Seismic damage identification of cable-stayed bridge in near-real-time using unsupervised deep neural network

Kim, Minkyu; Song, Junho

Seismic damage identification of cable-stayed bridge in near-real-time using unsupervised deep neural network

Minkyu Kim

Department of Civil & Environmental Engineering, Seoul National University

Junho Song

Department of Civil & Environmental Engineering, Seoul National University

ABSTRACT: Prompt damage identification of infrastructure systems is essential for effective post-disaster responses. However, most infrastructure systems have a high level of structural complexity, making damage identification extremely difficult. To overcome the challenge, the authors recently proposed a deep neural network (DNN) based framework for identifying the seismic damage based on the structural response data recorded during an earthquake event (Kim and Song, 2022). The DNN of the proposed framework is constructed by a Variational Autoencoder, one of the self-supervised DNNs capable of constructing a continuous latent space of input data by learning probabilistic characteristics. The DNN model is trained using the covariance matrices of the snapshot of the response data obtained from the undamaged structure. To consider the load-dependency, the undamaged state of the structure is represented by the covariance matrix, which is closest to that obtained from the measured seismic response in the latent space. To identify the severity of the structural damage, a structural damage index based on the difference in the covariance matrices is introduced. This paper improves the DNN-based framework to facilitate its applications to complex structural systems such as the Incheon Grand Bridge, a reinforced concrete cable-stayed bridge in South Korea. To generate train, validation, and test datasets, structural analyses are first performed under the ground motions from the PEER-NGA strong motion database. The proposed framework is verified with near-real-time simulations using ground motions with various time steps from the test dataset. The example shows that the proposed framework can accurately identify seismic damage of the complex structural system in near-real-time.

1. INTRODUCTION

Structural Health Monitoring (SHM), utilizing numerous sensor data and measurements, has arisen as an alternative for traditional inspection methods. SHM can handle a variety of important tasks related to the infrastructure systems, including post-disaster damage identification. Generally, SHM processes are categorized into long-term and short-term SHM (Dawson, 1976). The long-term SHM utilizes periodically updated information about the ability to perform its intended function. On the other hand, the short-term SHM aims to rapidly identify structural condition changes and provide information on the structural integrity in near-real-time.

Recently, pattern recognition methods combined with vibration-based damage identification are considered particularly suitable for the short-term SHM. This is because such methods can utilize the pre-trained model, such as a Deep Neural Network (DNN), using the features extracted from the vibration signals (Pathirage et al., 2018). Therefore, it requires low computational effort to recognize the changes in vibration characteristics. This paper focuses on developing a near-real-time damage identification method under earthquakes using DNN-based pattern recognition.

Nonetheless, identifying damage from the raw vibration signals is generally challenging because of its insensitivity to structural damage. Furthermore, massive infrastructure systems, such as cable bridges, are more challenging to identify structural damage because they have a high level of structural complexity, and the dominant natural frequencies are in the narrow range. By contrast, the structural response characteristics, which can be obtained by post-processing the raw vibration signals, are relatively sensitive to structural damage. In this paper, the sample covariance matrix of the raw signals is utilized as the response characteristics.

Response characteristics, however, tend to show more variability under seismic load conditions. This makes it difficult to identify structural damage by comparing response characteristics during an earthquake event. To address this issue, this study adopts a variational autoencoder (VAE) (Kingma and Welling 2013), an autoencoder (AE) with probabilistic learning. VAE learns how to encode input data into a distribution over the continuous latent space. Because of this continuity, the distance between two points in the latent space can be utilized to measure the similarity of corresponding input data. Therefore, the latent variables should be clustered in terms of the load conditions because uncertain characteristics are significantly affected by those. In addition, since local damage does not drastically change the response characteristics, the response characteristics in a damaged state are located around a latent cluster with the most similar load condition. Therefore, it is possible to find the undamaged structural response characteristic under the most similar load condition to the damaged one, based on the distance in the latent space.

The authors recently proposed a DNN-based framework for near-real-time damage identification under the seismic load condition (Kim and Song, 2022). In this paper, the DNN-based framework is improved to deal with complex structural systems. The proposed framework consists of four steps: (1) structural analyses of the target structure at an undamaged state are first performed under the ground motions from the PEER-NGA strong motion database (Chiou et al. 2008), and covariance matrices are prepared as input dataset; (2) the DNN model is trained to learn the latent space and fine-tuned using the validation dataset; (3) real-time simulations are performed with the test dataset for various damage conditions to verify the pre-trained network; and (4) the SDIs are calculated in near-real-time. A structural damage index (SDI) based on the difference in the covariance matrices is introduced. As the DNN model, convolutional VAE (CVAE; Goodfellow et al., 2016) is selected. As a numerical example of the proposed framework, the Incheon Grand Bridge, the reinforced concrete cable-stayed bridge in South Korea, is investigated under seismic load conditions.

