposed two generative temporal CNN-based networks: dilated causal temporal CNN (DCTCNN) and CNN-LSTM hybrid model. Their performance on underground coal mine seismic prediction can be seen in Figures 12 and 13. They successfully complete the prediction task, and all the seismic hazards have been accurately forecasted in advance. DCTCNN solved the long-term historical memory problem with expanded convolution receptive fields; it could obtain more historical information from the input monitoring time series. CNN-LSTM hybrid network addresses the corresponding issue by utilizing advantages of CNN and LSTM. When testing on a small dataset, it is difficult to identify which is the better model. However, DCTCNN beat CNN-LSTM on large volume seismic dataset, and CNN-LSTM spends less running time. From this perspective, the power of deep learning on large volume data is demonstrated, and it can be clearly observed in Figures 7–13.

The training processes of DCTCNN and LSTM can be seen in Figures 14 and 15. From the model training perspective, DCTCNN performed more stable and convergent than CNN-LSTM. In particular, the mechanism of DCTCNN would not cause an extra increase on parameter set. Furthermore, the application of residual connection and L2 regularization avoids information omission and model overfitting. However, CNN-LSTM has a simpler network structure, and it can be trained faster than DCTCNN in about 2 times. Furthermore, CNN-LSTM is more appropriate in long sequence situations, and DCTCNN could give a better prediction.

## 6. Conclusions

In this paper, we proposed two generative deep learning networks in order to address the long-term historical memory issue of underground coal mine seismic

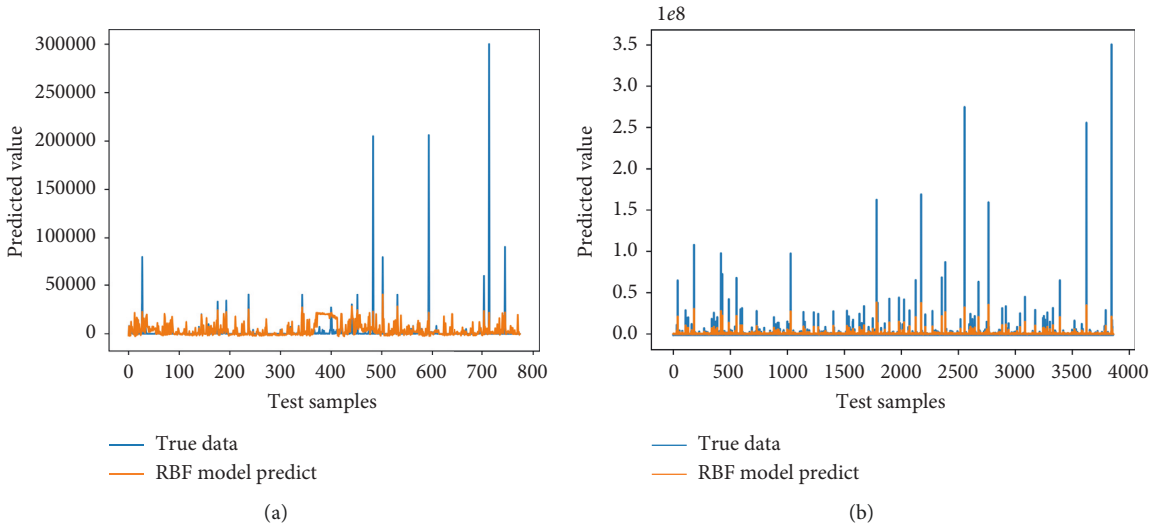


Figure 8: Testing performance of SVR on (a) UCI seismic bumps and (b) AAIA′16 seismic.

