Transfer-AE: A novel Autoencoder-based Impact Detection Model for Structural Digital Twin

Chengjia Han[1], Zixin Wang[2], Yuguang Fu[1,*], Shirley Dyke[2], Adnan Shahriar[3]

[1] *School of Civil and Environmental Engineering, Nanyang Technological University, Singapore, 639798*

[2] *Lyles School of Civil Engineering, Purdue University, West Lafayette, IN 47907, USA*

[3] *L&T Technology Services, Plano, TX 75093, USA*

* Corresponding author: Yuguang Fu, *yuguang.fu@ntu.edu.sg*

Abstract: Accurately detecting the location and intensity of impacts is crucial for ensuring structural safety. Currently, AI-based structural impact detection methods are widely used for their excellent detection accuracy. However, their generalization capability is limited by the scenarios present in the training data. Many complex and dangerous impact scenarios are difficult to conduct real-world experiments on to collect sufficient samples. To capture all impact scenarios and fully leverage the advantages of AI-based detection technologies, advanced methods involve combining real-world structural monitoring data with corresponding numerical models to construct digital twins. These methods continuously refine the created numerical models with limited real-world data and provide diverse impact scenarios through numerical model simulations. However, there are inevitable differences between digital models and physical models that are challenging to correct through mechanical means. This discrepancy in data distribution between the two models significantly hinders the application of digital twin technology in impact/event identification tasks. To address this challenge, this study proposes a novel model based on autoencoders, named Transfer-AE. Transfer-AE encodes the common features of digital twins in the latent space to bridge the uncertainty gap at a macro scale between numerical models and physical models and synchronously fits the magnitude and location of the impact load in the decoder. This enables consistent detection results for the same impact event, whether the sample comes from the numerical model or the physical model. Transfer-AE includes two operating modes: Mode 1 has a fixed computational complexity with stable inference speed, but the training cost and difficulty increase with data distribution. Mode 2's computational complexity increases with data distribution, but it has a fixed training cost and speed. In both cases involving the geodesic dome structure simulating a deep space habitat and the IASC-ASCE benchmark structure, Transfer-AE demonstrated the best performance in impact localization and quantification tasks compared to mainstream domain-adaptive transfer models.

Keywords: Digital Twin, Impact quantification, Impact localization, Transfer learning, Autoencoder, Generative network, Domain adaptive.

---

1. Introduction

When a structure is subjected to an impact, it is crucial for the safety monitoring of the structure to analyze the mechanical response signals of the structure to determine the magnitude of the impact and its specific location on the structure [1-4]. The capture of a structure mechanical response is typically achieved through sensors installed on the structure, such as accelerometers, fiber Bragg gratings, and distributed acoustic sensors, among others [5-8]. These devices can collect physical signals such as acceleration, strain, or vibration when the structure is impacted. In practice, the task of detecting and locating impact forces involves using technical methods to establish a mapping between the physical signals characterizing the structure response and the magnitude and location of the impact force [9-10]. Currently, many scholars have conducted related research based on mechanical analysis, data mining or artificial intelligence [11-13]. However, relying solely on measured structural responses for impact detection is limited [14,15]. For scenarios where it is difficult to obtain sufficient monitoring data from real structures, as well as in situations that are challenging to fully account for in reality, conducting numerical simulations to supplement real monitoring data with more scenarios is a widely accepted solution [16,17]. Currently, there is limited research on applying digital twins to impact detection tasks. Therefore, we referred to digital twin tasks in structural health. Many scholars are assisting the identification of structural impacts through the results of numerical simulations using finite elements. Furthermore, they are researching the correlation between numerical models in digital space and physical structures in actual physical space through monitoring data, thereby constructing digital twins [18,19]. This type of methods aims to enhance the reliability and accuracy of structural health monitoring tasks. However, for state-of-the-art digital twin technologies, there is an unavoidable systematic error between numerical models in digital space and physical structures in real space [20], which stems from the differences in complex impact/boundary conditions that are difficult to describe accurately, epistemic uncertainty, differences in material constitutions, and computational errors inherent to the computational methods themselves. Systematic errors are pervasive and challenging to overcome, directly leading to inconsistencies between the simulation results of numerical models and the actual measurement results of physical structures when describing the same event [21]. Some scholars have proposed technical routes that domain adaptation (DA), which mitigate this difference by transferring data from different distributions to the same domain through transfer learning [22]. Despite this, in the current impact detection tasks for complex structures, the statistical analysis-based DA methods lack robustness, while the GANs-based DA methods are challenging to train [23,24]. Especially for the task of impact identification and localization in structures, there is currently a lack of a reliable and general model to effectively bridge the gap between numerical models and physical structures. This would enable digital twins of structures to fully leverage their advantage in simulating diverse impact scenarios and achieve high-precision impact identification and localization on physical structures.

1.1 motivation

Despite the systemic errors between the numerical model and the physical model that constitute the digital twin of a structure, which manifest as distribution discrepancies in the acceleration signals detected/calculated across multiple degrees of freedom for the same impact event, the signals in the digital space and the real physical space still maintain a macroscopic similarity. This is because they both reflect the same impact event governed by identical physical laws. This implies that we can reduce the dimensionality of the differently distributed signals, minimizing feature dimensions until, in a certain low-dimensional latent space, the numerical signals and physical signals responding to the same impact event achieve a high degree of similarity. This process retains sufficient effective information to fit the impact location and magnitude. In response to the above issue, this research has introduced the Transfer-AE model, aimed at eliminating the systematic errors between numerical and physical structures. Our approach can also be classified as a domain-adaptive transfer learning method. Unlike mainstream GAN-based DA models, our model uses an autoencoder as its framework. Proposed Transfer-AE model ensures that structural response data from numerical and physical structures of the same impact event can achieve consistent predictions of location and magnitude.

1.2 Contributions

The contributions of the research are as follows:

- A novel generative transfer learning paradigm based on autoencoders is proposed, which effectively achieving adaptive alignment of systematic errors between finite element models and physical structures in structural impact detection tasks. It means that the diverse data from numerical models can be used to enhance the task model's generalization ability in detecting the impact location and magnitude on real structures. Based on this, the network structure of Transfer-AE was designed, which couples the tasks of impact localization and impact quantification, optimizes the structural hyperparameters and types of feature extraction layers, and establishes the most suitable network structure for impact localization and quantification tasks.
- Based on the proposed transfer learning paradigm and the Transfer-AE structural design, two different Transfer-AE models were developed, both undergoing Phase 1 pre-training on finite element data. In Phase 2, Model 1 employs a fixed network structure, which has the advantage of not increasing in size with the introduction of new distribution data. However, it requires sampling of some old distribution data, making the training process more costly. Model 2 uses fixed training data, which has the benefit of lower training costs when new distribution data are introduced, but it necessitates the addition of an extra encoder module to the network.
- In the case studies of impact localization and quantification for the 3D geodesic dome structure and the IASC-ASCE benchmark structure, the proposed Transfer-AE model achieved the best performance compared to mainstream DA-based models.

1.3 Organization

This study is organized as follows. Section 2 provides a literature review of previous research. Section 3 presents the principles of Transfer-AE and the two Transfer-AE modes designed for impact detection problems. Section 4 presents the performance of the proposed Transfer-AE model on a 3D geodesic dome structure and the IASC-ASCE benchmark structure. Section 5 discusses the significance and applications of the proposed Transfer-AE in digital twins, based on the results of the case study. Finally, the conclusions of the study are presented in the last section, summarizing the key contributions, and outlining potential directions for future research. The abbreviations of technical terms used in this study are summarized in **Table 1**.

Table 1. List of used abbreviations.

Full Forms	Abbreviations
Autoencoder	AE
Mean Absolute Error	MAE
Mean Absolute Percentage Error	MAPE
Convolutional Neural Network	CNN
Long Short-Term Memory	LSTM
Multi-Head Attention	MHA
Domain Adaptation	DA
Generative Adversarial Network	GAN
Degrees of Freedom	DOF
Finite Element	FE
Fiber Bragg Grating	FBG

2. Literature Review

2.1 Research on structural impact detection and location models

Xu proposed impact identification for aluminum plate structures based on the least squares support vector machine, extracting two features from each sensor signal through Hilbert transform, and using the least squares support vector machine to establish an intelligent model [25]. Tabian et al. reported on an impact identification and localization model for complex composite structures based on convolutional neural networks. It converts ultrasonic signals recorded by piezoelectric sensors into 2D images and establishes a mapping to the magnitude and location of impacts [26]. Sai et al. developed a collision localization system based on a FBG sensor network, which processes FBG signals for collision detection and localization using an Extreme Learning Machine [27]. Tan et al. discussed various existing methods for optimizing sensor placement to help collect higher quality structural response data for structural health diagnosis [28]. Huang et al. proposed a novel impact force identification method named Distance-assisted Graph Neural Network (DAGNN), which incorporates the physical distance between the impact location and sensors as a physical constraint into the training process, achieving high-precision collision localization [29].

2.2 Research on digital twin-based structural health monitoring methods

Seventekidis et al. proposed a novel structural health monitoring method where labeled damaged data is generated through FE models for a pin-joint composite truss structure, adopting a model-based approach as an alternative to vibration measurements to solve the data acquisition problem [30]. Zhang et al. attempted to combine the advantages of data-driven and physics-based structural health monitoring methods through physics-guided machine learning, which extended the original modal-property based features with the damage identification result of finite element model updating, aiming to enhance damage detection performance [31]. Colombo et al. proposed a structural damage identification method based on the inverse finite element method (iFEM) and proposes a superimposition of effects approach to weigh the contribution of different basic models in simulating the real behavior of the structure [32]. Massari et al. proposed a design and calibration method based on numerical FE method modeling for developing soft tactile sensors capable of simultaneously determining the magnitude and application location of normal loads imposed on their surface [33].

2.3 Research on DA-based structural health monitoring models

Poole et al. proposed a DA method based on Statistical Alignment and demonstrated how to make these methods robust to class imbalance [34]. Giglioni et al. proposed a domain adaptation-based transfer learning method that utilizes information from fully labelled bridge structure data to accurately predict new instances in unknown target domains [35]. Besides statistical analysis-based DA, in the field of deep learning, Ge et al. proposed an unsupervised domain adaptation method based on CycleGAN that incorporates physical constraints and self-attention mechanisms [36]. Luleci et al. developed a variant of Generative Adversarial Networks, a generative model named Cycle-Consistent Wasserstein Deep Convolutional GAN with Gradient Penalty (CycleWDCGAN-GP) for undamaged-to-damaged domain translation [37]. We have summarized the objectives, domains, advantages, and limitations of related previous research in **Table 2**.