2. THEORETICAL BACKGROUND FOR DEEP NEURAL NETWORK

2.1. *Autoencoder*

The aim of Autoencoder (AE) is to learn patterns hidden in a set of data by finding a DNN structure that can reconstruct the input data by going through the mapping functions called encoder and decoder sequentially. A classical AE consists of an encoder and decoder with a single hidden layer (Vincent et al. 2010). One can construct a deep AE by introducing multiple hidden layers. A conceptual illustration of deep AE is shown in Figure 1.

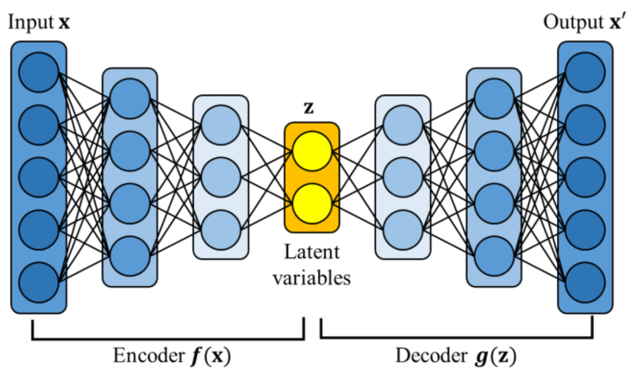


Figure 1 Conceptual illustration of AE.

Encoder: The mapping function $f(\mathbf{x})$, which transforms a d-dimensional input vector $\mathbf{x} \in \mathbb{R}^d$ into an r-dimensional latent variable $\mathbf{z} \in \mathbb{R}^r$, in which $d > r$, is called an encoder. $\boldsymbol{f}(\mathbf{x})$ is usually described as a nonlinear transformation

$$\mathbf{z} = \boldsymbol{f}(\mathbf{x}) = \boldsymbol{\sigma}(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{r \times d}$ denotes the mapping weight matrix of the encoder; $\mathbf{b} \in \mathbb{R}^r$ is the bias vector; and $\boldsymbol{\sigma}$ is the activation function, which is usually a nonlinear function such as sigmoid, tangent hyperbolic, rectified linear unit (ReLU), and exponential linear unit (ELU) function.

Decoder: The mapping function $\boldsymbol{g}(\mathbf{z})$, which transforms the latent variable $\mathbf{z}$ back into a reconstructed vector $\mathbf{x}' \in \mathbb{R}^d$, is called a decoder. Usually, $\boldsymbol{g}(\mathbf{z})$ is described as

$$\mathbf{x}' = \boldsymbol{g}(\mathbf{z}) = \boldsymbol{\sigma}(\widehat{\mathbf{W}}\mathbf{h} + \hat{\mathbf{b}}) \tag{2}$$

where $\widehat{\mathbf{W}} \in \mathbb{R}^{d \times r}$ denotes the mapping weight matrix of the decoder; $\hat{\mathbf{b}} \in \mathbb{R}^d$ is the bias vector; and σ is the activation function described above.

To estimate the optimal parameters of AE model $\boldsymbol{\theta} = \left[\mathbf{W}, \mathbf{b}, \widehat{\mathbf{W}}, \hat{\mathbf{b}}\right]$ based on the training data, AE algorithms aim to minimize the loss function $\mathcal{L}_{AE}(\mathbf{x})$ often defined as the mean squared error (MSE)

$$\mathcal{L}_{AE}(\mathbf{x}) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left\|\boldsymbol{x}^{(i)} - \boldsymbol{g}\left(\boldsymbol{f}(\boldsymbol{x}^{(i)})\right)\right\|^2 \tag{3}$$

where m is the number of training samples; and $\boldsymbol{x}^{(i)}$ is the i-th input. $\mathcal{L}_{AE}(\mathbf{x})$ is generally difficult to minimize due to its non-linearity, gradient descent-based optimizers, such as Adam (Kingma and Ba, 2014), are commonly used.

2.2. *Variational autoencoder*

A variational autoencoder (VAE) is also composed of both an encoder and a decoder, but the VAE encodes the input data as a probabilistic distribution over the latent space by introducing the regularization term. The VAE uses a variational inference approach to learn latent representation, which results in a loss function with a regularization term, termed the Stochastic Gradient Variational Bayes (SGVB) estimator (Kingma and Welling, 2013).