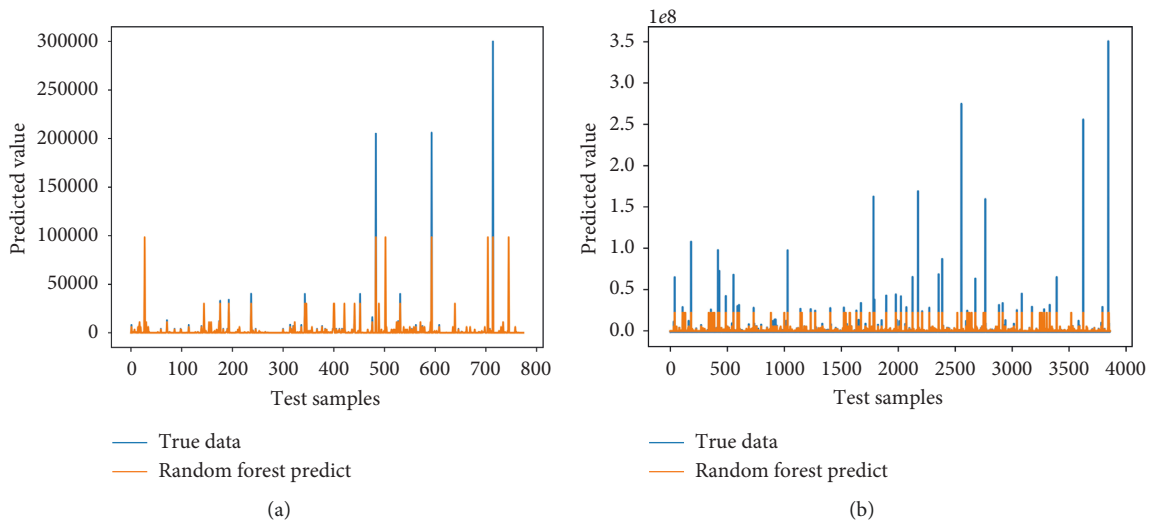


Figure 9: Testing performance of random forest regression on (a) UCI seismic bumps and (b) AAIA′16 seismic.

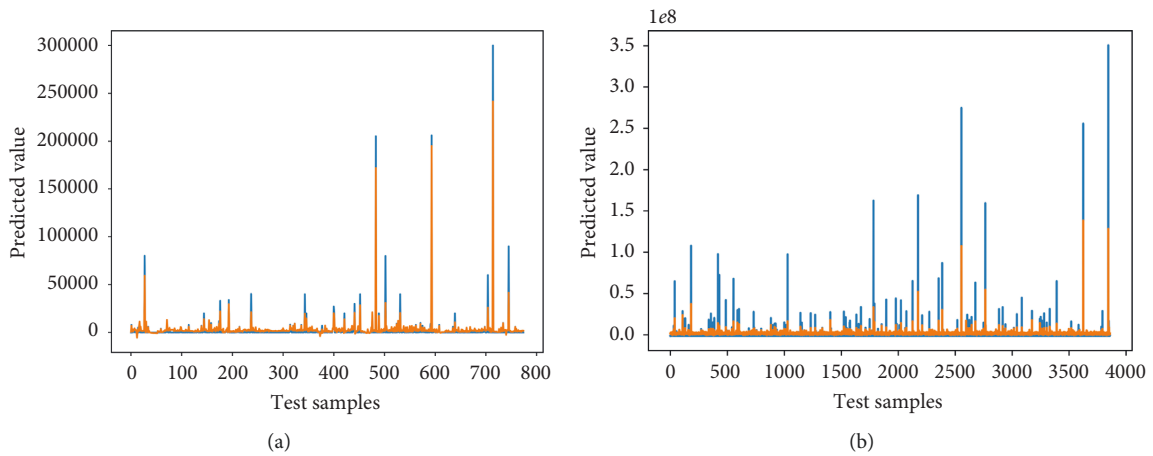


Figure 10: Testing performance of LSTM on (a) UCI seismic bumps and (b) AAIA′16 seismic.

