A Bridge Structure 3D Representation for Deep Neural Network and Its Application in Frequency Estimation

Kejian Hu and Xiaoguang Wu

Highway School, Chang'an University, Xi'an, China

Correspondence should be addressed to Xiaoguang Wu; wxgwst.cn@126.com

Received 15 October 2021; Revised 12 January 2022; Accepted 7 March 2022; Published 22 March 2022

Academic Editor: Iman Mansouri

Copyright © 2022 Kejian Hu and Xiaoguang Wu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Currently, most predictions related to bridge geometry use shallow neural networks, which limit the network’s ability to fit since the input form limits the depth of the neural network. Therefore, this study proposed a new 3D representation of bridge structures. Based on the geometric parameters of the bridge structure, three 4D tensors were formed. This form of representation not only retained all geometric information but also expressed the spatial relationship of the structure. Then, this study constructed the corresponding 3D convolutional neural network and used it to estimate the frequency of the bridge. In addition, this study also developed a traditional shallow neural network for comparison. The application of 3D representation and 3D convolution could effectively reduce the prediction error. The 3D representation presented in this study could be used not only for frequency prediction but also for any prediction problems related to bridge geometry.

1. Introduction

In recent years, deep learning (DL), as a class of machine learning algorithm, has gained rapid development and demonstrated tremendous success in various fields, such as computer vision [1], speech recognition [2], and natural language processing [3]. In civil engineering, it has been used in concrete crack detected, construction site management, structural health monitoring (SHM), and damage detected [4–7]. However, at present most studies use convolutional neural networks (CNNs) or long short-term memory (LSTM) as feature extractor for classification/recognition, but less to perform regression or function approximation problems. In terms of bridge structure prediction, most of the traditional shallow neural networks are used. Nguyen et al. [8] applied a hybrid PSO-based ANN model to the calculation of horizontal deflection of columns in short building after being subjected to a significant seismic loading. The input was some ground motion description parameters, such as dynamic, friction angle ($\phi$), and dilation angle ($\psi$), which all are manually selected. Asteris and Nikoo [9] built an artificial bee colony-based neural network with 2 hidden layers to predict the fundamental period of infilled frame structures using the number of storeys, the number of spans, the span length, the infill wall panel stiffness, and the percentage of openings within the infill panel. An integrated multiobjective harmony search with shallow artificial neural networks was proposed by García-Segura et al. [10] to find optimum PSC box-girder cross-section bridge design parameters which minimize the cost and maximize the overall safety factor and the corrosion initiation time. Chatterjee et al. [11] proposed a model based on neural network-particle swarm optimization (NN-PSO) to predict the structural failure of a multistoried reinforced concrete building.

The reason that the traditional shallow neural network is used in the prediction of bridge structure is that the input form limits the depth of the neural network. The outstanding advantage of deep learning is that it can model high-level abstractions from input data by stacking multiple nonlinear layers. Each layer corresponds to different levels of features so that deep network has a higher characterization ability. However, for the above shallow neural network, the inputs are selected manually, usually some structural parameters
that have a great influence on the results. There are two main problems with this way. First, the number of features, that is, the number of inputs to the network, is too few. This leads to overfitting when there are too many stacked layers in the network. Second, the selected features do not necessarily contain all the information of the structure. Some important information may be discarded, while some useless information is used. As a result, the network cannot acquire and learn enough features. When dealing with simple problems, it has less impact, but when the problems become complex, it will affect the final prediction accuracy. Besides, it is not reasonable to use some discrete parameters to represent structures. As civil structures are 3D in real world, their spatial structure information is very important, and it is difficult to express it with one-dimensional discrete parameters.

Nevertheless, there are still some studies that apply deep learning to the prediction of structural response. Zhang et al. [12] developed a physics-guided convolutional neural network (PhyCNN) to predict structural response. The network took ground acceleration-time sequence $\mathbf{x}$ as input and output the state space variables $\mathbf{z}(t)$ including displacement $\mathbf{x}(t)$, velocity $\mathbf{v}(t)$, and restoring force $\mathbf{g}(t)$. Zhang et al. [13] established a convolutional long short-term memory (ConvLSTM) network to predict the bridge’s strain response. The inputs of the network were strain measurements over three years of a sensor grid, and the output were the corresponding future strain records. By utilizing the advantage of long short-term memory (LSTM) network for time-series prediction, a deep learning model in minute scale based on monitoring data was developed by Yue et al. [14] to predict temperature-induced deflection.

In these studies, time sequences are used as input to predict the dynamic response of the structure. These time series can be divided into two types, ground motion [12] and structural response [13, 14], such as displacement-time sequence, velocity-displacement-time sequence, acceleration-time sequence, or frequency-time sequence. The above studies have proved that these two approaches are feasible, but there are still some drawbacks. First, for the first type of input, it can only be used for the prediction of dynamics-related problems. Because all the features are extracted from the ground motion which does not include anything information about the structure, therefore, when applied to statics or instability problems, it does not provide any useful information. For the same reason, when the parameters of the structure change, the original trained network becomes inapplicable. It is because the parameters associated with the structure are implicit in the weight and paranoia of each neuron rather than input. In the second type, the input comes from the historical data measured by sensors. Therefore, this method can only be used for the prediction of structures already built or under construction and cannot be used for structures under design.

The above two ways attempt to let the deep network learn the structural features in an indirect way, but both still have limitations. Therefore, it is necessary to develop a form to express the structure directly and effectively. This representation should contain enough structural information as well as spatial characteristics. At present, there have been many research studies on the representation of 3D objects, such as Point Cloud, Mesh, Multiview, and Voxel. Point cloud is a collection of points distributed in 3D space (mathematically expressed as an $N \times 3$ matrix, where $N$ is the number of points), which belongs to nonstructural 3D model data. PointNet proposed by Qi et al. [15] is the first neural network model that directly use point cloud data for 3D model classification, and its accuracy on ModelNet40 test set reached 89.2%. Later, Qi et al. [16] added the extraction of the local features of the point cloud based on the original model and proposed PointNet $+$ $+$, which improved the accuracy to 91.9%. However, its point features were abstracted into independent and isolated ways, ignoring the relative layout of adjacent points and their features.