VAE assumes that the data is generated from the decoder $p_\theta(\mathbf{x}|\mathbf{z})$, and the encoder learns an approximation of the true posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$, denoted by $q_\phi(\mathbf{z}|\mathbf{x})$ where ϕ and θ are the parameters of the encoder and decoder respectively. The loss function of VAE is given as

$$\mathcal{L}_{VAE}(\boldsymbol{x}; \phi, \theta) = D_{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \parallel p_\theta(\boldsymbol{z})\right) - E_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}(log\, p_\theta(\boldsymbol{x}|\boldsymbol{z})) \tag{4}$$

where D_{KL} stands for the Kullback–Leibler divergence (KL divergence); $p_\theta(\mathbf{z})$ is the true prior distribution over the latent variables; and $q_\phi(\mathbf{z}|\mathbf{x})$ is the variational distribution. Note that the first term in the loss function makes $q_\phi(\mathbf{z}|\mathbf{x})$ similar to $p_\theta(\mathbf{z})$, i.e., the regularization term, and the second term is the reconstruction error. Introducing the regularization term to the loss function makes the latent space have the following two main properties: (1) continuity; and (2) completeness, which respectively mean that close points in the latent space should give similar output once decoded, and a point sampled from the latent space should give meaningful outputs.

Figure 2 provides a conceptual illustration of VAE. The training process of the VAE model consists of the following four steps: (1) the input is encoded as a probabilistic distribution over the latent space (characterized by $\mu_\mathbf{z}$ and $\log \sigma_\mathbf{z}^2$ if the latent variables are assumed to follow a multivariate Gaussian distribution); (2) the latent variable $\mathbf{z}$ is sampled in the latent space according to the encoded distribution; (3) the sampled variable is decoded through $p_\theta(\mathbf{x}|\mathbf{z})$; and (4) the loss value is calculated, and the parameters of VAE is updated by back-propagation through the network.

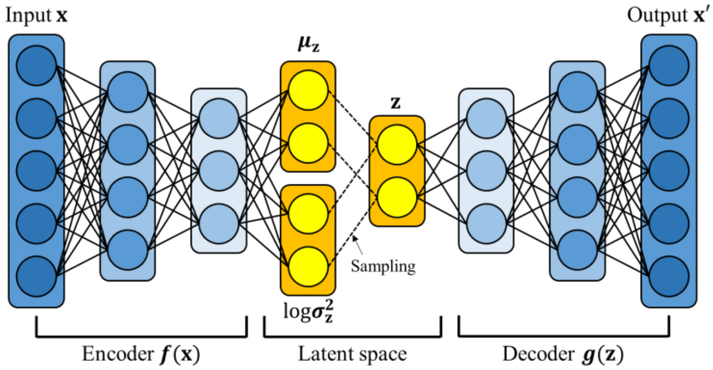


Figure 2 Conceptual illustration of VAE.

3. PROPOSED DNN-BASED DAMAGE IDENTIFICATION FRAMEWORK

A DNN-based damage identification framework recently proposed by the authors (Kim and Song, 2022) is improved in this paper, as described below. Figure 3 illustrates the proposed framework divided into two main parts: (1) offline, and (2) online processes.

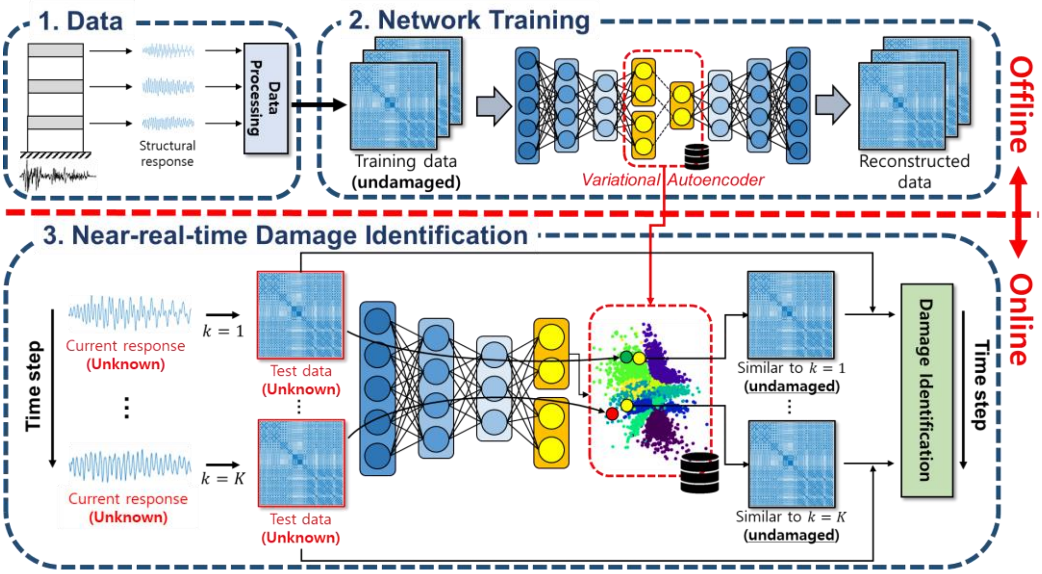


Figure 3 Process of proposed DNN-based damage identification framework.