Table 2. Summary of related previous works

Research	Objective	Domain	Advantages	Limitations
Xu [25]	Impact identification for aluminum plate structures	Structural Impact Detection	Intelligent model using least squares support vector machine	Only applicable to measured data, does not involve model discrepancy issues.
Tabian [26]	Impact identification and localization in composite structures	Structural Impact Detection	Uses convolutional neural networks, converts signals to 2D images	Only applicable to measured data, does not involve model discrepancy issues.
Sai [27]	Collision localization using FBG sensor network	Structural Impact Detection	Utilizes Extreme Learning Machine for processing signals	Only applicable to measured data, does not involve model discrepancy issues.
Tan [28]	Optimize sensor placement for structural health diagnosis	Structural Health Diagnosis	Helps collect higher quality data	Only applicable to measured data, does not involve model discrepancy issues.
Huang [29]	High-precision collision	Structural	Incorporates physical	Only applicable to

	localization	Impact Detection	distance as a constraint in Graph Neural Network	measured data, does not involve model discrepancy issues.
Seventekidis [30]	Structural health monitoring using FE models	Structural Health Monitoring	Generates labeled damaged data through FE models	Not address the response gap between FE models and real structures.
Zhang [31]	Combine data-driven and physics-based methods	Structural Health Monitoring	Enhances damage detection by extending modal-property based features	Designed for damage tasks, performance on impact detection tasks is unknown.
Colombo [32]	Structural damage identification using iFEM	Structural Health Monitoring	Weighs contributions of different models for real behavior simulation	Not address the response gap between FE models and real structures.
Massari [33]	Design and calibration of soft tactile sensors	Structural Health Monitoring	Simultaneously determines magnitude and location of loads	Not address the response gap between FE models and real structures.
Poole [34]	DA method with Statistical Alignment	DA	Robust to class imbalance	Effect on large datasets is unknown, and performance on impact monitoring tasks is also unknown.
Giglioni [35]	Domain adaptation-based transfer learning for bridge structures	DA	Accurate predictions in unknown target domains	Designed for damage tasks, performance on impact detection tasks is unknown.
Ge [36]	Unsupervised DA using CycleGAN	DA	Incorporates physical constraints and self-attention mechanisms	Focuses on general signal domain translation, not specific to structural monitoring tasks.
Luleci [37]	Domain translation using CycleWDCGAN-GP	DA	Effective undamaged-to-damaged domain translation	Designed for damage tasks, performance on impact detection tasks is unknown.

3. Methodology

3.1 Principle of Transfer-AE model

Addressing the issue of impact detection for structures, it is feasible to design an artificial network model that fits the mapping between the response signals of a structure when impacted and the impact. However, due to the systematic errors between numerical simulations and real physical structures mentioned in the introduction, there exists a distribution difference between the structural responses in digital space under the same impact and those in real space. This distribution difference is complex and difficult to directly eliminate through differentiation. However, despite the differences in distribution, the signals from digital space and those from real physical space still maintain macroscopic similarity, as they reflect the same impact event governed by the same physical laws. Therefore, it is precisely because of this similarity, based on the law of manifold distribution that high-dimensional data of the same category tends to concentrate around a low-dimensional manifold [38]. It means that if two sets of data reflect the same event but have distribution differences, they will exhibit consistent trend features and divergent detail features in high-dimensional space. When compressed to a low-dimensional space, the signal details gradually diminish, and the trend features become more pronounced. If we find a maximal low-dimensional space where all divergent detail features are lost, and all consistent trend features are preserved, then the two sets of signals, which have systematic errors in high-dimensional space, will become indistinguishable in this low-dimensional space. Therefore, for our research, there exists a certain low-dimensional space, in where the numerical signals and physical signals responding to the same impact event can achieve a high degree of similarity. Furthermore, the evaluation of similarity is not based on divergence or spatial distance metrics but rather on an assessment from the perspective of the task model. That is, when a latent space is identified that can map the finite element model signals and physical structure signals from the same event in such a way that the task model, from the perspective of impact detection, judges them to be sufficiently similar to produce consistent identification results. For example, in the research by Ge et al. [36], the objective of domain translation is to ensure that the signals in the target domain are as similar as possible to the signals in the current domain after conversion. In contrast, our research does not require the signals to be sufficiently similar in geometric distance because our model is task-oriented. We only require the signals to be sufficiently similar from the perspective of the task model. Therefore, we discarded the norm distance loss between the decoder's reconstructed signal and the input signal. This approach helps to increase model redundancy and improve generalization capability. This research utilizes autoencoders to find a latent space [39], which essentially can be considered a DA-based transfer learning paradigm achieving the task transfer of the network in digital twins between digital and physical spaces, as shown in **Figure 1**. For DA model, it is important to introduce two key objects, a domain and task. A domain $D=\{\chi, P(X)\}$ is defined by of a tensor space χ and a marginal probability distribution $P(X)$, while the corresponding task $\Gamma=\{Y, f(\cdot)\}$, where Y is the label space and the $f(\cdot)$ is the mapping function learnt from a finite sample $\{x_i, y_i\}_{i=1}^{n}$, where $x_i \in \chi$, $y_i \in Y$ [40]. However, Unlike traditional DA method, in our research, our source domain is the physical structure $D_P=\{S_{P,i}, (L_{P,i}, F_{P,i})\}_{i=1}^{n_P}$ and the FE model $D_F=\{S_{F,i}, (L_{F,i}, F_{F,i})\}_{i=1}^{n_F}$, where S is the structure response, F is the impact magnitude and L is the impact position. The target domain is the latent Space $D_L=\{S_{L,i}, (L_{L,i}, F_{L,i})\}_{i=1}^{n_L}$. Our objective is to design a neural network mapping that identifies a set D_L such that the union of the label spaces of D_S and D_F becomes a subset of the label space of D_L, that $(Y_S \cap Y_F) \subset Y_L$.

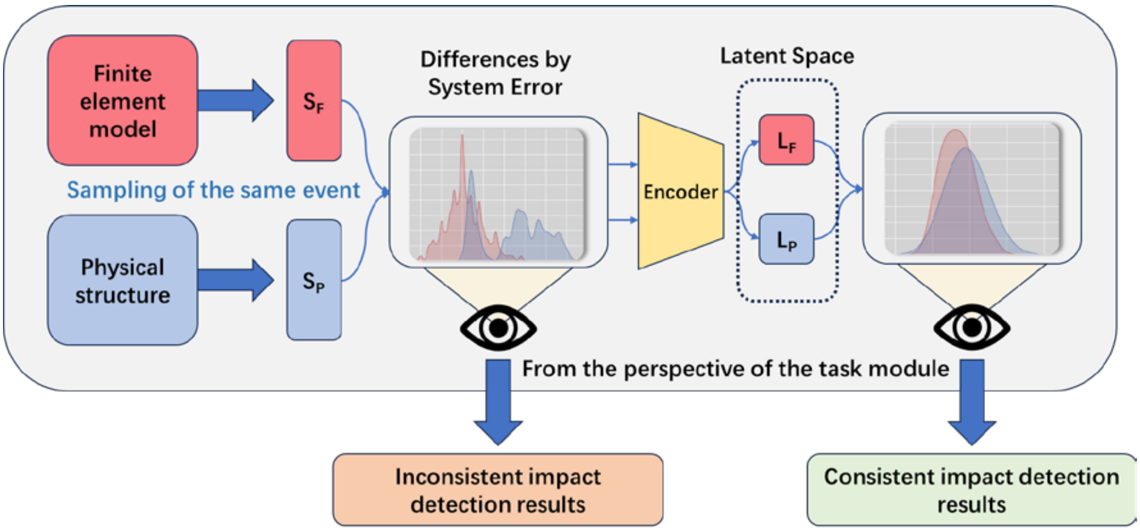


Figure 1. Diagram of the transfer learning paradigm based on autoencoders.

Based on the proposed transfer learning paradigm, the study established the Transfer-AE network structure as shown in **Figure 2**. The Transfer-AE structure consists of three sequential parts: an Encoder module for mapping high-dimensional features to a low-dimensional latent space, a Decoder module for restoring the low-dimensional latent space features back to the high-dimensional space, and a Task module for synchronously identifying the location and magnitude of impact based on the restored high-dimensional signal.

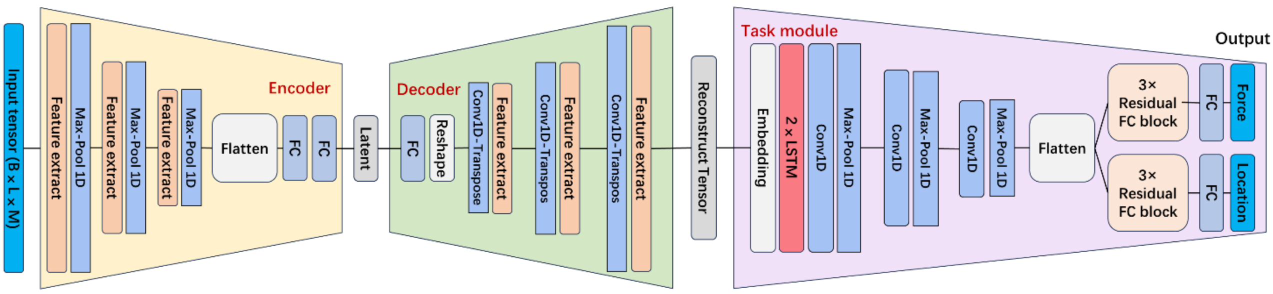


Figure 2. The structure of the Transfer-AE model.

The input to the Transfer-AE model is the structural force response, denoted as tensor $S \in \mathbb{R}^{B \times L \times M}$, where B is the batch size, L is the length of response time series and M is the number of sensors on structure. After entering the Encoder, tensor S will pass through three feature extraction layers and three maximum pooling layers with a kernel size of k_M. The feature extraction layers can adopt different structures, such as convolutional units, recurrent neural network units, or attention units etc. For the downsampling block, after the tensor S passes through a downsampling block with a 1-D convolutional feature extraction layer, the output tensor S_d can be calculated as [41],

$$S_d(t) = (x * w)(t) = \sum_{i=0}^{k-1} S(t-i)w(i), \tag{1}$$

where $w(i)$ is the convolution kernel with a length of k, and t is the time step/position in tensor. For LSTM units, the downing sampling block can be calculated as [42],

$$
\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
o_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_o) \\
h_t &= o_t \cdot \tanh(f_t \cdot C_{t-1} + i_t \cdot \tanh(W_C \cdot [h_{t-1}, x_t] + b_C))
\end{aligned}
\quad , \tag{2}
$$

where h_t is the hidden state of the t-th time step/position in tensor, the f_t, i_t and o_t are the forget gate, input gate, and output gate, respectively. σ the sigmoid activation function, W is the weight matrix, and b is the bias vector.

For MHA units, the downing sampling block can be calculated as [43],

$$
\begin{aligned}
z_1 &= \mathrm{concat}(Attention(Q_n, K_n, V_n)) \\
&= \mathrm{concat}(softmax(\frac{Q_n K_n^T}{\sqrt{d}})V_n)
\end{aligned}
\quad , \tag{3a}
$$

with

$$
softmax(x) = \frac{1}{1+e^{-x}} . \tag{3b}
$$

where, query $Q_{\mathrm{n}} \in \mathbb{R}^{B \times L \times d}$, key $K_n \in \mathbb{R}^{B \times L \times d}$ and value $V_{\mathrm{n}} \in \mathbb{R}^{B \times L \times d}$ are the matrices split by input tensor S in the second dimension, n is the number of divisions ($d \times n = M$). Then, tensor z_1 is directly added to S and normalize together to obtain $z_2 \in \mathbb{R}^{B \times L \times M}$. The tensor z_2 passes through a feed-forward layer ($FFL(\cdot)$ function) to obtain $S_d \in \mathbb{R}^{B \times L \times M}$, which is calculated as,

$$
S_d = FFL(Z_2) = ReLU(z_2 w^1 + b^1) w^2 + b^2 \tag{3c}
$$

with

$$
ReLU(x) = \max(0, x) . \tag{3d}
$$

After we get the extracted feature S_d, we use max pooling to reduce the dimensionality of the tensor, get tensor S_{small} as,

$$
S_{\mathrm{small}}(i) = \max(S_d^{i \cdot k_M}, S_d^{i \cdot k_M + 1}, \ldots, S_d^{i \cdot k_M + k_M - 1}) , \tag{4}
$$

where the pooling window size and stride are both k_M. After going through three downing sampling blocks, the length of the output tensor is reduced to L/k_{M}^3. Subsequently, the tensor is flattened and passes through two fully connected layers with P units to adjust its size, resulting in the latent tensor $L \in \mathbb{R}^{B \times P}$ which is the input to the Decoder module. The Decoder module performs mirror operations to those of the Encoder module, also utilizing three feature extraction modules identical to those in the Encoder, along with three 1-D transposed convolutions to upsample the tensor L, obtaining a reconstructed tensor with the same dimensions as the input features, denoted as $R \in \mathbb{R}^{B \times L \times M}$.