A mesh is a structured form of data formed through point cloud. In addition to the coordinates of points, it also contains the connection relationship and various attribute information of points, lines, and planes, which makes it have a stronger 3D shape description ability. However, because of the complexity of grid data, there are few research studies and application on feature extraction of mesh data in recent years. Feng et al. [17] proposed a mesh-based deep learning model, MeshNet, which can better solve the complex grid problems, and the classification accuracy on ModelNet40 reached 91.9% and the average retrieval accuracy reached 81.9%.

The multiview method converts the 3D model into several 2D views by means of multiview projection. And 2D view has been widely applied in deep learning and is very mature in technology. Therefore, typical CNNs can be directly transplanted to learn and extract 3D shape features based on multiple views. Su et al. [18] first presented a CNN architecture named multiview convolutional neural networks (MVCNN) trained to recognize the shapes’ rendered views independently of each other, and the classification accuracy of this model on ModelNet40 reached 90.1%, and the retrieval accuracy reached 80.2%. In this way of thinking, Jiang et al. [19] proposed a multilayer-view convolutional neural network (MLVCNN) which took multigroups of views extracted from different loop directions as input for 3D retrieval. Its retrieval accuracy on ModelNet40 test set reached 92.84%. However, this approach has a fatal drawback. Because the inner void is not visible in the projected view, it is not suitable for representing the object with inner void.

Voxels are the smallest units that make up a 3D model, like pixels and images. It is the smallest unit that makes up the 3D model. The tensor obtained by voxelization of the 3D model can be directly used in 3D convolutional network to extract features. Wu et al. [20] first proposed a 3D ShapeNets for voxel depth representation of 3D models in 2015 and achieved a classification accuracy of 77% in ModelNet40. To solve the problems of traditional 3D-CNNs such as computationally expensive, memory intensive, prone to overfit, and improve feature learning capabilities, Kumawat and Raman [21] used a Rectified Local Phase Volume (ReLPV) block replacing traditional convolution layers and proposed LP-3DCNN models. The accuracy of the network on
ModelNet10 and ModelNet40 reached 92.1% and 94.4%, with fewer parameters.

In this study, a new 3D representation of the bridge structure is proposed, and a 3D-CNN is constructed to fulfill the prediction task. This representation retains the original information and three-dimensional characteristics of the structure as much as possible, and its performance is also verified by the results of the prediction task. To the best of our knowledge, this study presents probably the first attempt to express bridge in this way and to employ 3D-CNN for bridge prediction task. Such an approach is fundamental and can be applied to other prediction problems.

2. Methods

2.1. 3D Representation for Bridge. Point clouds are the most universal 3D representation. Due to the need to express a variety of different shapes of objects, the spatial relationship between each sampling point is not fixed. This forced the point clouds to adopt \((3 \times n)\) matrix, which abandoned the spatial information of the sampling points. However, when it comes to bridge structures, the situation is different. For a specific bridge type, its shape is roughly the same; only in the design parameters are different. They can all be represented by the same kind of mechanical diagram, such as simply supported beam, continuous beam, and arch. In addition, the shape of the bridge is generally composed of regular geometry, which can be described with fewer sampling points. And the choice of the locations of these points is clear and fixed. All these reasons make it possible to represent the bridge structure with a structured 3D data form. In this study, the 4-dimensional tensor was adopted.

At present, 4-dimensional tensor is mainly used to represent video data \([22]\), medical CT, and MRI data \([23]\) and used to represent the set of multiple feature maps in neural network. Its shape can be expressed as \((A, B, C, D)\). When representing video data, A usually represents the selected key frame, B and C represent image’s height and width of each frame, and D is the channel of image. For color images, D is equal to 3, which corresponds to RGB. Inspired by this, the bridge’s 3D represent can be formed by the same way. The first dimension of the 4D tensor was used to represent the key cross section, the second and third dimensions represented the distribution of sampling points on the cross section, and the last one was the X, Y, and Z coordinate values of the sampling points.

The process of transforming the bridge structure into a 4D tensor is explained below, taking a 3-span continuous rigid frame bridge as an example.

2.1.1. Disassemble the Bridge into Smaller Part with the Same Shape. Because the cross sections of the beam and pier are too different to be represented by the same tensor, the first step was to disassemble the whole structure into three parts, the beam and two piers. Although the shapes of the two piers are similar, their coordinate values were independent of each other and therefore were treated separately.

2.1.2. Select Cross Section. Just as in video data, the key frames were used to describe the movement of objects; the key cross sections were used to describe the change of bridge shape. The beam of most continuous rigid frame bridge is the variable section. The beam is highest at the top of the pier and lowest at the side span fulcrum and midspan. So, according to the variation rule of beam height, the beam was divided into the following 4 types: constant height segment in side span, varying height segment in side span, constant height segment on pier, and varying height segment in midspan, a total of 8 segments, as shown in Figure 1. It was obvious from the figure that the cross sections connecting adjacent segments determine the shape of the whole beam. These 11 sections were the key cross sections. However, the key cross sections alone may not be enough to explain how the cross section changes. This problem can be solved by interpolating between the key cross sections. These interpolation sections were evenly distributed in the 2 constant height segments in side span, the 2 varying segments in side span, and the 2 varying height segments in midspan according to the ratio of 1:2:2:2:1:1, as shown in Figure 1.