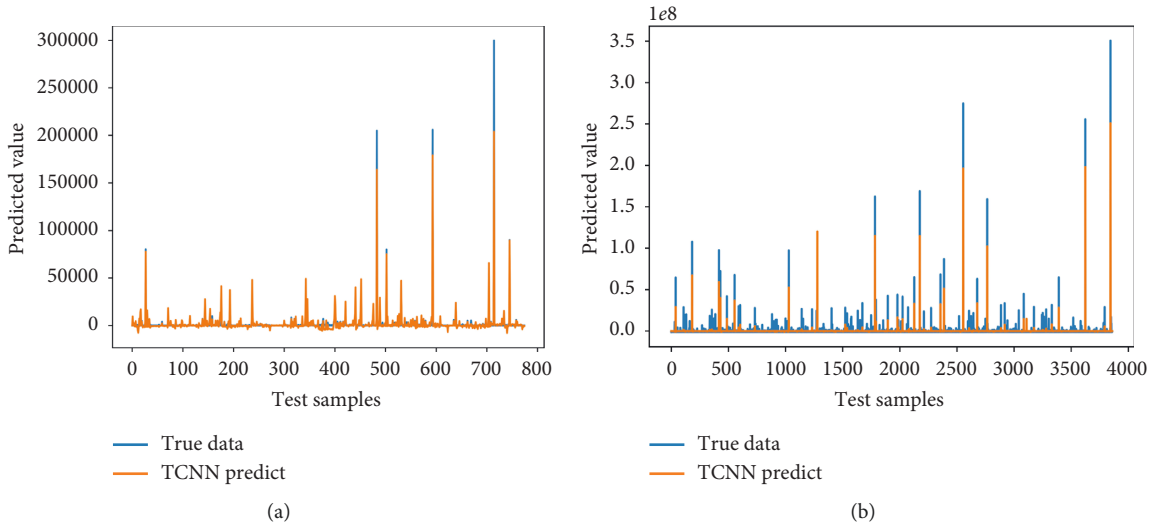


Figure 11: Testing performance of TCNN on (a) UCI seismic bumps and (b) AAIA′16 seismic.

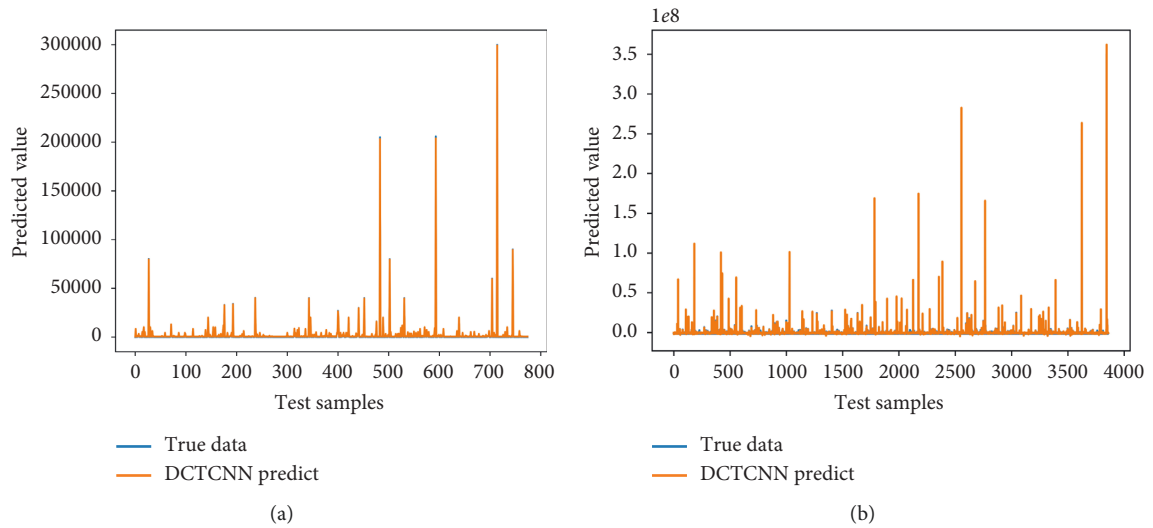


Figure 12: Testing performance of DCTCNN on (a) UCI seismic bumps and (b) AAIA′16 seismic.