3.1. *Offline process*

Before the near-real-time damage identification by the online process, the data processing and network training should first be performed in the offline process. The offline process has two steps: (1) data collection from structural analyses to obtain sample covariance matrices at the undamaged state; and (2) training the DNN model using the dataset from the first step. The offline process is illustrated in the upper part of Figure 3.

3.1.1. *Structural analysis and database generation*

To obtain training data of structural responses under seismic ground motions, structural analyses are performed using the model of the undamaged target structure. Live loads are applied before the earthquake event to simulate the normal operational circumstance. After the data collection, the sample covariance matrices $\mathbf{C}$ are calculated as

$$\mathbf{C} = \frac{1}{T-1}\mathbf{U}^{\mathrm{T}}\mathbf{U} \tag{5}$$

where $\mathbf{U}$ is a snapshot matrix of recorded response at certain time point; and T is the number of the time samples in $\mathbf{U}$. Lastly, the collected covariance matrices are shuffled and split into the train, validation, and test dataset following a pre-defined ratio.

In the previous study (Kim and Song, 2022), the approximated flexibility matrix, which requires the additional modal analysis process to obtain accurate modal characteristics and structural information, is used as the input data. In contrast, in this paper, the sample covariance matrix is only calculated from the raw response data, which requires no further post-processing and information. Therefore, the proposed framework is more robust to variability of earthquake ground motions than the framework of the previous study.

3.1.2. *Network training*

After database generation, the DNN model is trained using the covariance matrices from the undamaged structure as input data. Convolutional VAE (CVAE), which is suitable to learn features from two-dimensional data, is selected as the DNN model to construct the disentangled latent space of $\mathbf{C}$. The network is trained to minimize the reconstruction error between the input and decoded data, as well as to establish the meaningful latent space of given input data. To utilize the pre-established latent space in the online damage identification process, the decoder is removed from the network after training, and every point of the entire dataset in the latent space is stored in the database.

3.2. *Online process*

The online process for the near-real-time damage identification is performed using the trained DNN model and the pre-established latent space obtained from the offline process. The online damage identification process has two steps: (1) searching the latent space for the covariance matrix representing the undamaged state, which is closest to the current data; and (2) calculating the structural damage index (SDI) based on the difference between the undamaged and current covariance matrix. These two steps are performed at every time step. The conceptual illustration of the online process is shown in the lower part of Figure 3.

3.2.1. *Searching DB for covariance matrix representing undamaged state*

In the online process, structural responses are obtained from sensors in real-time while the actual state of the structure is unknown. Using Eq. (5), the covariance matrix at the unknown state $\mathbf{C}^u$ is obtained from the snapshot of recorded responses with a certain length of the time window. Here, the superscript 'u' stands for 'unknown.' Note that additional post-processing, such as modal analysis, is not required to obtain $\mathbf{C}^u$ from the measured response data, as mentioned. Then, as with the method used by Kim and Song (2022), the corresponding point in the latent space can be obtained through the encoder of the pre-established DNN. The similarity can be quantified by measuring the distance between the current (unknown) and pre-stored (undamaged) data. Since the network training or any computationally intensive calculations are not included in this step, the entire process can be finished in a second. As a result, the pair of the matrices, i.e., the covariance matrix representing the undamaged state, $\mathbf{C}$ and the current matrix $\mathbf{C}^u$, is obtained in near-real-time for the next step.

3.2.2. *Calculation of structural damage index*

The SDI is then calculated based on the difference between $\mathbf{C}$ and $\mathbf{C}^u$. In this paper, the SDI is defined as the mean squared error (MSE) between $\mathbf{C}$ and $\mathbf{C}^u$ as follows:

$$SDI_i = \frac{1}{N}\sum_{j=1}^{N}\left(\mathbf{C}_{ij} - \mathbf{C}_{ij}^u\right)^2 \tag{6}$$

where SDI_i is the SDI of i-th sensor; N is the number of sensors; and $\mathbf{C}_{ij}$ and $\mathbf{C}_{ij}^u$ are elements of $\mathbf{C}$ and $\mathbf{C}^u$, respectively. Since this step includes just a few matrix calculations, SDI_i can also be obtained rapidly. As a result, the near-real-time damage identification can be performed by repeating the online process at every time step.