After that, the reconstructed tensor R serves as the input to the Task module and add positional encoding for embedding which helps prevent the model from repeatedly using the same input to generate outputs, thereby enhancing the diversity and generalization ability. The embedding is calculated as [43],

$$
\begin{aligned}
&PE(pos, 2i) = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \\
&PE(pos, 2i+1) = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \\
&R_e(\in \mathrm{R}^{B\times P\times M}) = R(\in \mathrm{R}^{B\times P\times M}) + \mathrm{PE}(R(\in \mathrm{R}^{B\times P\times M}))
\end{aligned}
\quad , \tag{5}
$$

where $PE(\cdot)$ is positional encoding function, the *pos* is the number at each position in tensor R, and the output R_e has the same size with input R. After that, R_e will go through a combination of LSTM attention units [44] and 1-dimensional convolutional units [45]. After flattening the features, two separate residual fully connected blocks independently output the required impact location and quantification.

It should be noted that although the initial part of the Transfer-AE network is an autoencoder module, the network training does not require the reconstructed tensor R to be similar to the input tensor S [46]. In fact, to enhance the generalizability of network, this constraint has been reduced. Instead, the focus is on obtaining a reconstructed tensor R that is more suited to the Task module. This corresponds to our earlier description of similarity between numerical signal and physical response, where the required similarity is considered from the perspective of the Task module, rather than a direct measure of distance. Based on the foundational structure of Transfer-AE, the study proposes two modes for data transfer between numerical and physical spaces, referred to as Transfer-AE (mode1) and Transfer-AE (mode2), and each has its own advantages.

2.2 Transfer-AE (mode1)

Transfer-AE (mode1) uses a fixed network structure, and its training process is divided into two stages, as shown in **Figure 3**. In Training Stage 1, the network is trained solely with finite element data, utilizing structural responses sampled from the finite element model under different impact events (a combination of the magnitude of the force (denoted as F_f) and the impact location (denoted as F_l)). Generally, finite element models can generate a large number of samples in batches, while the number and types of samples from real physical structures are limited. Therefore, in this study, we conduct Phase 1 pre-training with the more abundant finite element samples. The Transfer-AE network outputs the predicted magnitude of the force (denoted as F_f^P) and location (denoted as F_l^P). The total loss function is denoted as L_{mode1}, which is consists of two parts: the prediction loss for force and the prediction loss for position. The prediction of force uses MAE loss [47], while the prediction of position uses multi-class cross-entropy loss [48]. The L_{mode1} is calculated as

$$
L_{\mathrm{mode1}} = w_1 \cdot \frac{1}{n}\sum_{i=1}^{n}\left|F_{f-i} - F_{f-i}^{P}\right| - w_2 \cdot \frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{M} F_{l-ic}\log(F_{l-ic}^{P}) , \tag{6}
$$

where n is the number of forces, w_1 is the weight for the first term of the loss, M is the number of all possible classifications of position, and w_2 as the weight for the second term of the loss.

After completing Training Stage 1, we introduce real physical structure data with a different distribution from the finite element data for fusion training. Training Stage 2 is conducted alternately on two datasets with different distributions, with the loss function calculated in the same way as in Stage 1. We first freeze the network parameters of the Decoder module and Task module, only updating the network parameters of the Encoder module, and train the Encoder module using sampled response data from the physical structure model. To ensure that Transfer-AE (mode1) remains applicable to the original dataset (finite element data), after updating the Encoder with physical structure data a few times, we need to randomly sample a dataset of the same size as the physical structure training data from the original dataset (finite element data) and train the Encoder module. The Encoder is alternately trained with physical data and finite element data until the loss of Transfer-AE converges. The reason we freeze the Decoder and Task modules and alternately train the weights of the Encoder is to find the union of the projection domains of the two different domains in the low-dimensional latent space. After dimensionality reduction by the Encoder, the feature projection domain from the FE model and the feature projection domain from the real model form two intersecting sets in the low-dimensional space. Alternating training controls the learning rate, allowing the Encoder to extend the boundary to the real model domain (enhancing generalization ability) while continuously reinforcing the original boundary of the FE domain. This process continues until the Encoder identifies the entire projection domain of both the FE domain and the real model domain. The pseudocode for the training process of Transfer-AE (mode1) is as follows:

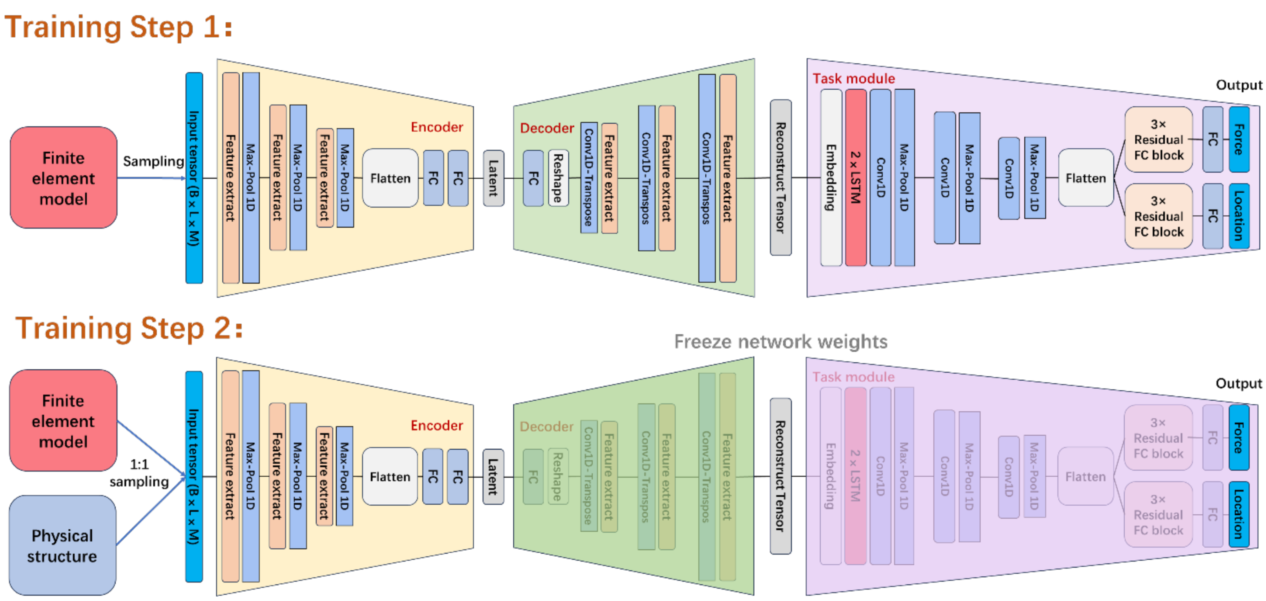


Figure 3. The training process of Transfer-AE (mode1)

Algorithm: The training process of Transfer-AE (mode1)

Input: Finite element dataset: ***F***, physical structure dataset: ***P***, Transfer-AE with initial parameters: *Model*

Output: Trained Transfer-AE: $Model_f$

Training Stage 1

For *epoch* **in** (Training Epochs=100):
  Do:
    Randomly sample B samples from $\boldsymbol{F}$ and train *Model*
  While: traversal of $\boldsymbol{F}$, Obtain the updated model $Model_{\mathrm{s1}} \leftarrow Model$

Training Stage 2

Freeze the Decoder and Task module: $Model_{\mathrm{sF}} \leftarrow Model_{\mathrm{s1}}$
For *epoch* **in** (Training Epochs=100):
  Randomly sample from $\boldsymbol{F}$ to construct a dataset $\boldsymbol{F_p}$ of the same scale as $\boldsymbol{P}$
  For epochs **in** (Training Epochs=5):
    Do:
      Randomly sample B samples from $\boldsymbol{P}$ and train $Model_{\mathrm{sF}}$
    While: traversal of $\boldsymbol{P}$
  For epochs **in** (Training Epochs=5):
    Do:
      Randomly sample B samples from $\boldsymbol{F_p}$ and train $Model_{\mathrm{sF}}$
    While: traversal of $\boldsymbol{F_p}$
Return the trained model $Model_f \leftarrow Model_{\mathrm{sF}}$

2.3 Transfer-AE (mode2)

Compared to Transfer-AE (mode1), the training of Transfer-AE (mode2) is also divided into two stages, but it no longer uses a fixed network structure, as shown in **Figure 4**. The Training Stage 1 of Transfer-AE (mode2) is the same as that of Transfer-AE (mode1), where it uses finite element data to train the Transfer-AE structure, thus fitting the mapping between the structure response and the impact in the numerical space. However, in Training Stage 2 of Transfer-AE (mode2), we only use sampled data from real physical structures for training. To ensure the applicability of Transfer-AE (mode2) on the original dataset, we freeze all the parameters of the network trained in Stage 1 and duplicate the Encoder structure and training parameters from Stage 1, denoted as Encoder-delta. Response signals from the physical structure are dimensionally reduced by the Encoder module to obtain the latent variable L_1, and similarly reduced by the Encoder-delta module to obtain the latent variable L_2. By adding L_1 and L_2 together, we obtain the complete latent variable L, which then enters the subsequent Decoder and Task modules for the quantification and localization of impact forces. The reason for this operation is that the mapping recorded by the Encoder module is tailored to the original dataset features (FE data), so the L_1 obtained from the new distribution dataset through the Encoder module has an offset. According to the Generalization Bounds theory [49] that the generalization error on the target domain can be bounded by the generalization error on the source domain plus a discrepancy term between the source and target domains, as:

$$\epsilon_t(h) \le \epsilon_s(h) + \frac{1}{2} d_{H\Delta H}(D_s, D_t) + \lambda \,, \tag{7}$$

where, $\epsilon_t(h)$ is the target error, $\epsilon_s(h)$ is the source error, $d_{H\Delta H}$ is the domain discrepancy measure, and λ is a term representing the complexity of the hypothesis class H. Therefore, theoretically, the newly added Encoder-delta's function is to generate a residual which related to the discrepancy between the domains and the complexity of the hypothesis space, to compensate for this offset between the two domains. It should be noted that when using Transfer-AE (mode2), the response from numerical models still employs the network from Training Stage 1 for the quantification and localization of impact, and only for responses from real physical structures is an additional module added.