prediction. Dilated casual temporal CNN (DCTCNN) was designed with dilated CNN kernels, causal strategy, and residual connection mechanism. CNN-LSTM was established in a hybrid modeling way, and it took advantage of CNN and LSTM. Tested on two real-life coal mine monitoring seismic datasets, both of the proposed models achieved convincing performance. They were also compared with traditional time series prediction method (ARIMA), classic machine learning algorithms (SVR and RF), and standard deep learning networks (TCNN and LSTM). DCTCNN and CNN-LSTM outperformed all those five algorithms. Their experimental results were better than the above algorithms in 1~3 orders magnitudes on RMSE and MAE metrics. Compared with each other, DCTCNN could be trained more stable, and CNN-LSTM runs faster. Both of the proposed models of this work can address the long-term historical dependence issue. DCTCNN worked better with $(6.673 \times 10^{-4}, 6.080 \times 10^{-3})$ RMSE and $(1.031 \times 10^{-3}, 0.070)$ MAE on two datasets. Thus, DCTCNN can forecast the underground coal mine seismic effectively. In future work, seismic events prediction with deep temporal convolutional networks can be expanded with further investigations. First, the effect of monitoring seismic dataset for the proposed networks will be analyzed, and they will be validated on larger underground coal mine seismic time series datasets. Second, the spatial information of coal mine roadway, e.g., distances between seismic monitoring points or other spatial features, will be taken into account. Third, the experimental datasets will be collected from more than one coal mine, and the proposed temporal convolutional networks will be trained in a transfer learning way.

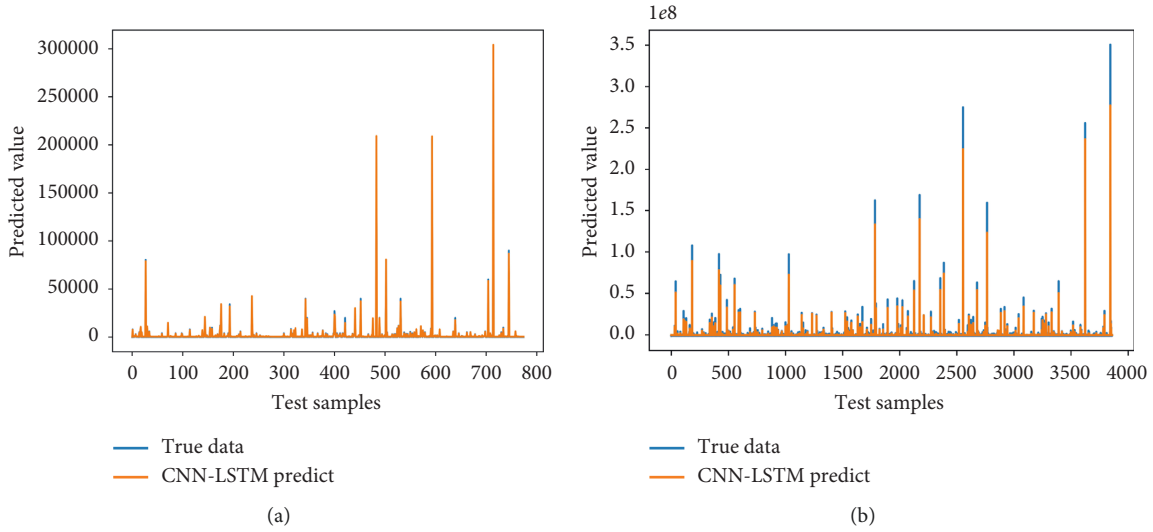


Figure 13: Testing performance of CNN-LSTM on (a) UCI seismic bumps and (b) AAIA′16 seismic.