4. NUMERICAL INVESTIGATION

4.1. *Target structure: Incheon Grand Bridge*

As a target structure, the Incheon Grand Bridge in Figure 4(a) is selected (Kim et al., 2021). The total bridge length is 1,480 m, with a main span of 800 m. The bridge is supported by two pylons (Py-L and Py-R) and four piers (Pi-L1, Pi-L2, Pi-Rl, and Pi-R2). The original nonlinear finite element (FE) model was constructed by Kim et al. (2021) using OpenSees (Mckenna et al., 2010). In this paper, the FE model is reconstructed using OpenSeesPy (Zhu et al., 2018), presented in Figure 4(b), to utilize the libraries and applications in Python. At the undamaged state, the modal periods of the first six modes are 7.169, 5.457, 4.529, 3.677, 3.030, and 2.852 sec, respectively.

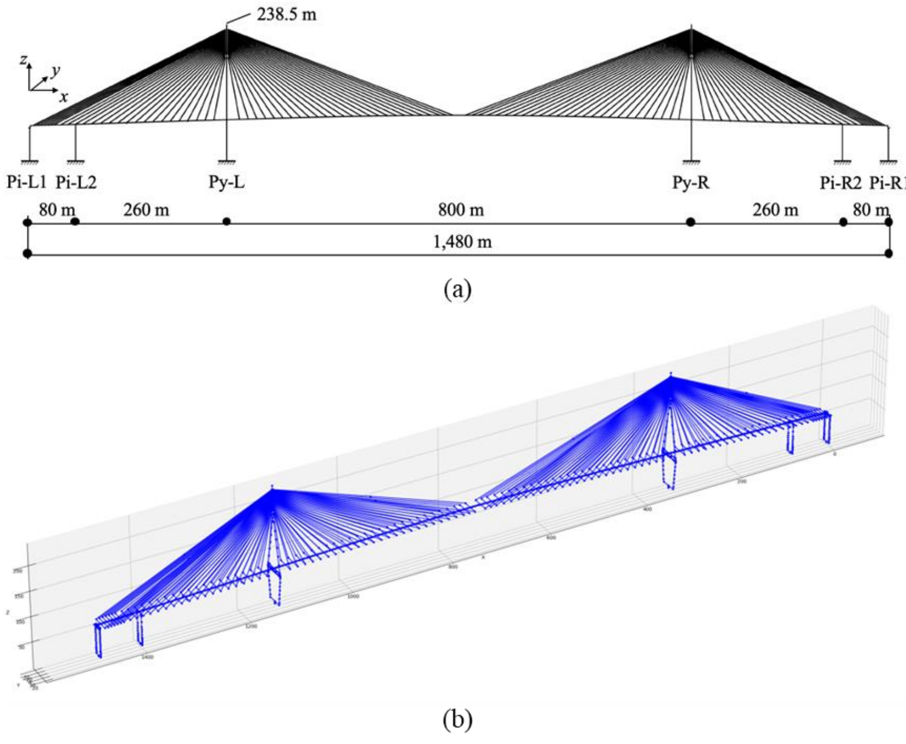


Figure 4 (a) Configuration; and (b) OpenSeesPy model of the Incheon Grand Bridge.

4.2. *Data generation and pre-processing*

Structural analyses are performed using seismic ground motions from the PEER-NGA strong motion database (Chiou et al. 2008). A total of 3,512 ground motions are used and split into train, validation, and test datasets with a ratio of 9:0.5:0.5. In addition, random excitations simulating live loads are applied to the deck before an earthquake event. To capture the response properties effectively, the time window of the snapshot matrix should be set long enough. To this end, the length of the time window is set to 60 sec, and the time interval of the time window is set to 2 sec.

It is assumed that a total of 33 dual-axis accelerometers are installed on the bridge to record the *x*- and *y*-axis responses. After the data collection, the covariance matrices $\mathbf{C}$ are calculated using Eq. (5). Note that the dimension of the covariance matrix $\mathbf{C}$ is 66×66 since there are 33 measured responses for the *x*- and *y*-axis respectively. Finally, the datasets are scaled to the range of $[-1, 1]$, normalized by the largest or smallest value.