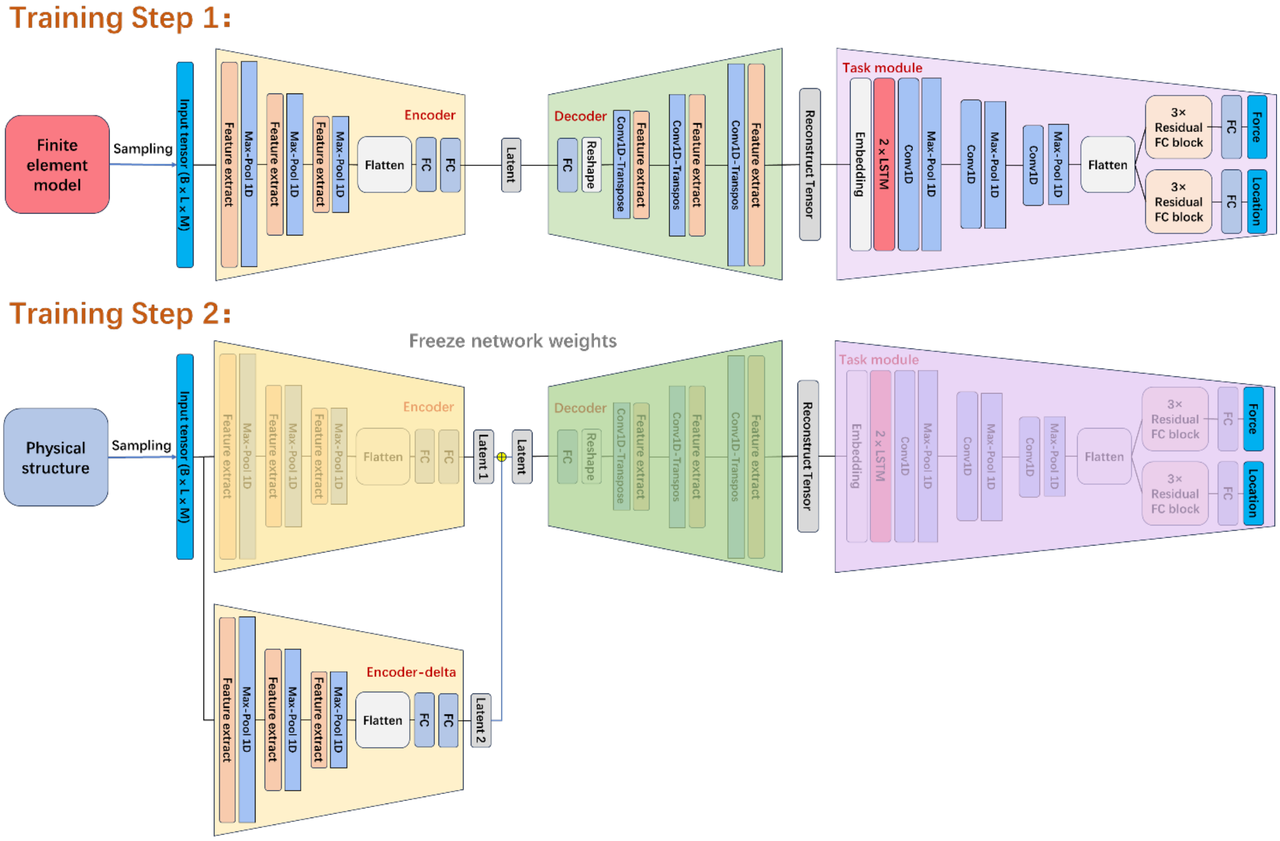


Figure 4. The training process of Transfer-AE (mode2)

The pseudocode for the training process of Transfer-AE (mode2) is as follows:

Algorithm: The training process of Transfer-AE (mode2)

Input: Finite element dataset: ***F***, physical structure dataset: ***P***, Transfer-AE with initial parameters: *Model*

Output: Trained Transfer-AE: $Model_f$

Training Stage 1

For *epoch* **in** (Training Epochs=100):
  Do:
    Randomly sample *B* samples from ***F*** and train *Model*
  While: traversal of ***F***, Obtain the updated model $Model_{s1} \leftarrow Model$

Training Stage 2

Freeze the Transfer-AE: $Model_{sF} \leftarrow Model_{s1}$, Copy Encoder module from $Model_{s1}$ as E_{delt}
Build new model $Model_{s2} \leftarrow Model_{sF} + E_{delt}$
For *epoch* **in** (Training Epochs=100):
    Do:
      Randomly sample *B* samples from ***P*** and train $Model_{s2}$
    While: traversal of ***P***
Return the trained model $Model_f \leftarrow Model_{s2}$

Overall, both Transfer-AE (mode1) and Transfer-AE (mode2) have achieved task transfer between numerical and physical structures. Although their Training Stage 1 is consistent, however in training stage 2, the training cost (in terms of sample size) for mode1 is twice that of mode2, while the model size for mode1 is fixed and does not increase the computational complexity of the inference process. The performance of both will be tested in the following section.

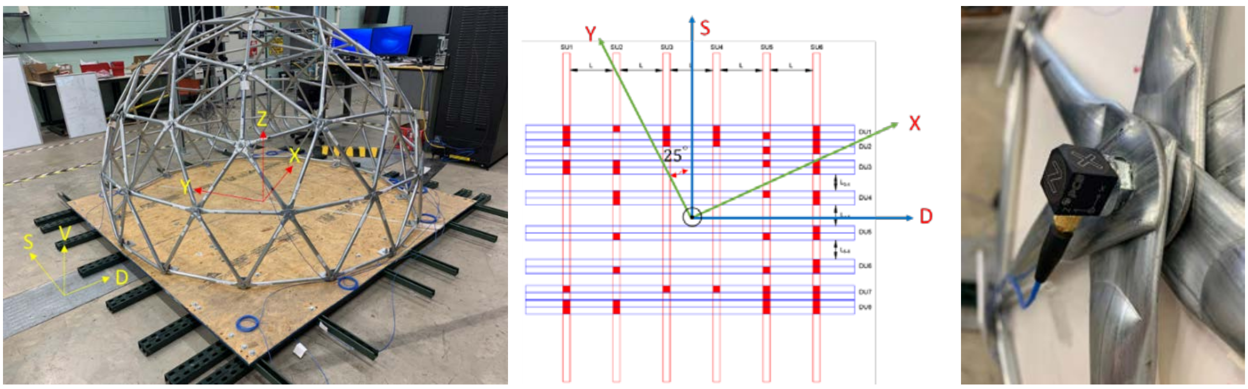
4. Case Study

4.1 Impact localization and quantification test on the 3D geodesic dome structure

In this section, we will take the impact detection problem faced by a 3D geodesic dome structure as an example. This study is part of the resilient extraterrestrial habitats initiative, where the establishment of a 3D geodesic dome structure for impact simulation is aimed at researching fault detection and diagnostic strategies for deep space human habitats under the influence of meteoroid impacts. A 3D geodesic dome structure is a spherical space frame structure composed of a complex network of triangles, which can distribute structural stress throughout the entire structure, enabling it to withstand impacts that are quite heavy relative to its size, making it very suitable for harsh environments [50]. When a 3D geodesic dome structure is subjected to an external impact, we want to determine the location of the impact and the magnitude of the impact based on the structure response. Firstly, we constructed a scaled-down model of a 3D geodesic dome structure in the laboratory.

4.1.1 3D geodesic dome structure: physical structure test

The 3D geodesic dome structure model constructed in the laboratory is shown in **Figure 5**. It displays its geometric shape, support structure, and the accelerometers installed at the nodes for monitoring the structural response. **Figure 5(a)** shows the dome structure model constructed with rods and nodes in the laboratory. The geodesic dome has a height of 1.5 meters and a diameter of 2.5 meters. It comprises 61 nodes and 165 members, forming the skeleton of the dome structure. The dome structure is fixed on a wooden board supported by a frame. The red coordinate system represents the local coordinate system of the dome model, while the yellow coordinate system represents the global coordinate system based on the grid support. **Figure 5(b)** shows the overall geometry information of the support. The global coordinate system of base as ***D-S-V*** and the global coordinate system of dome as ***X-Y-Z***. The planar angle between the two coordinate systems is 25 degrees. Each strut of support frame is considered a beam element capable of bearing both axial and bending impacts, with each node connecting up to 6 members. **Figure 5(c)** shows the accelerometer installed at the nodes of the dome, with the Z-axis arranged perpendicularly outward to the node surface, to collect the three-axis acceleration signals when the dome is impacted.



(a) dome sturctue in lab (b) frame support (c) accelerometer

Figure 5. Physical structure of 3D geodesic dome structure.

We numbered the nodes and struts of the geodesic dome structure, as shown in **Figure 6**. The dome structure can be disassembled from the ground up into a combination of triangles arranged in five layers. We installed accelerometers at 11 nodes, specifically H61, H54, H51, H47, H45, H41, H37, H34, H29, H24, and H18 (highlighted as bright spots in **Figure 6**), to comprehensively characterize the response of the geodesic dome from various angles.

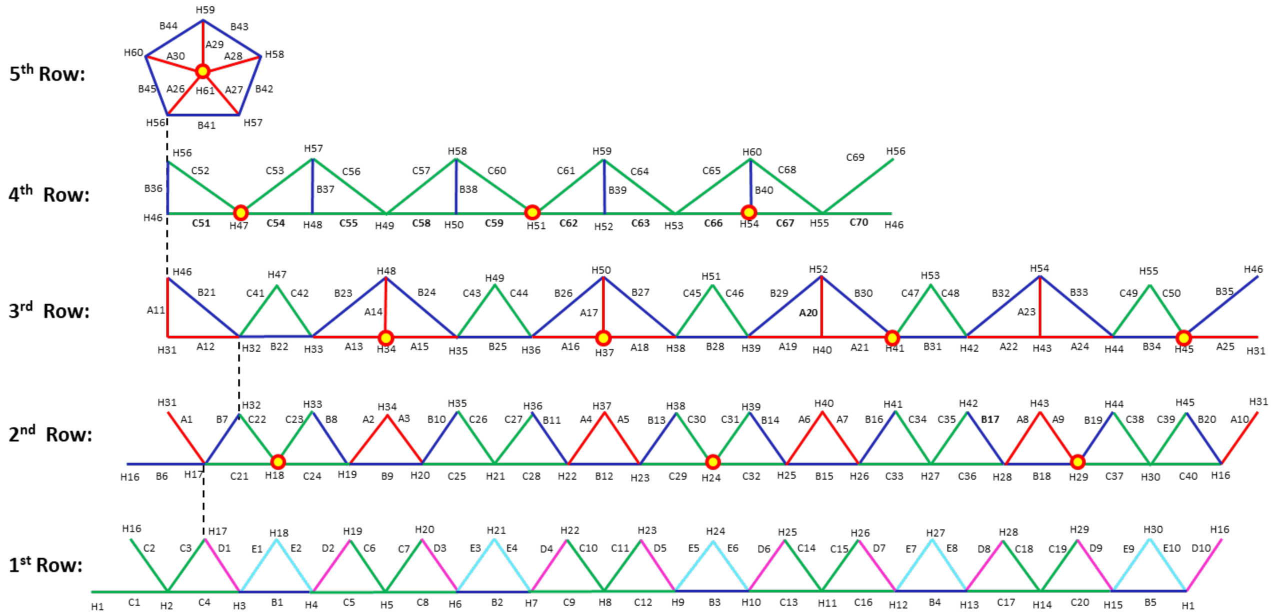
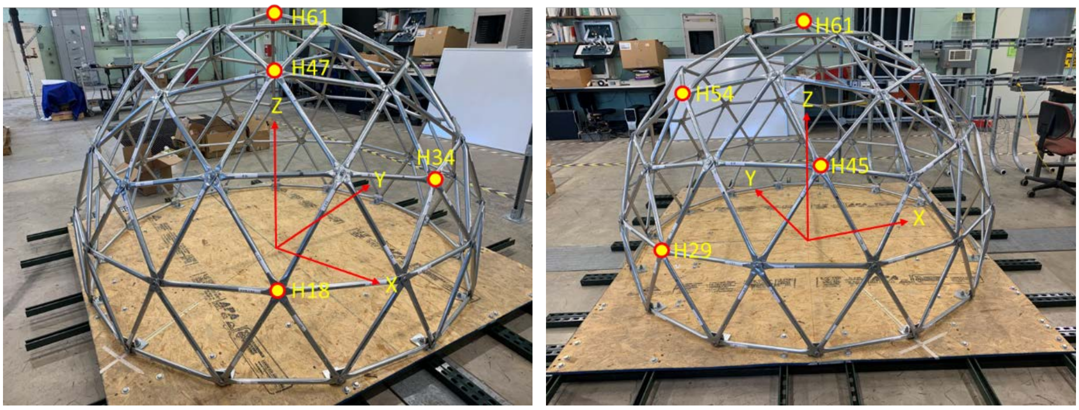


Figure 6. Node and rod numbers of 3D geodesic dome structure (light points show the sensor location).