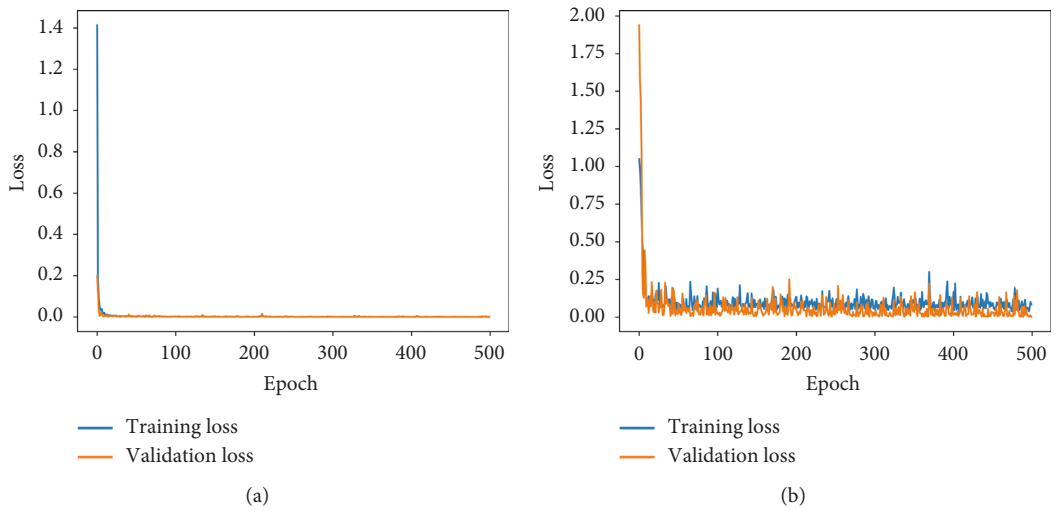


Figure 14: The training processes of the proposed models on UCI seismic bumps: (a) DCTCNN; (b) CNN-LSTM.

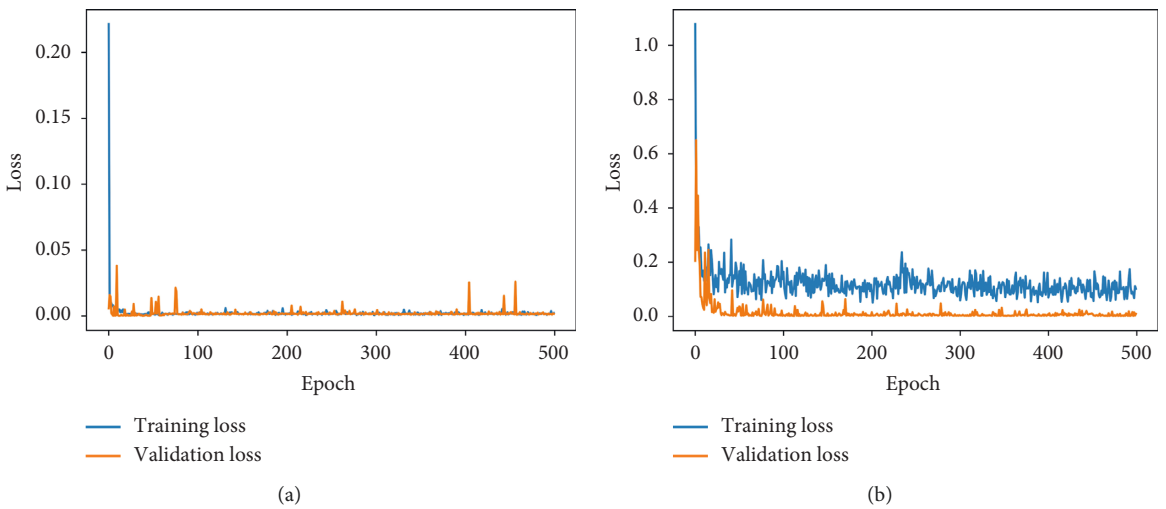


Figure 15: The Training Processes of the Proposed Models on AAIA′16 seismic: (a) DCTCNN; (b) CNN-LSTM.