4.3. *Network training*

CVAE is then trained to construct the disentangled latent space of the undamaged covariance matrix. The architecture of CVAE is proposed as illustrated in Figure 5. CVAE generally consists of three main parts: (1) the convolutional encoder, (2) latent space; and (3) the convolutional decoder. After convolutional operations, the output of the last convolutional layer is reshaped into the one-dimensional vector by *Flatten* layer, which is followed by two *Dense* layers with 16 nodes representing μ_z and $\log \sigma_z^2$ of the latent distribution, respectively. In general, the decoder has the inverse architecture of the encoder and uses the same hyper-parameters as the encoder (Kim and Song, 2022). In contrast, this paper adopts the skip connection (He et al., 2016) for the decoder to directly utilize more information from the latent space for reconstructing the input data. As shown in Figure 5, a total of three skip connection blocks are used to construct the decoder. The value of the loss function $\mathcal{L}_{VAE}(\cdot)$ is calculated using the SGVB estimator at the last step of the forward-propagation, and the value is back-propagated through the network to optimize network parameters.

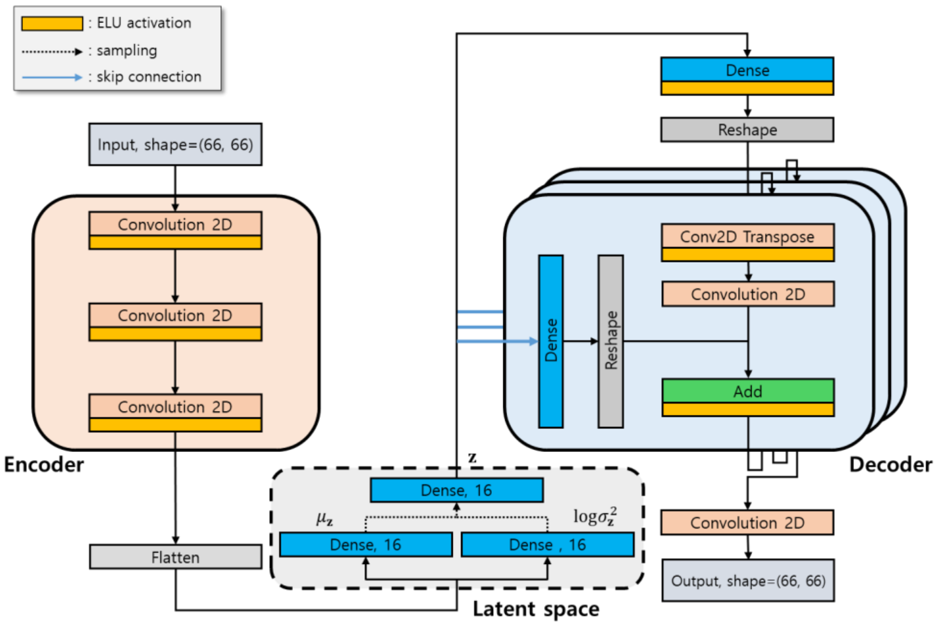
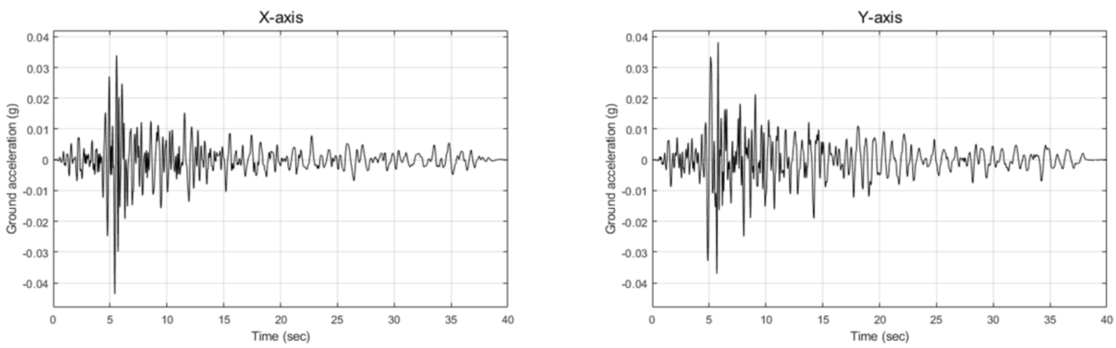


Figure 5 Architecture of CVAE proposed in this study.

CVAE is constructed using the Python deep learning library Keras with the Tensorflow backend and trained on a server with 2x Intel(R) Xeon(R) Gold 6126 2.60GHz, two NVIDIA TITAN RTX graphics cards, and 128GB RAM. The numbers of epochs and batch size are set to 500 and 32, respectively. The Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.001 is used for minimizing the loss function. The training process takes about 24 hours while the loss function converges fast and stably without overfitting or explosion of the validation loss.