Compared with girder, it was much easier to deal with the pier cross section. Because most conventional piers adopt simple columns, no matter solid piers, hollow piers, or double-limb thin-walled piers, there was no abrupt change in pier contour. Therefore, only need to select the section at both ends of the pier and insert a plurality of sections between them.

The VS means varying height segment and the CS means constant height segment. s means the segment is in the side span, and m means that the segment is in the middle span. For example, VS-s means the segment is a varying height segment in the side span. The red color means it is a girder section and the blue color means it is a pier section. The solid line means it is a key section, and the dashed line means it is an interpolated ordinary section.

2.1.3. Select Key Points. The main girder of a continuous rigid frame bridge is generally a box girder, as shown in Figure 2(a). There was no doubt that vertices in the cross section must be picked and marked with solid triangles. These were the first kind of points. From these points, horizontal and vertical lines were drawn to form a grid. The newly created intersection on the grid was the second kind of points. When these points fall into the section, they were marked with a solid circle; otherwise, they were marked with a hollow circle. These two types of points formed the basic frame, while more points were needed to enrich the information of cross section, such as selecting cross sections. These points were the third type of points, which can be obtained through refining grid. Such points were marked by squares in the diagram. Again, solid means the point is in the plane and hollow means out of the plane. The pier section was the same. Figure 2 is the schematic diagram of key points’ selection of bridge pier and beam section. In this figure, a key point was interpolated between every two points of the first or second type.
2.1.4. Calculate the Coordinates of Key Points.
Calculating the key points was the last and most important step in forming the tensor. First, determine a global coordinate system. The coordinate origin can be in any position, but for ease of calculation, it was recommended to be in the symmetry axis of the structure. In this study, the longitudinal direction of the bridge was determined as the $Z$-axis, the transverse direction as the $X$-axis, and the vertical direction as the $Y$-axis.

$L_+$ hen, select a cross section to determine the coordinate value of the key points on it. The coordinates of the points marked with hollows were determined first and were all set to 0, regardless of $x$, $y$, and $z$. Because these points do not really exist, things that do not exist are typically set to zeros in deep learning so that the network will not learn any features. After that, the coordinates of the first kind of points needed to be determined, which could be determined by the structural parameters of the bridge. For the main girder section, since all the key points were in the same $X$–$Y$ plane, their $Z$ coordinates were the same value. Similarly, for the pier section, the $Y$ coordinates were the same. For the second kind of points, the coordinates could be obtained by combining the first kind of points’ coordinates. And then, there was the third type of section. Because it comes from interpolation, their coordinates could be determined by two adjacent points on the same line.

After the above steps, the 3D representation of the continuous rigid frame bridge was obtained. In this tensor, any two adjacent points were also adjacent in the actual bridge, so the spatial information of the structure could be retained. Figure 3 is a schematic diagram of the 4-dimensional tensors. To visualize the coordinate change of each section, the entire tensor is scaled to between 0 and 1 to represent the RGB color. Taking the beam section as an example, the farther the point on the flange plate of the beam is from the center, the greater the coordinate $X$ value will be and the redder it is in the figure. As the height of the beam increases, the farther the point on the bottom plate of the beam is from the top surface, the more inclined to green it is in the figure. In the direction of beam length, the farther away from the origin it is, the bluer the whole picture will be, especially in the center of the picture. However, it should be noted that the figure is only for illustration. Since the length of the actual bridge is much longer than the other two dimensions, the color of the cross section is mostly a single blue, not colorful.

![Figure 1: Schematic diagram of main beam and pier section selection.](image1)

![Figure 2: Key points’ selection of bridge pier and beam section. (a) Box girder. (b) Hollow pier.](image2)
2.2. 3D Convolution. After the 3D representation of the bridge was obtained, the corresponding neural network was needed to deal with this data structure. 3D convolution neural network is the most commonly used neural network for processing 4D tensor [16, 17, 20, 21, 23]. Usually, a normal 2D-CNN is composed of the convolution layer and the pooling layer. In the convolution layer, multiple convolution kernels (filters) sweep the entire input and convert it into a feature map as output. Then, in the pooling layer, the output feature map is compressed to simplify the model complexity and extract the main features. After several convolution layers and pooled layers, the feature map is flattened into a feature vector, which is input to the fully connected layer and finally outputs the predicted value. The same is true for 3D-CNN, except that the 2D filters are replaced by 3D and the 3D pooling layer is used.

In 2D-CNNs, convolutions are applied on the 2D input or feature maps in previous layers to extract features. Then, the result is added with a bias and passed through an activation function to produce the feature maps of this layer. This process can be formulated as

$$v_{ij}^{xy} = \sigma \left( b_{ij} + \sum_{m} \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} \sum_{r=0}^{R-1} w_{ijm}^{pqr} v_{(i-1)m}^{p(x+p)(y+q)(r+r)} \right),$$

where $v_{ij}^{xy}$ is the value of an unit at position $(x, y)$ in the $i$th feature map in the $j$th layer, $\sigma$ is the activation function, $b_{ij}$ is the bias for this feature map, $m$ indexes over the set of feature maps in the $(i-1)$ layer connected to the current feature map, $w_{ijm}^{pqr}$ is the value at the position $(p, q)$ of the kernel connected to the $m$th feature map, and $P_i$ and $Q_i$ are the height and width of the kernel, respectively.

From the above description, 2D-CNNs can only extract features from two dimensions, $x$ and $y$. However, for 3D structure, features from three dimensions are desirable, which can be achieved by 3D convolution. It applies a 3D kernel on a cube from the 4D tensor, thereby capturing the 3D spatial information. Similarly, the process can be represented by the following formula:

$$v_{ij}^{xyz} = \sigma \left( b_{ij} + \sum_{m} \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} \sum_{r=0}^{R-1} w_{ijm}^{pqr} v_{(i-1)m}^{p(x+p)(y+q)(z+r)} \right),$$

where $R_i$ is the size of the 3D kernel along the third dimension, $w_{ijm}^{pqr}$ is the value of the kernel connected to the $m$th feature map in the previous layer. A comparison of 2D and 3D convolutions is illustrated in Figure 4.