## Acronyms

CNN: Convolution neural network
DCTCNN: Dilated causal temporal convolution network
LSTM: Long short-term memory network
CNN-LSTM: Convolution and long short-term memory network hybrid model
ARIMA: Autoregressive integrated moving average model
SVM: Support vector machine
SVR: Support vector regressor
RF: Random forest
RNN: Recurrent neural network
1D CNN: Convolution neural network with one-dimensional filters
2D CNN: Convolution neural network with two-dimensional filters
WaveNet: A dilated convolution neural network
ReLU: Rectifier linear unit
MAE: Mean absolute error
RMSE: Root mean squared error
RBF: Radial basis function.

## Data Availability

The seismic datasets used to support the findings of this study have been deposited in the UCI repository (https://archive.ics.uci.edu/ml/datasets/seismic-bumps) and a Polish data challenge platform (https://knowledgepit.fedcsis.org).

## Conflicts of Interest

The authors declare no conflicts of interest.

## Authors' Contributions

YG performed the experiment and wrote the paper. LS provided suggestions about the method and experiment. YJ analyzed the experimental results. CH checked the paper.

## References

[1] W. Shuwen, M. Debing, P. Junfeng et al., "Measurement on the whole process of abutment pressure evolution and microseismic activities at the lateral strata of goaf," *Journal of the China Coal Society*, vol. 40, pp. 2772–2779, 2015.

[2] L. Jinguo, J. Yaodong, Z. Yixin et al., "Hierarchical monitoring for coal bumps and its study and application of early warning methods," *Journal of the China Coal Society*, vol. 38, pp. 1161–1167, 2013.

[3] W. Meng, Y. Bochao, Z. Kaige et al., "Study on prediction of mine water inflow volume based on ARIMA product seasonal model," *Coal Science and Technology*, vol. 45, pp. 199–204, 2017.

[4] Q. Yang and X. Wu, "10 challenging problems in data mining research," *International Journal of Information Technology & Decision Making*, vol. 5, no. 4, pp. 597–604, 2006.

[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[6] Z. Jian, L. Xibing, and S. Xiuzhi, "Long-term prediction model of rockburst in underground openings using heuristic algorithms and support vector machines," *Safety Science*, vol. 50, pp. 629–644, 2012.

[7] C. Juisheng and J. P. P. Thedja, "Metaheuristic optimization within machine learning-based classification system for early warnings related to geotechnical problems," *Automation in Construction*, vol. 68, pp. 65–80, 2016.

[8] P. Yahui, P. Kang, Z. Jian et al., "Prediction of classification of rock burst risk based on genetic algorithms with SVM," *Applied Mechanics and Materials*, vol. 628, pp. 383–389, 2014.

[9] Z. Jian, L. Xibing, H. S. Mitri, S. Wang, and W. Wei, "Identification of large-scale goaf instability in underground mine using particle swarm optimization and support vector machine," *International Journal of Mining Science and Technology*, vol. 23, pp. 701–707, 2013.

[10] Z. Jian, L. Xibing, and H. S. Mitri, "Comparative performance of six supervised learning methods for the development of models of hard rock pillar stability prediction," *Natural Hazards*, vol. 79, pp. 291–316, 2015.

[11] Z. Jian, L. Xibing, M. HS et al., "Classification of rockburst in underground projects: comparison of ten supervised learning methods," *Journal of Computing in Civil Engineering*, vol. 30, no. 5, article 04016003, 2016.

[12] F. K. John and C. K. Stefan, "Gradient flow in recurrent nets: the difficulty of learning long term dependencies," in *A Field Guide to Dynamical Recurrent Networks*, S. C. Kremer and J. F. Kolen, Eds., Wiley-IEEE Press, New York, NY, USA, 2001.