4.4. *Near-real-time damage identification*

To verify the performance of the pre-trained CVAE, real-time test simulations are performed with a test ground motion randomly selected from the test dataset. The records of *x*- and *y*-axis component of the test ground motion are shown in Figure 6(a) and (b), respectively. The total length is 40.0 s with 200 Hz of the sampling frequency, and the peak ground acceleration (PGA) occurs at 5.41 s with a value of 0.0436 g. Random excitations are also applied to the deck before the earthquake event to simulate the operational situation. The covariance matrix is obtained at every time step, and SDI_i $(i = 1, \cdots, 66)$ are calculated simultaneously through the pre-trained network.



(a) (b)

Figure 6 Ground acceleration records of the test ground motion in (a) *x*-axis; and (b) *y*-axis.

To verify the identification performance, three damage cases are investigated: (1) Case 1: the lower part of Py-L is damaged; (2) Case 2: the lower parts of Py-L and Py-R are damaged; and (3) Case 3: a single cable is damaged. Damages are simulated by 50% degradation in Young's modulus, occurring at the time of PGA. The total duration of the simulation is set to 130.0 s to verify the stable performance over a long period of time. The near-real-time identification results for the three cases in Figure 7 show that the proposed method successfully identifies structural damage under an earthquake event in near-real-time.

In Case 1, the SDIs of Py-L and cables attached to Py-L increase as the damage occurs. The SDI of Py-L is estimated as around 0.9×10^{-6} at the simulation end time. Furthermore, it only takes less than a second to obtain the SDIs for every time step, which is much shorter than the pre-set time interval. The damage can be readily located since only the SDIs of cables attached to Py-L are inaccurately identified as damaged because of their high correlation.

In Case 2, the SDIs of Py-L, Py-R, and cables increase as the damage occurs. The SDIs of Py-L and Py-R are estimated as around 1.2×10^{-6} and 1.0×10^{-6}, respectively. As with Case 1, only the SDIs of cables attached to pylons are falsely identified as damaged.

In Case 3, only the SDIs of cables increase as the damage occurs. The SDI of the damaged cable is estimated as around 1.0×10^{-5}, which is much higher than the result in previous cases because of its higher flexibility than other elements. It is noted that only the SDIs of cables close to the damaged cable are inaccurately identified as damaged.