3. Example Application

In this section, the 3D representation of bridges mentioned above was applied to predict the frequencies of bridges. The frequency is an important parameter reflecting the dynamic characteristics of the structure, especially in the seismic design of the bridge [24].

The frequency of bridges can be obtained by solving the characteristic equation of the damped multiple degree of freedom (MDOF) system:

$$[-M + \omega_n^2 K] \psi = 0,$$

where $M$ is the mass matrix, $K$ is the stiffness matrix, and $\omega_n$ is the natural frequency. It can be seen from (3) that the frequency is determined by the mass matrix and the stiffness matrix. And, the mass matrix is determined by the geometric shape of the bridge and the material density, while the stiffness matrix is also affected by the geometric shape, as well as the elastic modulus and boundary conditions of the material. Therefore, the 3D shape of the bridge has a great influence on its frequency, and the frequency prediction can well verify the performance of the abovementioned 3D representation.

3.1. Dataset and Preprocessing. Since deep learning is dedicated to mining the implicit rules of datasets from a large amount of data, it requires the dataset to be large, high quality, and diverse. On-site testing can undoubtedly obtain accurate bridge frequencies, but this type of testing often requires a lot of time and expense. Finite element analysis is now widely recognized as a powerful and versatile analytical tool with which to estimate the dynamic characteristics of
bridge structures efficiently [24]. Therefore, all the bridge frequencies in this study were obtained by the finite element method. The finite element model is established by using ANSYS based on 18 parameters of material and geometry, listed in Table 1, as well as their ranges. These parameters’ ranges were referred to some real bridges and design practical experience [25, 26]. All the 3D finite element models used SOLID185 for the concrete girder and pier. As for boundary conditions, fixed constraints were applied to both sides of the pile caps, and the vertical and horizontal directions were constrained at both ends of the box girder. As for the unit size, 4 different sizes, 0.25 m, 0.5 m, 1.0 m, and 2.0 m, were tested on the same bridge. The finite element model of the bridge is shown in Figure 5, and the parameter values are listed in Table 1. It can be seen from results that when the cell size is 0.25 and 0.5, the difference is not large. So, the unit size was set as 0.5 m, considering spend time and precision [27, 28]. Li studied the effect of prestressing on the vibration frequency of bridges. In his study, he found that prestressing does reduce the fundamental frequency of the structure, but the increase of prestress from 0 to 1 times the design prestress produces very little change in the fundamental frequency of the structure, so the effect of prestressing can be ignored. Similarly, the effect of common steel reinforcement is neglected in this paper [29]. The first 5 mode shapes and frequencies were computed in this study, which were the most used in actual engineering.

After the parameter set was obtained, the corresponding 3D representation of the bridge could be formed according to the method described in Section 2.1. In order to analyze the influence of tensor size on prediction accuracy, this study introduced the cross-section interpolation number \( N_s \) and the key point interpolation number \( N_p \) to control the size of the input tensor. The cross-section interpolation number \( N_s \) was the number of sections added in addition to the basic 11 key sections, distributed in a ratio of 1 : 2 : 2 : 2 : 1 across the beam, as described in Section 2.1.2. The number of pier sections is consistent with the number of sections. The key point interpolation number \( N_p \) was the number of key points interpolated for every two points of the first or second type in each section, as described in Section 2.1.3. For the piers, use the same method. The points on the cross section were also obtained by interpolation. Therefore, the shape of the input tensor can be expressed as

\[
\text{Girder Tensor shape} = (11 + N_s, 4 + N_p \times 3, 6 + N_p \times 5, 3).
\]

\[
\text{Pier Tensor shape} = (11 + N_s, 4 + N_p \times 3, 4 + N_p \times 3, 3).
\]  

(4)

In order to ensure that the interpolated section of each segment is an integer, the section interpolation number must be a multiple of 10. In this study, the cross-section interpolation number \( N_s \) was taken as 0, 10, and 20, where 0 means no interpolation was performed. And the key point interpolation number \( N_p \) was 0, 1, 3, and 5.

The parameter set, 3D representations, and the first five frequencies of each bridge made up the entire dataset for the neural network. The size of dataset was 8626, including 5331 training set (around 60% of total data), 1777 validation set (around 20% of total data), and 1778 test set (around 20% of total data).

Before training, it is important to standardize the data. It makes the features of different dimensions in the same numerical order of magnitude, improves accuracy by reducing the influence of features with large variance, and accelerates the convergence speed of the learning algorithm. In this study, each 3D representation tensor was divided by the element with the biggest value in the tensor so that the values of each tensor are distributed between 0-1, thus achieving standardization. For the 18 parameters, the same processing method was also used.