We also marked the positions of these 11 accelerometers on the actual photo of the geodesic dome structure shown in **Figure 7**. It should be noted that the accelerometers collect three-axis acceleration data based on their own local coordinate systems, which need to be converted to the global coordinate system ***X-Y-Z*** of the geodesic dome to facilitate multi-sensor information fusion.



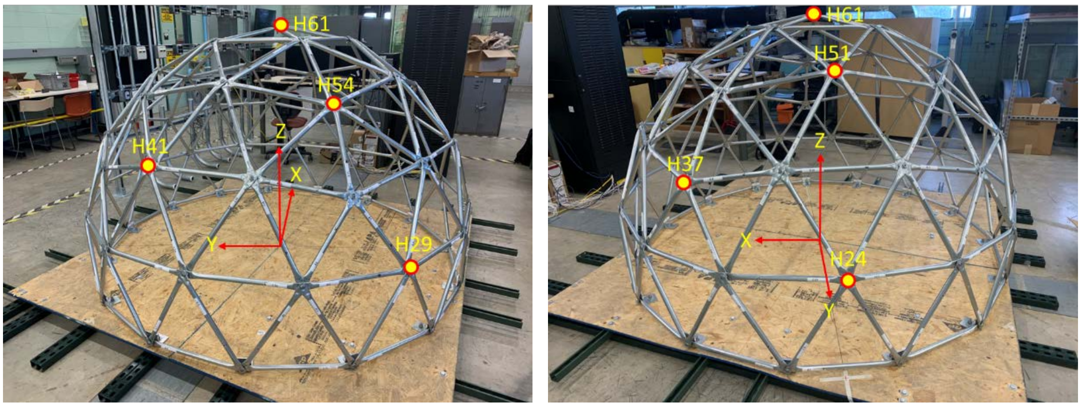


Figure 7. Location of sensors in 3D geodesic dome structure.

After setting up the physical structure of the 3D geodesic dome, we used a digital hammer to strike the dome to record the magnitude of the impact force. It should be noted that currently, we only strike at the nodes of the dome, which means, excluding the 15 nodes where the dome contacts the base (1st Row in **Figure 6**), there are 46 possible positions for the impact force to act on. We conducted a total of 4 sets of striking experiments, carried out over 4 days. The information about the experiments is shown in Table 2 below.

Table 2. Physical structure test information

Test day	Order of the stuck nodes	Number of strikes
Day1	[H32, H39, H44, H33, H50, H52]	[15, 15, 15, 6, 6, 7]
Day2	[H16, H17, H19, H20, H21, H26, H27, H28, H30, H31, H32, H33, H35, H42, H43, H44, H46, H48, H53, H55]	Each node 10 times
Day3	[H16, H17, H19, H20, H21, H26, H27, H28, H30, H31, H32, H33, H35, H42, H43, H44, H46, H48, H53, H55]	Each node 3 times
Day4	[H16, H17, H19, H20, H21, H26, H27, H28, H30, H31, H32, H33, H35, H42, H43, H44, H46, H48, H53, H55]	Each node 3 times

It should be noted that the sampling rate of the accelerometers is 1600Hz. We extracted samples from the time-domain signal within 1 second of the impact occurring. Moreover, each sensor captures acceleration signals in three directions in the global coordinate system. In this study, we only use the signals in the Z direction for analysis. In the physical experiments, we obtained a total of 384 sets of samples with size (384, 1600, 11). We randomly selected a set of time-domain signal samples and performed a Fast Fourier Transform to obtain the corresponding frequency-domain signals.

4.1.2 3D geodesic dome structure: numerical model experiment

Referring to the physical structure in laboratory, we established a proportional finite element model of the geodesic dome, which is also made of 165 struts and 61 hubs for sampling the numerical response. The whole modeling process is divided into two parts: the numerical modeling and the model updating. This approach aims to obtain a numerical model that aligns as closely as possible with the actual conditions, thus, reducing the distribution difference between numerical and physical responses.

- **Numerical modeling**

The numerical modeling consists of modeling the dome, the base and the connection between dome and base. The total stiffness matrix and mass matrix of the model can be set as

$$[K]=[K_d]+[K_c]+[K_b],\ [M]=[M_d]+[M_c]+[M_b], \tag{2}$$

where, K_d, K_c and K_b are the stiffness matrices for dome, connection and base section respectively, M_d, M_c, M_b are the mass matrices for dome, connection and base section, respectively. Each of struts in dome (**Figure 8(a)**) is considered a beam element. Such elements can carry both axial and bending loading. A schematic of the joint section is presented in **Figure 8(b)**. A strut of the joint (black) constitutes two nodes *i* and *j*. A local coordinate is assumed as (ξ, η, ζ), is along the axis of the strut. The rotation about the η and ζ axis are θ_η and θ_ζ . The stiffness matrix of a strut, k_e is computed as

$$[k_e]=E_T[k_{eT-unity}]+E_B[k_{eB-unity}], \tag{3}$$

where, E_T and E_B are the modulus of elasticity for axial and transverse loading, respectively. The [$k_{eT-unity}$] is the stiffness matrix accounting only for the axial loading, whereas [$k_{eB-unity}$] for transverse loading. Similarly, considering ρ_d as the density of the material at dome, the mass matrix can be defined as

$$[m_e]=\rho_d[M_{e-unity}]. \tag{4}$$

Then, the global stiffness and mass matrices follow

$$[K_d]=\sum_{e=1}^{e=61}[T_e][k_{e-unity}] \quad \text{and} \tag{5a}$$

$$[M_d]=\sum_{e=1}^{e=61}[T_e][m_{e-unity}], \tag{5b}$$

where, T_e is the transformation matrix for strut *e*. After that, we model the dome and base connection. There are 15 connections for the dome and the base. The connection is modeled as beam elements as shown as the red colored elements of **Figure 8(c)**. Considering E_c and ρ_c as modulus of elasticity and density of the connection, the stiffness and mass matrices follow

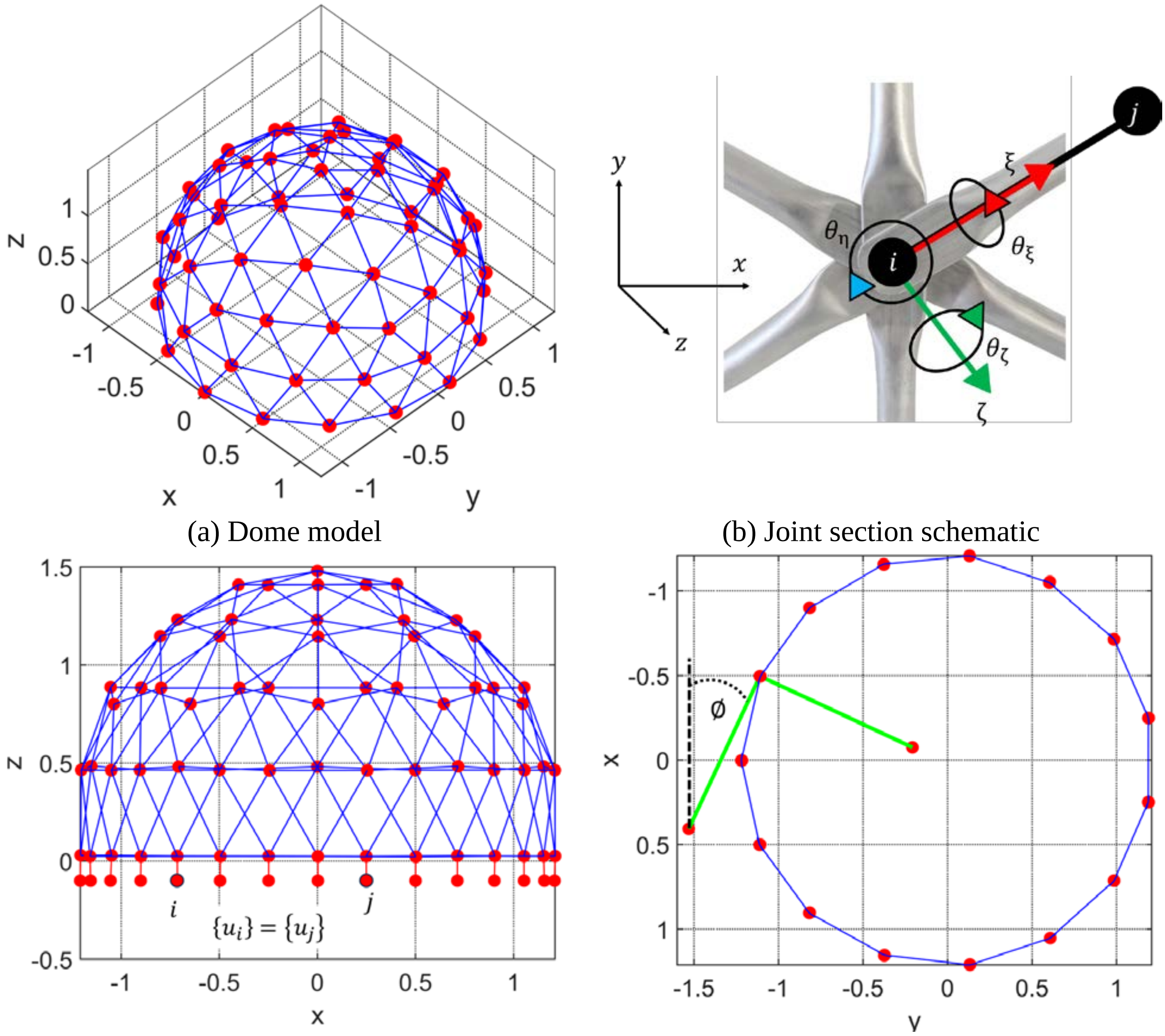
$$[K_c]=E_c\sum_{e=1}^{e=15}[k_{e-unity}] \quad \text{and} \tag{6a}$$

$$[M_c]=\rho_c\sum_{e=1}^{e=15}[m_{e-unity}]. \tag{6b}$$

Finally, we begin modeling the base. The base is modeled as a simple 2 spring system with stiffness E_p as presented in **Figure 8(d)**. The angle of the spring with respect to the X-axis Φ=25°. For the nodes at the connection, the rigid body constraint is implemented such that, all these nodes will have same displacement and rotation, as

$$\{u_i\}=\{u_j\}, i, j\in\{1,2,...,15\}. \tag{7}$$

The stiffness and mass matrices corresponding to the 15 ground nodes are combined into a single point to satisfy Equation (7) and assembled with the spring model of the base.



(a) Dome model (b) Joint section schematic

(c) Dome and base connection schematic (d) Base model and connection to the dome

Figure 8. Numerical modeling of the 3D geodesic dome.