[13] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proceedings of the 32nd International Conference on Machine Learning*, pp. 2342–2350, Lille, France, July 2015.

[14] J. C. B. Gamboa, "Deep learning for time-series analysis," 2017, https://arxiv.org/abs/1701.01887.

[15] B. Anastasia, B. Sander, and C. W. Oosterlee, "Conditional time series forecasting with convolutional neural networks," in *Proceedings of the 26th International Conference on Artificial Neural Networks (ICANN)*, Alghero, Italy, September 2017.

[16] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, https://arxiv.org/abs/1803.01271.

[17] A. van den Oord, S. Dieleman, H. Zen et al., "WaveNet: a generative model for raw audio," in *Proceedings of the 9th ISCA Speech Synthesis Workshop*, Sunnyvale, CA, USA, September 2016.

[18] W. He, "Load forecasting via deep neural networks," *Procedia Computer Science*, vol. 122, pp. 308–314, 2017.

[19] Z. Rui, Y. Ruqiang, W. Jinjiang et al., "Learning to monitor machine health with convolutional Bi-directional LSTM networks," *Sensors*, vol. 17, pp. 273–291, 2017.

[20] A. Haytham, G. Salem, M. Gabor et al., "Urban water flow and water level prediction based on deep learning," in *Proceedings of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, Skopje, Macedonia, September 2017.

[21] J. Zhou, X. Li, and H. S. Mitri, "Evaluation method of rockburst: state-of-the-art literature review," *Tunnelling and Underground Space Technology*, vol. 81, pp. 632–659, 2018.

[22] T. Hui, M. Xiaoping, and Q. Meiying, "Rock burst prediction on multivariate chaotic time series," *Journal of China Coal Society*, vol. 37, pp. 1624–1629, 2012.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] N. Laptev, J. Yosinski, L. E. Li et al., "Time-series extreme event forecasting with neural networks at uber," in *Proceedings of the ICML 2017 Time Series Workshop*, Sydney Australia, August 2017.

[25] Z. Lingxue and L. Nikolay, "Deep and confident prediction for time series at uber," in *Proceedings of the 2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, New Orleans, LA, USA, November 2017.

[26] V. Flunkert, D. Salinas, and J. G. DeepAR, "Probabilistic forecasting with autoregressive recurrent networks," 2017, https://arxiv.org/abs/1704.04110.

[27] F. Fernández-Navarro, M. A. de la Cruz, P. A. Gutiérrez, A. Castaño, and C. Hervás-Martínez, "Time series forecasting by recurrent product unit neural networks," *Neural Computing and Applications*, vol. 29, pp. 779–791, 2018.

[28] M. Alberto, O. Bruno, and G.-C. Sandra, "Forecasting short-term data center network traffic load with convolutional neural networks," *PLoS One*, vol. 13, no. 2, Article ID e0191939, 2018.

[29] D. Yunxuan, W. Jianzhou, and Z. Guo, "Research and application of local perceptron neural network in highway rectifier for time series forecasting," *Applied Soft Computing*, vol. 64, pp. 656–673, 2018.

[30] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," 2016, https://arxiv.org/abs/1612.01022.

[31] Z. Rui, Y. Ruqiang, and W. Jinjiang, "Learning to monitor machine health with convolutional bi-directional LSTM networks," *Sensors*, vol. 17, pp. 1–18, 2017.

[32] G. Lai, W.-C. Chang, Y. Yang et al., "Modeling long-and short-term temporal patterns with deep neural networks," 2017, https://arxiv.org/abs/1703.07015.

[33] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, May 2015.

[34] S. Marek and W. Lukasz, "Application of rule induction algorithms for analysis of data collected by seismic hazard monitoring systems in coal mines," *Archives of Ming Sciences*, vol. 55, pp. 91–114, 2010.

[35] M. Boullé, "Predicting dangerous seismic events in coal mines under distribution drift," in *Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 221–224, Gdansk, Poland, September 2018.