3.2. 3D Convolution Neural Network. Based on the 3D convolution described on Section 2.2, various CNN architectures can be constructed. In this section, the 3D-CNN architecture developed for predicting the frequency of bridges is described. In addition to the basic input layer and output layer, the entire architecture also includes 4 stages, 1 flatten part, 1 concatenate part, and 1 fully connected (FC) part, as shown in Table 2 and Figure 6. Wherein, the stage is a collection of convolutional layers with the same out channel. Each stage contains 7 convolutional layers with a convolution kernel of (1, 1, 1). The stride of the first 6

![Figure 4: Comparison of (a) 2D convolution and (b) 3D convolution.](image-url)
convolutional layers is 1, while the last is 2. This is because the last layer also needs to reduce the volume of the tensor. These 4 stages are stacked in sequence, and as the network depth increases, the out channel gradually increases from 32 to 512. Immediately, after each layer of 3D convolutional layer, there is a layer of the batch normalization layer and the activation layer. The batch normalization layer makes the data into a distribution that satisfies or approximates Gaussian form by normalization, scaling, and translation. Thus, the training speed and accuracy of the neural network are accelerated. The last layer of the stage is a dropout layer, which prevents the neural

Table 1: Geometric and material parameters.

| Parameters                        | Range            | Values of the sample bridge | Unit |
|-----------------------------------|------------------|-----------------------------|------|
| Middle span length                | [100, 350]       | 209                         | m    |
| Side span length                  | [50, 280]        | 160                         | m    |
| Pier height                       | [10, 200]        | 161.91                      | m    |
| Pier width                        | [5, 10]          | 9.5                         | m    |
| Pier wall thickness               | [10, 500]        | 220                         | cm   |
| Girder height on pier             | [500, 2500]      | 1400                        | cm   |
| Web thickness on pier             | [50, 100]        | 99                          | cm   |
| Top slab thickness on pier        | [50, 70]         | 56                          | cm   |
| Bottom slab thickness on pier     | [25, 250]        | 159                         | cm   |
| Girder height on middle           | [200, 900]       | 500                         | cm   |
| Web thickness on middle           | [30, 50]         | 47                          | cm   |
| Top slab thickness on middle      | [25, 35]         | 28                          | cm   |
| Bottom slab thickness on middle   | [25, 35]         | 27                          | cm   |
| Girder curves’ power              | [1, 2]           | 1.5                         |      |
| Top slab width                    | [1000, 3000]     | 2100                        | cm   |
| Flange slab width                 | [20, 1000]       | 65.5                        | cm   |
| Concrete density                  | [1000, 3000]     | 1370                        | kg/m³|
| Concrete elasticity modulus       | [20000, 40000]   | 21500                       | Mpa  |

Figure 5: Finite element model and frequency result. (a) Finite element model. (b) First-order mode shape ($f = 0.258$ Hz). (c) Second-order mode shape ($f = 0.417$ Hz). (d) Third-order mode shape ($f = 0.655$ Hz). (e) Fourth-order mode shape ($f = 0.851$ Hz). (f) Fifth-order mode shape ($f = 1.129$ Hz).
network from overfitting by disabling some neurons randomly during training.

After the girder tensor and the 2 pier tensors enter the network from the input layer, they flow through 4 stages in turn and are converted into one-dimensional vectors in the flatten part. In the concatenate part, these 3 one-dimensional vectors are combined with 18 design parameters and sent to the fully connected part. The fully connected part is composed of 5 parallel dense layers, corresponding to the first 5 frequencies. Each dense layer includes 256 neurons. Finally, the vector flowing from the dense layer passes through the ReLU activation function to form the final output, which is the first 5 order frequencies.

Two commonly used metrics were employed to quantitatively evaluate the performance of the proposed architecture, including mean absolute error (MAE) and mean absolute percentage error (MAPE), and the MAE was the loss function of the network. They were defined as follows:

### Table 2: 3D-CNN configuration.

| Input layer | Girder’s tensor | 1st pier’s tensor | 2nd pier’s tensor | 18 design parameters |
|-------------|----------------|-----------------|-----------------|---------------------|
| Stage 1     | 6 × Conv (1,1,32) | 6 × Conv (1,1,32) | 6 × Conv (1,1,32) | —                   |
|             | Conv (1,2,32)    | Conv (1,2,32)   | Conv (1,2,32)   |                     |
|             | Dropout (0.3)    | Dropout (0.3)   | Dropout (0.3)   |                     |
| Stage 2     | 6 × Conv (1,1,64) | 6 × Conv (1,1,64) | 6 × Conv (1,1,64) | —                   |
|             | Conv (1,2,64)    | Conv (1,2,64)   | Conv (1,2,64)   |                     |
|             | Dropout (0.3)    | Dropout (0.3)   | Dropout (0.3)   |                     |
| Stage 3     | 6 × Conv (1,1,128) | 6 × Conv (1,1,128) | 6 × Conv (1,1,128) | —                   |
|             | Conv (1,2,128)   | Conv (1,2,128)  | Conv (1,2,128)  |                     |
|             | Dropout (0.3)    | Dropout (0.3)   | Dropout (0.3)   |                     |
| Stage 4     | 6 × Conv (1,1,256) | 6 × Conv (1,1,256) | 6 × Conv (1,1,256) | —                   |
|             | Conv (1,2,256)   | Conv (1,2,256)  | Conv (1,2,256)  |                     |
|             | Dropout (0.3)    | Dropout (0.3)   | Dropout (0.3)   |                     |

| Flatten part | Flatten layer | Flatten layer | Flatten layer | —                     |
| Concatenate part | Dense (256) | Dense (256) | Dense (256) | —                     |
| Concatenate layer | Dense (256) | Dense (256) | Dense (256) | —                     |
| Output layer  | 1st order frequency | 2nd order frequency | 3rd order frequency | 4th order frequency | 5th order frequency |

Note. 6 × Conv (1, 1, 32) means 6 3D convolution blocks, each block including a 3D convolution layer, a batch normalization layer, and a ReLU activation layer. The convolution size is (1, 1, 1), stride is (1, 1, 1), and the out channel is 32; Conv (1, 2, 32) means a convolution layer with a convolution kernel of (1, 1, 1), a step of (2, 2, 2), and an out channel of 32; Dropout (0.3) means a dropout layer which randomly sets input units to 0 with a frequency of 0.5 at each step during training time; Dense (256) means a dense layer of 256 units.