➢ **Model updating**

After establishing the numerical model, we need to match the numerical model with the real physical structure as closely as possible through Bayesian model updating. To obtain the posterior probability density function (PDF) of the model parameters θ conditioned on the measurement data D, the transitional Markov chain Monte Carlo (TMCMC) [51] approach is employed. The idea is to construct a series of intermediate PDFs that start from the prior PDF $f(\theta|M)$ and converge to the posterior PDF $f(\theta|M,D)$. Suppose m stages are utilized:

$$f_j(\theta) \propto f(D|\theta, M)^{\rho_j} f(\theta|M),\ j = 0,\ldots,m, 0 = \rho_0 < \rho_1 < \ldots < \rho_m = 1, \quad (8)$$

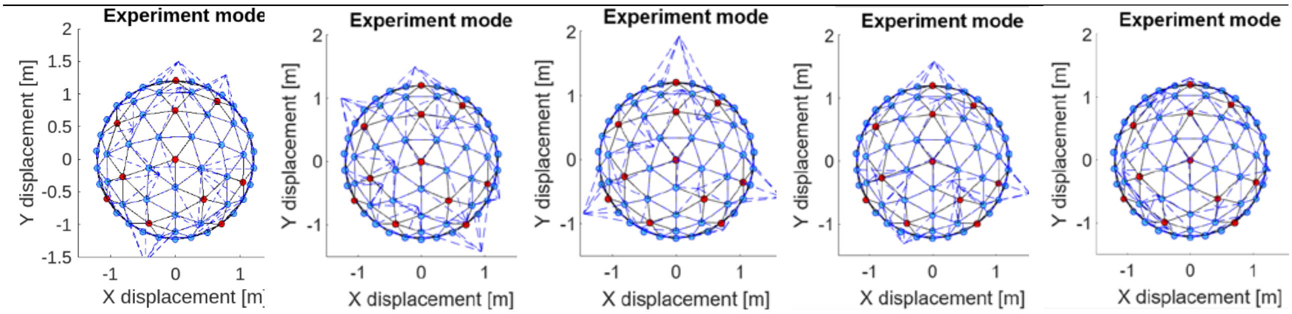
Where M is the probabilistic model class for the physical structure. After the model update, **Table 3** and **Table 4** respectively display the comparison of natural frequencies and modes between the updated numerical model and the physical structure. It should be noted that since only 11 nodes in the laboratory model were equipped with sensors, to correspond with the laboratory results, the finite element model also compared the shapes at only these 11 corresponding nodes.

Table 3. Natural frequencies comparison between experiment and numerical model

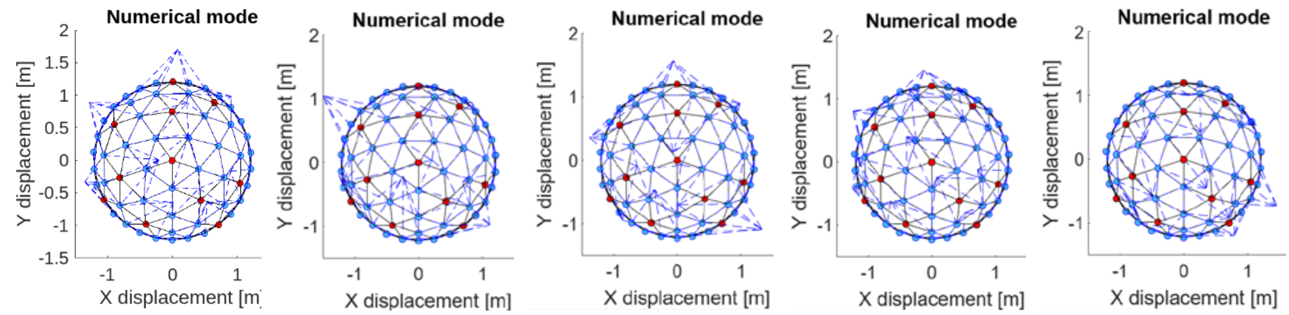
Mode	X-Direction			Y-Direction			Z-Direction		
	Num.(Hz)	Exp. (Hz)	Error	Num. (Hz)	Exp. (Hz)	Error	Num. (Hz)	Exp. (Hz)	Error
1	142.59	141.38	1.21	142.25	141.35	0.9	142.3	141.4	0.9
2	142.71	142.64	0.07	143.59	142.63	0.96	150.89	142.62	8.27
3	155.91	153.48	2.43	150.94	153.39	2.45	152.07	153.37	1.3
4	159.58	163.49	3.91	159.34	163.56	4.22	154.52	163.57	9.05
5	162.41	174.75	12.34	162.46	174.86	12.4	154.63	174.81	20.18

Table 4. Mode shapes comparison between experiment and numerical model (top view).

Mode 1	Mode 2	Mode 3	Mode 4	Mode 5



(a) Mode shapes obtained from experiment



(b) Mode shapes obtained from numerical model

Based on the results of the model update using TMCMC, the numerical model corresponds well with the physical structure in terms of the principal frequency. The simulation results differ by an average of 2Hz from the physical experiment in the first three modes in the X, Y, and Z directions. However, the numerical model cannot fully match the physical model in terms of modes. This is because the experimental dome is a complex structure with many degrees of freedom, but the number of sensors used in the experiment is limited. For the laboratory test dome structure, accelerometers were installed at only 11 of the 61 total nodes. Therefore, the task of updating the dome model involves more unknowns than known conditions, making it an ill-posed problem and thus significantly increasing the difficulty of model updating [52]. As shown in **Table 4**, the comparison of the five modes between the physical model and the numerical model shows that the deformation at the positions of the 11 red nodes is difficult to match accurately. Moreover, there are many uncertainties in the dome experiment, such as the base not being directly fixed to the ground. Additionally, when striking a joint, the struts will experience rigid displacement at the location of the joint. Therefore, merely updating the model cannot compensate for the systematic differences between the numerical model and the physical structure, the study employs deep learning methods to perform this task.

After completing the establishment of the numerical model, we struck all 46 nodes of the dome (excluding the 15 grounded nodes) with forces ranging from 2N to 500N, in 2N increments, obtaining a total of 11,500 sets of samples. The sampling rate was set to 1600Hz, consistent with the physical structure experiment. We randomly selected a set of time-domain signal samples and performed a Fast Fourier Transform to obtain the corresponding frequency-domain signals.

4.1.3 Impact localization and quantification

Based on the physical structure experiments and numerical model experiments, we obtained two datasets: the physical experiment set, with 384 samples, and the numerical experiment set, with 11500 samples. We divided them into training, validation, and test sets in the proportions of 0.64, 0.16, and 0.2, respectively. **Table 5** shows the specific format and description of the data used.

Table 5. The size and description of used data.

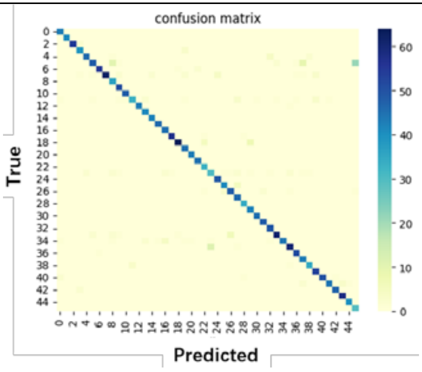
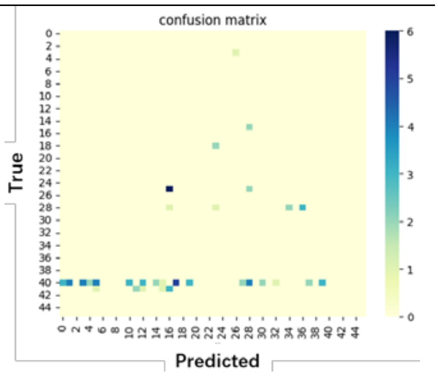
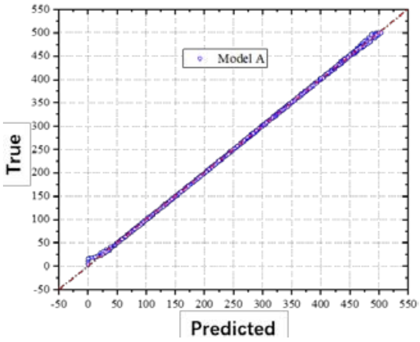
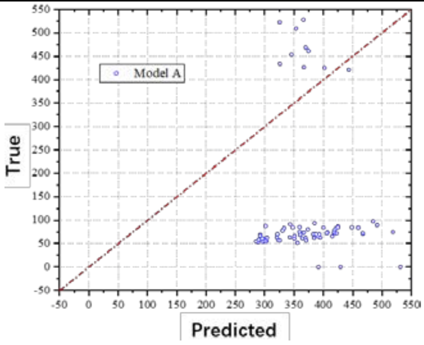
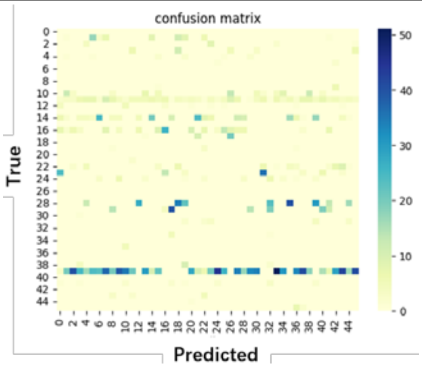
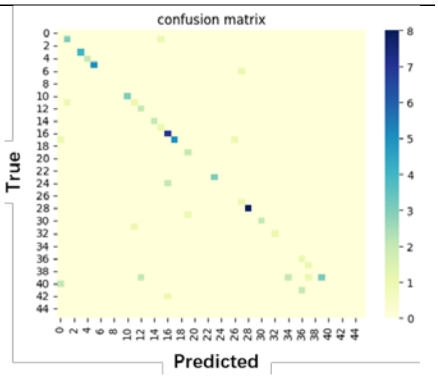
Dataset	**Input**	**Ground Truth**	
	11 channels of acceleration signals	Impact Force (N.)	Impact position (one-hot type)
Training set			
Numerical model	(7,360, 1600, 11)	(7,360, 1)	(7,360, 46)
Physical experiment	(245, 1600, 11)	(245, 1)	(245, 46)
Validation set			
Numerical model	(1840, 1600, 11)	(1840, 1)	(1840, 46)
Physical experiment	(61, 1600, 11)	(61, 1)	(61, 46)
Test set			
Numerical model	(2300, 1600, 11)	(2300, 1)	(2300, 46)
Physical experiment	(78, 1600, 11)	(78, 1)	(78, 46)