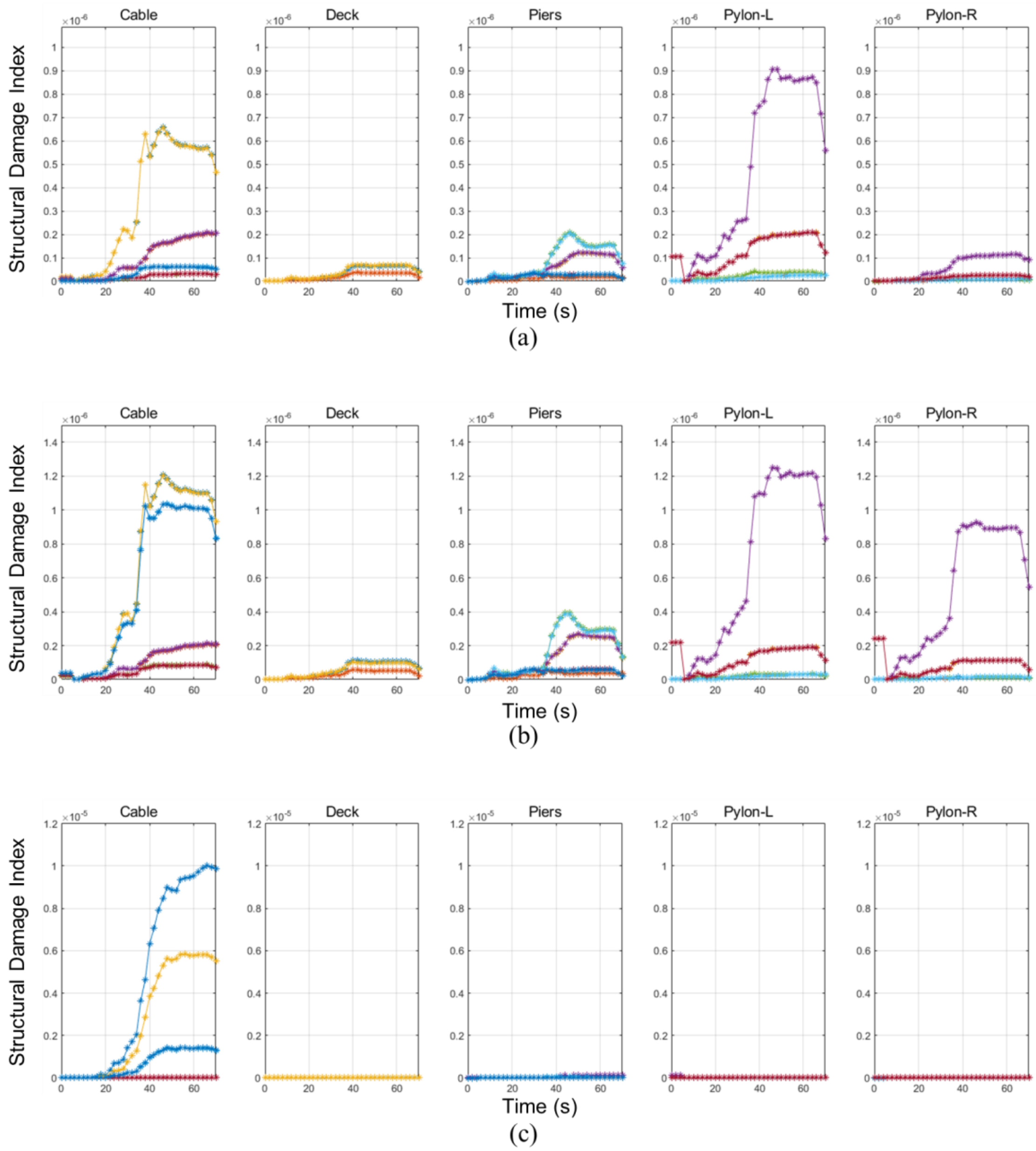


Figure 7 Results of near-real-time damage identification with damage on (a) Py-L; (b) Py-L and Py-R; and (c) a single cable.

5. CONCLUSIONS

This paper proposed a new DNN-based framework for identifying near-real-time seismic damage of complex infrastructure systems. The CVAE model was trained to construct a disentangled latent space of the sample covariance matrix of the measured structural responses identified from the undamaged structure. In near-real-time identification, the undamaged covariance matrix, which is the most similar to the seismic response data, is extracted from the database by measuring the distance in the latent space. The SDI based on the difference in covariance matrices was introduced, and the damage was identified by the SDI in near-real-time. A numerical example of the real-time simulation was provided to test and demonstrate the proposed framework. We selected the Incheon Grand Bridge as the target structure to verify the proposed framework. The framework successfully identified damage under seismic load conditions in near-real-time. The robust performance of the proposed method under seismic load conditions is expected to help reduce the time required for the post-disaster decision-making process. Eventually, the proposed framework will be utilized to prepare effective post-disaster operational and maintenance strategies.

ACKNOWLEDGEMENTS

The authors are supported by the project "Deep Learning Technologies for Assessment of Seismic Responses and Damage of Nuclear Power Plant Structures and Equipment" of the Ministry of Science and ICT (MSIT) of the Korean Government (Grant No. RS-2022-00144434).

REFERENCES

Chiou, B., Darragh, R., Gregor, N. and Silva, W. 2008. NGA project strong-motion database. *Earthquake Spe*ctra, 24(1): 23-44.

Dawson, B. 1976. "Vibration condition monitoring techniques for rotating machinery." *The shock and vibration digest*, 8(12): 3.

Goodfellow, I., Bengio, Y. and Courville, A. 2016. *Deep learning*. Cambridge: MIT press.

He, K., Zhang, X., Ren, S. and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770-778.

Kim, M. and Song, J. 2022. Near-real-time identification of seismic damage using unsupervised deep neural network. *Journal of Engineering Mechanics*, 148(3): 04022006.

Kim, T., Kwon, O. S. and Song, J. 2021. Seismic performance of a Long-Span Cable-stayed bridge under spatially varying bidirectional spectrum-compatible ground motions. *Journal of Structural Engineering*, 147(4): 04021015.

Kingma, D.P. and Welling, M. 2013. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114*.

Kingma, D.P. and Ba, J. 2014. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980*.

McKenna, F., Scott, M. H. and Fenves, G. L. 2010. Nonlinear finite-element analysis software architecture using object composition. *Journal of Computing in Civil Engineering*, 24(1): 95-107.

Pathirage, C.S.N., Li, J., Li, L., Hao, H., Liu, W. and Ni, P. 2018. "Structural damage identification based on autoencoder neural networks and deep learning." *Engineering Structures*, 172: 13-28.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y. and Manzagol, P.A. 2010. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." *Journal of machine learning research*, 11(Dec): 3371-3408.

Zhu, M., McKenna, F. and Scott, M. H. 2018. OpenSeesPy: Python library for the OpenSees finite element framework. *SoftwareX*, 7: 6-11.