**Figure 6:** The 3D-CNN architecture. (a) 3D-CNN architecture. (b) Details of stage. (c) Details of conv block.
MAE = \frac{1}{n} \sum_{i=1}^{n} |\bar{y}_i - y_i|,

MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\bar{y}_i - y_i}{y_i} \right| \times 100\%,

where $\bar{y}_i$ and $y_i$ represented true value and predicted value and $n$ represented the number of subjects in dataset.

3.3. Compared Work. In order to explore the impact of 3D representation on the prediction results, this study constructed a multilayer perceptron (MLP) and the simplest and most used artificial neural network, as a comparison network. The entire neural network was obtained by cropping the 3D-CNN mentioned above, deleting all the layers related to 3D convolution, leaving only the input layer of design parameters, fully connected parts, and the output layer. In addition, other parts such as loss function, activation function, and metric were consistent with the 3D-CNN.

3.4. Implementation. We implemented the proposed framework based on TensorFlow and Keras library using dual Intel(R) Xeon(R) Gold 6248 CPU 2.50 GHz and a GPU of NVIDIA TESLA V100S 32G. The networks were trained with the following hyperparameters: Adam optimizer with a learning rate of 0.001, dropout rate $= 0.3$, and batch size $= 32$. The trainable weights were randomly initialized from the Gaussian distribution and updated with standard backpropagation.

4. Results and Discussion

The prediction results of all models in this study on the test set are summarized in Table 1. The first 12 rows in the table are the results of the 3D-CNN with input tensors of different sizes, and the last row is the results of the MLP. The second and third columns of the table are the key point interpolation number and the cross-section interpolation number, which are used to control the size of the input tensor. The fourth column is the total loss of the corresponding 3D convolutional network on the test set. The total loss is the sum of the mean absolute error (MAE) of the first 5 frequency prediction values. Columns 5 to 9 are the mean absolute error (MAE) and mean absolute percentage error (MAPE) of each order of frequency prediction. The upper one in each cell is MAE, and the lower one is MAPE.

According to the results in the table, it could be found that, among all 3D-CNN models, the No.12 model had the smallest prediction error and the best prediction effect. The input tensor shapes of this model are (31, 19, 3) of girder and (31, 19, 19, 3) of pier. The total loss of this model was 0.3687, the MAEs of 1st to 5th order frequency were 0.0473, 0.0540, 0.0705, 0.0912, and 0.1057, respectively, and the MAPEs were 14.5%, 12.7%, 11.5%, 12.3%, and 11.5%. As the order increases, the MAE of the predicted value got larger and larger, but the MAPE got smaller and smaller. It is the same in the results of other models. This is because as the order increases, the frequency also increases.

The No. 12 model’s percentage error distribution of each order frequency model is plotted in Figure 7, as well as its probability density function (PDF) curve by kernel function estimation (KDE). In statistics, KDE is a nonparametric way to estimate the PDF of a random variable. Since KDE does not use the prior knowledge of data distribution and does not attach any assumptions to the data distribution, it is a method to study the data distribution characteristics from the data sample itself, so it can describe the data distribution more accurately, KDE uses a smooth peak function, kernel function, to fit the observed data points, thereby simulating the true probability distribution curve. Considering the ease of use of the function in the waveform synthesis calculation, this study uses the Gaussian curve (normal distribution curve) as the kernel function of KDE.

It could be found from Figure 7 that these percentage errors were roughly normally distributed. And the peaks all appear at positions less than 0 slightly, which showed that the frequency predicted by the model was often less than the correct value. In addition, comparing the distribution of frequencies of different orders, it could be found that, as the order increased, the peak value of the distribution became higher, and the distribution range became narrower, showing a trend of being taller and thinner. This showed that, for high-order frequencies, the prediction error was smaller and the model had better prediction performance. This is consistent with the changes of MAPE.

4.1. Comparing with MLP. The last column in Table 3 lists the total loss of MLP, which was 0.4995. The MAEs of the predicted values of the first 5 frequencies were 0.0740, 0.0827, 0.0956, 0.1137, and 0.1336, and the corresponding MAPEs were 26.9%, 22.7%, 17.4%, 15.9%, and 15.2%, respectively. This error was the worst among all 13 models and was much larger than the model using 3D-CNN. Compared with the best 3D-CNN model, the total loss of the No. 12 model was reduced by 0.1308, which was about 26%. Even the worst 3D-CNN model, the No. 2 model, the total loss was also reduced by about 22%. From the point of view of MAPE, the gap between the two was also obvious. Compared with the No. 12 model, the largest difference was at the first-order frequency, with a difference of about 12%, and the smallest was at the fourth and fifth-order frequencies, with a difference of about 3%.

The percentage error distribution and probability density function curve of MLP are also plotted in Figure 7. Like 3D-CNN, the percentage error of MLP was roughly Gaussian. The position where the peak appears was also slightly less than 0, and as the frequency order increases, the distribution began to become higher and narrower. It should be noted that, in the percentage error distribution of the first- and second-order frequencies, a peak appeared in the error of the MLP at the -100% scale. This is because there are some MLP
percentage errors whose value is greater than -100%, but in statistics, they are all classified at -100%. This shows that MLP had negative values in the prediction of low-order frequencies, which was unacceptable.

Comparing the PDF curves of MLP and 3D-CNN, it can be found that no matter which frequency it was, the PDF curve of 3D-CNN was higher and thinner than that of MLP. It was especially prominent at the first 2 order frequencies. This showed that the 3D representation of the bridge structure had a significant improvement in the prediction of frequencies, especially in the prediction of low-order frequencies.

The calculation speed of each model on the test set is also listed in Table 3. Compared with MLP, 3D-CNN adds many 3D convolutional layers in structure, which requires additional convolution calculations. Therefore, when calculating the results, time-consuming increased. MLP calculation took 6 ms for each result, while the fastest 3D-CNN model took 8 ms. However, considering the improvement of prediction accuracy, the time-consuming increase is still worthwhile.