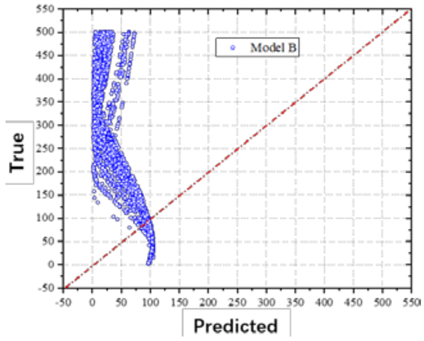
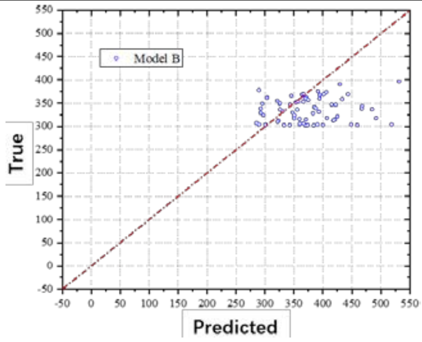
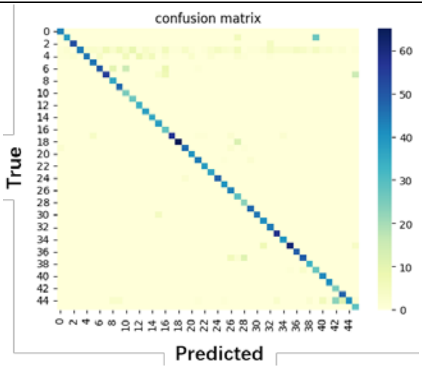
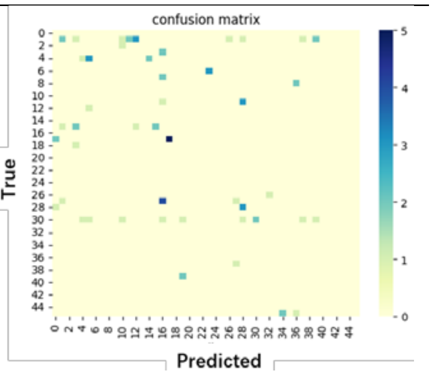
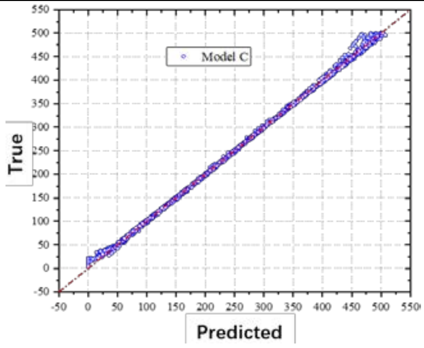
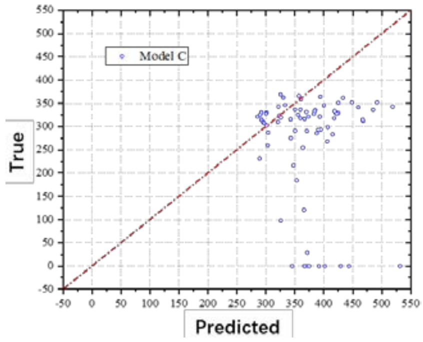
We used the TensorFlow 2.12 framework to establish all neural networks in the case study. The CPU and GPU for training these networks were the "13th Gen Intel(R) Core(TM) i9-13900 @ 3.00 GHz" with 64GB RAM and the "NVIDIA GeForce RTX 4090", respectively. The core code for the Transfer-AE is available on GitHub at the following address: *https://github.com/ChengjiaHanSEU/Transfer-AE.git*.

First, we conducted conventional neural network training as the baseline for this task. For ease of subsequent comparison, we used the Task module of Transfer-AE for training. The network input is the structural response, and the output includes two parts: the magnitude of the force and the location of the force. We simulated three training scenarios: 1) Training using only the numerical experiment set, denoted as Model A; 2) Training using only the physical data set, denoted as Model

B; 3) Training using a mix of the physical experiment set and the numerical experiment set, denoted as Model C. The networks for the three training scenarios were tested on both the numerical and physical test sets, with the results shown in **Table 6**. It should be noted that in **Table 6**, we use the F1-score to evaluate the performance of position classification and MAPE to assess the regression performance of the force.

Table 6. Performance of the baseline model on the test set under three training scenarios

Direct training model		**Numerical test set**	**Physical test set**
Model A: Training using only numerical model data	Impact location		
	F1-score	**0.9133**	0
	Impact quantification		
	MAPE	**0.0401**	0.7321
Model B: Training using only real model data	Impact location		
	F1-score	0.0432	**0.728**

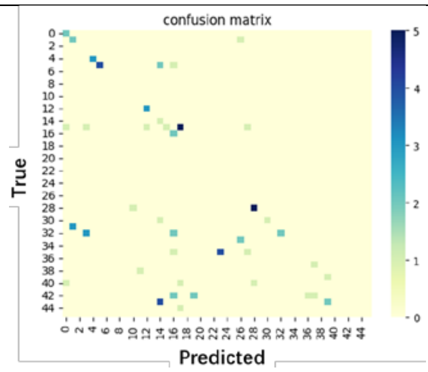
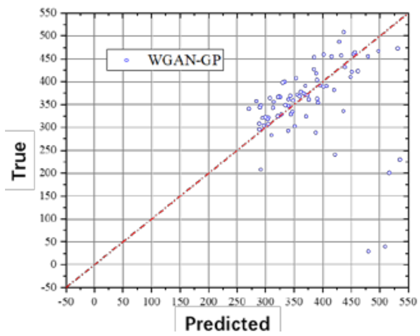
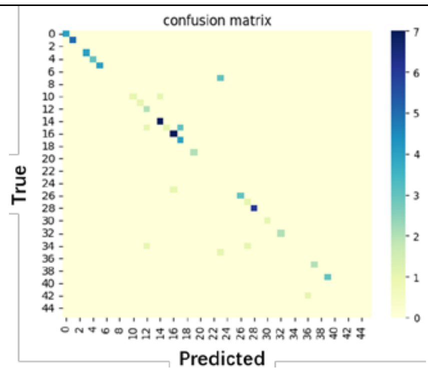
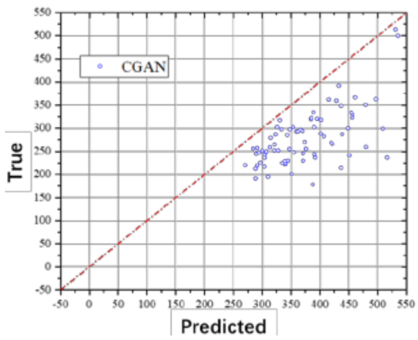
	Impact quantification		
	MAPE	1.384	**0.140**
Model C: Mixing numerical model data with real model data for training	Impact location		
	F1-score	0.8160	0.2668
	Impact quantification		
	MAPE	0.0586	0.3132

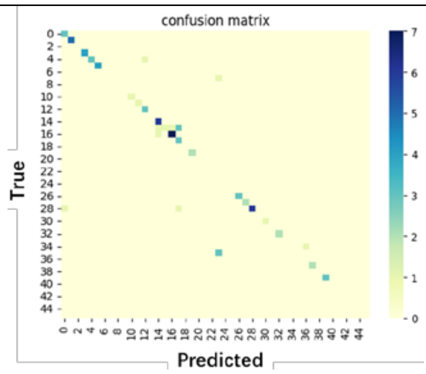
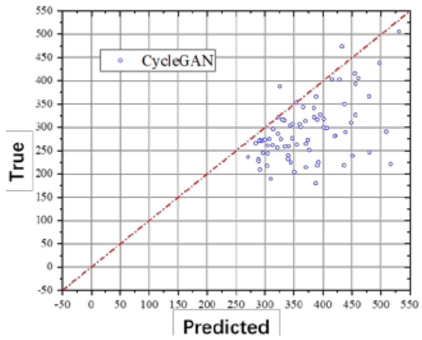
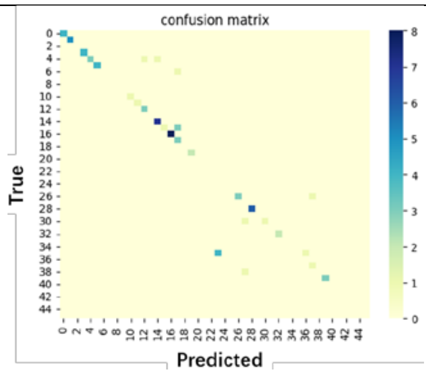
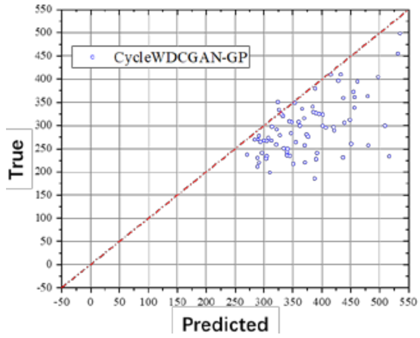
* The best results among the models in the table are underlined and bolded.

Table 6 shows that models trained on a single dataset (Numerical or Physical) perform well on their corresponding test set but perform poorly on the other test set. This can be understood as being due to the distribution differences between the datasets. However, it should be noted that due to some noise in the physical experimental data itself (there are also distribution differences between results tested at different test times), the model B trained with physical experimental data only achieved an F1-score of 0.728 and a MAPE of 0.140 on the physical test set. Model C was trained using both physical experimental data and numerical experimental data. From its performance on both the physical and numerical test sets, it can be found that due to the larger sample size and lower noise level of the numerical experiments, Model C performs better on the numerical test set than on the physical test set. However, compared to Models A and B, its performance on both test sets has declined. This outcome underscores the limitation of merely blending datasets for training purposes, highlighting that such an approach falls short in effectively addressing impact detection challenges posed by the discrepancies in data distribution across different datasets.

Building on this, we first explored the DA methods based on deep learning. That is, through DA, we transfer the response domain from physical structures to the response domain of numerical models (which have a larger volume of data and are more suitable for training impact detection models). Subsequently, we use models trained on numerical model data to detect impacts on physical structures. It should be noted that to establish a correspondence between the response domain of physical structures and that of numerical models, we selected 384 samples from the 11,500 finite element model samples that correspond to the 384 real samples to create pairs. These were then divided in a ratio of 0.8 to 0.2 to train the DA model. We established four DA models based on WGAN-GP [48], CGAN [53], CycleGAN [36], and CycleWDCGAN-GP [37]. All of them used Model A from **Table 6** as the impact detection model, so their results on the numerical test set are the same as those of Model A. After completing the training, we only need to test the results of these four models on the physical test set, and their results are summarized in **Table 7**.

Table 7. The performance of GAN-based DA models on physical test set.

Model	**Impact location**	**Impact quantification**
WGAN-GP	 F1-score: 0.3545	 MAPE: <u>**0.1402**</u>
CGAN	 F1-score: 0.8023	 MAPE: 0.2535

CycleGAN	 F1-score: 0.8037	 MAPE: 0.2152
CycleWDCGAN-GP	 F1-score: **<u>0.8116</u>**	 MAPE: 0.2099

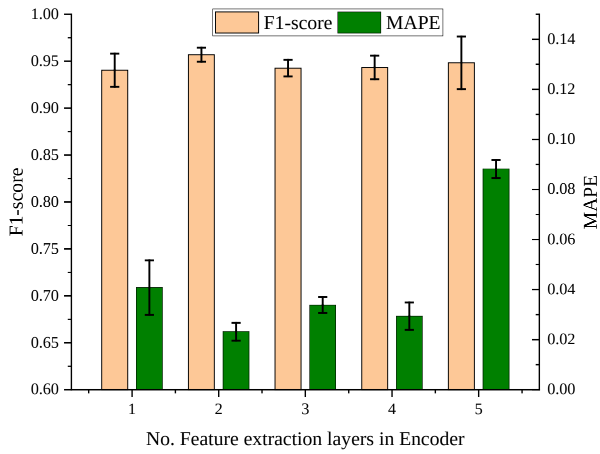
* The best results among the models in the table are underlined and bolded.

From the results in **Table 7**, it can be seen that WGAN-GP performed the worst in the task of impact localization but achieved the best performance in impact quantification with a MAPE of 0.1402. Meanwhile, CycleWDCGAN-GP achieved the best results in impact localization with an F1-SCORE of 0.8116 and ranked second in impact quantification with a MAPE of 0.2099. This is because the loss function of WGAN-GP does not include a direct calculation of the distance between tensors before and after the domain transfer, meaning that WGAN-GP has greater generative flexibility but is less stable. Considering the performance in both tasks, CycleWDCGAN-GP has the best overall effect.