4.2. Influence of Input Tensor Size. The worst model among all 3D-CNN models is the No. 2 model, and the shapes of its input tensor are (11, 9, 11, 3) of girder and (11, 9, 9, 3) of pier.
Comparing the results of the No. 2 model and the No. 12 model, it shows that the increase in the size of the input tensor could indeed improve the accuracy of the prediction. Figure 8 shows it more intuitively, in which all 3D-CNN models’ total loss are drawn. It can be seen from the figure that as key point interpolation number increases, the total loss was on a downward trend, as well as the cross-section interpolation number.

Table 3: Results of all models on the test set.

| No. | Ns   | Np   | Total loss | 1st | 2nd | 3rd | 4th | 5th | Speed |
|-----|------|------|------------|-----|-----|-----|-----|-----|-------|
| 1   | 0    | 0    | 0.3867     | 0.0471 | 0.0577 | 0.0750 | 0.0945 | 0.1124 | 8     |
| 2   | 0    | 1    | 0.3901     | 0.0472 | 0.0585 | 0.0758 | 0.0947 | 0.1139 | 8     |
| 3   | 0    | 3    | 0.3898     | 0.0472 | 0.0577 | 0.0752 | 0.0951 | 0.1146 | 9     |
| 4   | 0    | 5    | 0.3793     | 0.0460 | 0.0564 | 0.0729 | 0.0902 | 0.1138 | 9     |
| 5   | 10   | 0    | 0.3773     | 0.0462 | 0.0577 | 0.0736 | 0.0916 | 0.1082 | 9     |
| 6   | 10   | 1    | 0.3766     | 0.0445 | 0.0563 | 0.0723 | 0.0916 | 0.1119 | 9     |
| 7   | 10   | 3    | 0.3758     | 0.0441 | 0.0573 | 0.0730 | 0.0926 | 0.1089 | 9     |
| 8   | 10   | 5    | 0.3738     | 0.0441 | 0.0569 | 0.0729 | 0.0907 | 0.1093 | 10    |
| 9   | 20   | 0    | 0.3771     | 0.0458 | 0.0568 | 0.0732 | 0.0918 | 0.1094 | 10    |
| 10  | 20   | 1    | 0.3788     | 0.0454 | 0.0560 | 0.0752 | 0.0917 | 0.1105 | 10    |
| 11  | 20   | 3    | 0.3721     | 0.0447 | 0.0568 | 0.0737 | 0.0894 | 0.1075 | 10    |
| 12  | 20   | 5    | 0.3687     | 0.0473 | 0.0540 | 0.0705 | 0.0912 | 0.1057 | 12    |
| 13  | —    | —    | 0.4995     | 0.0740 | 0.0827 | 0.0956 | 0.1137 | 0.1336 | 6     |

Note: all the units of error are Hz; the unit of speed is milliseconds per data

However, this improvement is not big and has side effects. The calculation speed of each model on the test set is also listed in Table 3. It can be found from Table 3 that, as the volume of the tensor increased, the calculation time also increased. The No. 2 model took 8 ms to calculate each result, while No. 12 model took 12 ms. The time spent increased by 25%, but the total loss value only decreased by 5.5%. Therefore, for some application scenarios that require high calculation speed, the smallest input tensor size should be used.

5. Conclusion

The traditional artificial neural network limited by the input form cannot provide sufficient features, resulting in poor performance in the complex prediction problem. In this regard, this study proposes a new 3D representation of bridge structures, which not only retains the original design information but also provides additional spatial information. Based on this, a corresponding 3D convolutional neural network was built and applied to predict the frequency of bridges.

On the test set, the best 3D convolutional neural network’s mean absolute percentage errors of the first 5 frequencies were 14.5%, 12.7%, 11.5%, 12.3%, and 11.5%, and the mean absolute errors were 0.0473, 0.0540, 0.0705, 0.0912, and 0.1057, respectively. The comparison with the prediction results of traditional neural networks, a multilayer...
perceptron, proves that this 3D representation can effectively reduce prediction errors and improve the network’s fitting ability. The total error of the 3D convolutional neural network was reduced by 33%, and the average percentage error for the first-order frequency was reduced by 12%. In addition, this study also studied the influence of the size of the 3D representation on the prediction results. The results show that the increase in size can slightly reduce the prediction error, but it will greatly increase the calculation time.

The 3D representation proposed in this study, combined with the corresponding 3D convolutional layer, acts as a geometric shape feature extractor. Therefore, it can be used not only in frequency estimation but also in any tasks related to structural geometry. Besides, it can also be used in combination with other types of neural networks, such as combining with LSTM networks to predict the deflection of bridge structures under earthquake and so on. Its application in other tasks is worthy of further study.

Data Availability
The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

References

[1] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, “Deep learning for computer vision: a brief review,” Computational Intelligence and Neuroscience, vol. 2018, Article ID 7068349, 2018.

[2] A. B. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan, “Speech recognition using deep neural networks: a systematic review,” IEEE Access, vol. 7, pp. 19143–19165, 2019.

[3] T. Young, D. Hazarika, S. Poria, and E. Cambria, “Recent trends in deep learning based natural language processing [Review Article],” IEEE Computational Intelligence Magazine, vol. 13, no. 3, pp. 55–75, 2018.

[4] A. Zhang, K. C. P. Wang, B. Li, E. Yang, X. Dai, and Y. Peng, “Automated pixel-level pavement crack detection on 3D asphalt surfaces using a deep-learning network,” Comput Civ Infrastruct Eng, vol. 32, pp. 805–819, 2017.

[5] L. Ding, W. Fang, H. Luo, P. E. D. Love, B. Zhong, and X. Ouyang, “A deep hybrid learning model to detect unsafe behavior: integrating convolution neural networks and long short-term memory,” Automation in Construction, vol. 86, pp. 118–124, 2018.

[6] A. Ozdagli and X. Koutsoukos, “Machine learning based novelty detection using modal analysis,” Comput Civ Infrastruct Eng, vol. 34, pp. 1119–1140, 2019.

[7] M. Azimi, A. Eslamlou, and G. Pekcan, “Data-driven structural health monitoring and damage detection through deep learning: state-of-the-art review,” MD, vol. 20, 2020.

[8] H. Nguyen, H. Moayed, L. K. Foong, H. Al Najjar, W. Josoh, and A. Rashid, “Optimizing ANN models with PSO for predicting short building seismic response,” Engineering with Computers, vol. 36, pp. 823–837, 2020.

[9] P. Asteris and M. Nikoo, “Artificial bee colony-based neural network for the prediction of the fundamental period of infilled frame structures,” Neural Computing & Applications, vol. 31, pp. 4837–4847, 2019.

[10] T. Garcia-Segura, V. Yepes, and D. Frangopol, “Multi-objective design of post-tensioned concrete road bridges using artificial neural networks,” Structural and Multidisciplinary Optimization, vol. 56, pp. 139–150, 2017.

[11] S. Chatterjee, S. Sarkar, S. Hore, N. Dey, A. Ashour, and V. Balas, “Particle swarm optimization trained neural network for structural failure prediction of multistoried RC buildings,” Neural Computing & Applications, vol. 28, pp. 2005–2016, 2017.

[12] R. Zhang, Y. Liu, and H. Sun, “Physics-guided convolutional neural network (PhyCNN) for data-driven seismic response modeling,” Engineering Structures, vol. 215, pp. 1–24, 2020.

[13] R. Zhang, L. Meng, Z. Mao, and H. Sun, “Spatiotemporal deep learning for bridge response forecasting,” Journal of Structural Engineering, vol. 147, pp. 1–9, 2021.

[14] Z. Yue, Y. Ding, and H. Zhao, “Deep learning-based minute-scale digital prediction model of temperature-induced deflection of a cable-stayed bridge: case study,” Journal of Bridge Engineering, vol. 26, Article ID 05021004, 2021.

[15] C. Qi, H. Su, K. Mo, and L. Guibas, “PointNet: deep learning on point sets for 3D classification and segmentation,” in Proceedings of the 30th IEEE Conf Comput Vis Pattern Recognition, CVPR 2017 2017, pp. 77–85, Honolulu, HI, USA, January 2017.

[16] C. Qi, L. Yi, H. Su, and L. Guibas, I Guyon, UV Luxburg and S. Bengio, “PointNet plus plus: deep hierarchical feature learning on point sets in a metric space,” in Neural Information Processing Systems (NIPS), H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, p. 10010, NorthTorrey Pines Rd, La Jolla, California 92037 USA, 2017.

[17] Y. Y Feng, Y. Y Feng, H You, X Zhao, and Y Gao, “MeshNet: mesh neural network for 3D shape representation,” 2018, https://arxiv.org/abs/1811.11424.

[18] H Su, S Maji, E Kalogerakis, and E. Learned-Miller, “Multi-view convolutional neural networks for 3D shape recognition,” IEEE International Conference on Computer Vision, vol. 2015, pp. 945–953, 2015.

[19] J. W. Jiang, D. Bao, Z. Q. Chen, X. B. Zhao, Y. Gao, and AAAI, MVLCNN: Multi-Loop-View Convolutional Neural Network for 3D Shape Retrieval, ACM Digital library, 2019.

[20] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, and X. Tang, “3D ShapeNets: a deep representation for volumetric shapes,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1912–1920, Las Vegas, NV, USA, June 2015.

[21] S. Kumawat and S. Raman, “LP-3DCNN: unveiling local phase in 3D convolutional neural networks,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 4898–4907, Las Vegas, NV, USA, June 2019.

[22] S. Ji, W. Xu, M. Yang, and K. Yu, “3D Convolutional neural networks for human action recognition,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, pp. 221–231, 2013.

[23] Q. Dou, H. Chen, L. Yu, L. Zhao, J. Qin, and D. Wang, “Automatic detection of cerebral microbleeds from MR images via 3D convolutional neural networks,” IEEE Transactions on Medical Imaging, vol. 35, pp. 1182–1195, 2016.

[24] R. Akbari, S. Maadani, and S. Maalek, “On the fundamental natural frequency of bridge decks: review and applications,” Proc Inst Civ Eng Struct Build, vol. 171, pp. 931–945, 2018.
[25] H. Wang, C. Xie, D. Liu, and S. Qin, "Continuous reinforced concrete rigid-frame bridges in China," *Practice Periodical on Structural Design and Construction*, vol. 24, Article ID 05019002, 2019.

[26] K. C. Thienel, T. Haller, and N. Beuntner, "Lightweight concrete--from basics to innovations," *Materials*, vol. 13, 2020.

[27] X. Lin, L. Zhang, Q. Guo, and Y. Zhang, "Dynamic finite element model updating of prestressed concrete continuous box-girder bridge," *Earthquake Engineering and Engineering Vibration*, vol. 8, pp. 399–407, 2009.

[28] R. He, W. He, and G. H. Geng, "Research on dynamic properties of concrete steel tubular arch bridge based on ANSYS," *Applied Mechanics and Materials*, vol. 71–78, pp. 3233–3236, 2011.

[29] L. Fei, *Study of the Vibration Characteristics for Bridge Structures under the Influence of Prestress Effect*, Chongqing Jiaotong University, Chongqing, China, 2013.