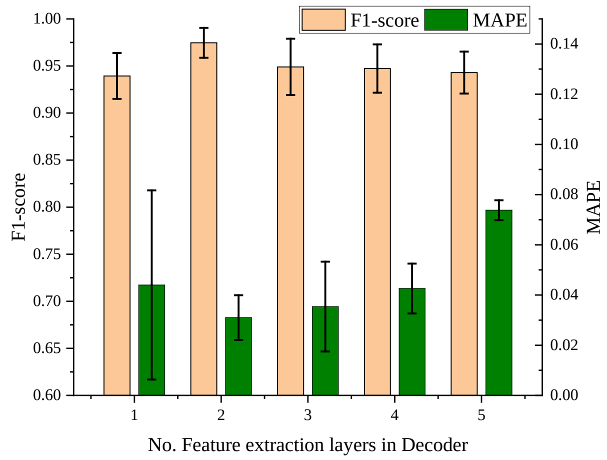
Next, before testing the proposed Transfer-AE (mode1 and mode2), we first conduct a hyperparameter analysis of the Transfer-AE structure using the numerical model data to determine the optimal hyperparameter of the structure for the impact detection task. Since the proposed Transfer-AE consists of three different modules, the depth of each module can significantly affect the model's performance. We have identified four key hyperparameters to focus on:

1. The number of feature extraction layers before each downsampling in the Encoder module.
2. The number of feature extraction layers before each upsampling in the Decoder module.
3. The number of 1D-convolutional layers before each downsampling in the Task module.
4. The number of fully connected (FC) layers before the final output in the Task module.

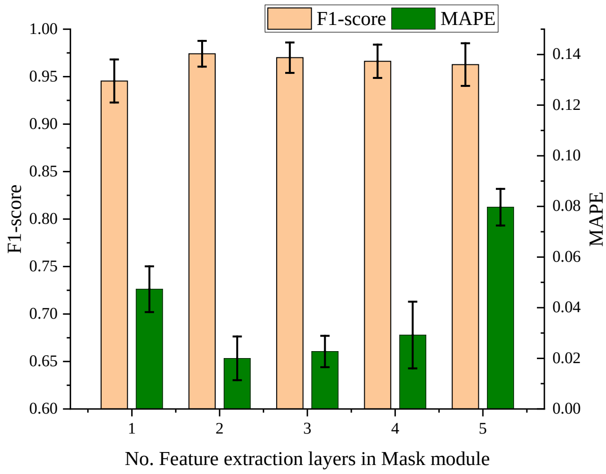
The values for the four hyperparameters range from 1 to 5 layers. Each hyperparameter was tested five times, and the results of the hyperparameter tests are shown in **Figure 9**. In the subplots of **Figure 9**, the x-axis represents the values of the hyperparameters. The left y-axis represents the network's F1-score for the impact localization task, and the right y-axis represents the network's MAPE score for the impact quantification task. From **Figures 9(a)**, **(b)**, and **(c)**, it can be observed that using 2 feature extraction layers before each downsampling in the Encoder, 2 feature extraction layers before each upsampling in the Decoder, and 2 layers of 1D convolution for each downsampling in the Mask module achieves the best classification F1-score and regression MAPE. Additionally, their error bars are the smallest. From **Figure 9(d)**, it can be observed that as the number of fully connected layers increases, both the classification accuracy and regression accuracy of the network decrease. The network performs best when only one fully connected layer is used before the final output in the task module. Therefore, in our subsequent research, we will assemble the Transfer-AE model using the optimal combination of "2 feature extraction layers/each pooling in Encoder" - "2 feature extraction layers/each pooling in Decoder" - "2 Conv1D layers/each pooling in Task module" - "1 fully connected layer before output."



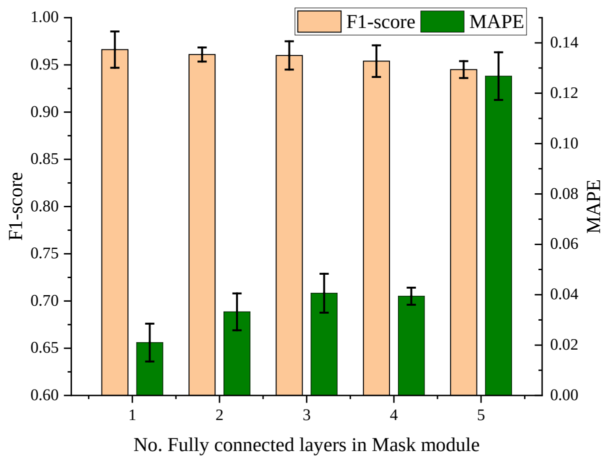
(a) Feature extraction layers in Encoder



(b) Feature extraction layers in Decoder



(c) Feature extraction layers in Task module

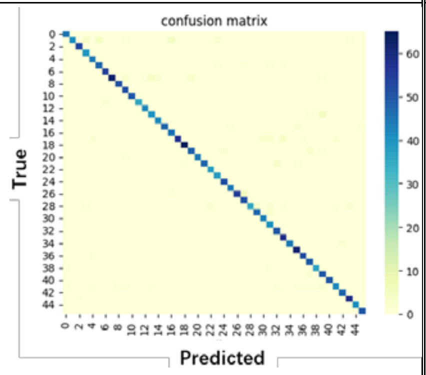
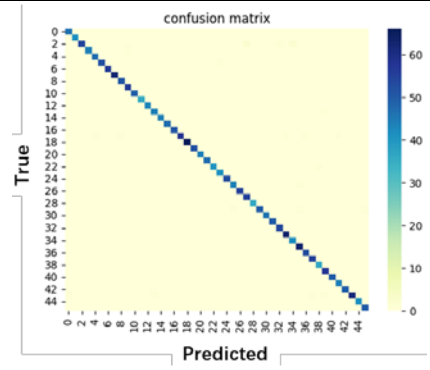
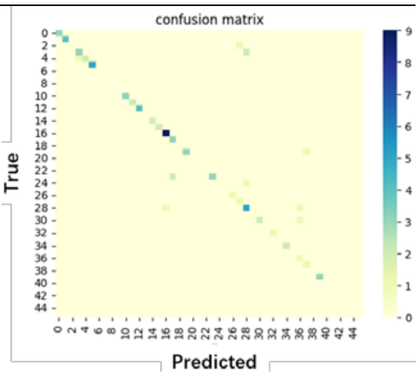
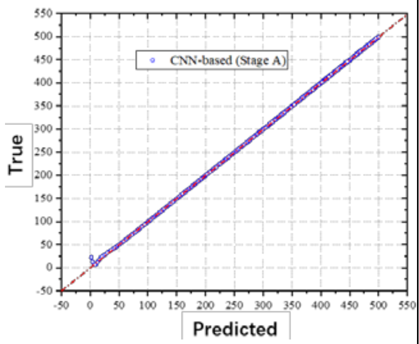
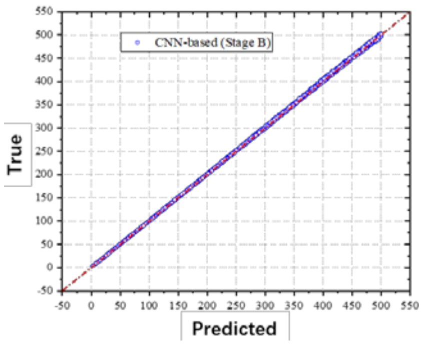
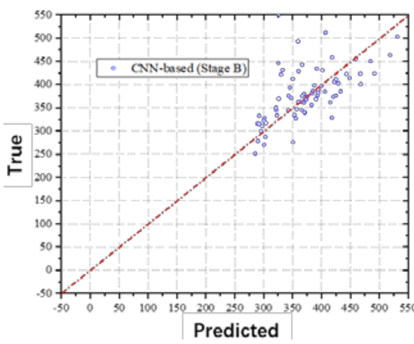
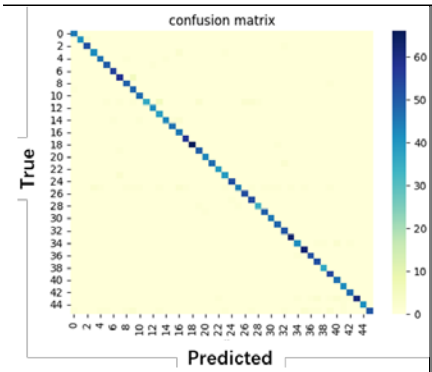
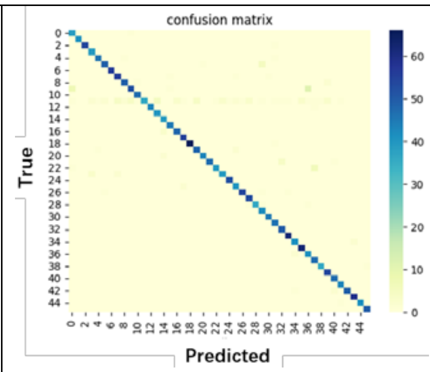
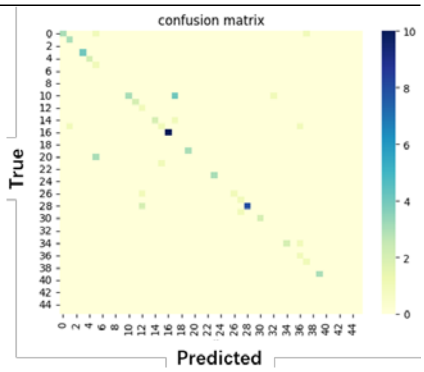


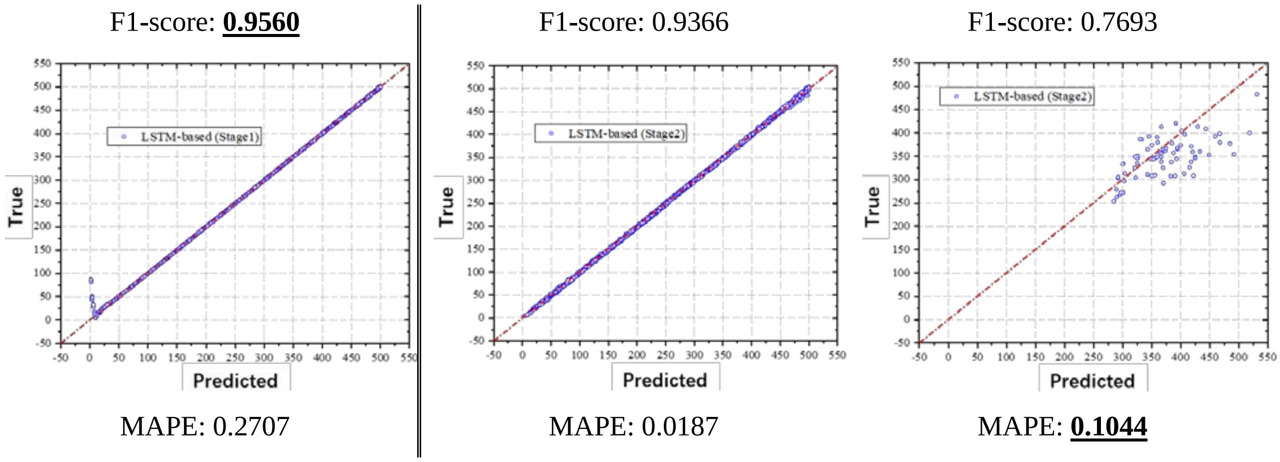
(d) Fully connected layers in Task module

Figure 9. Results of the hyperparameter tests for Transfer-AE.

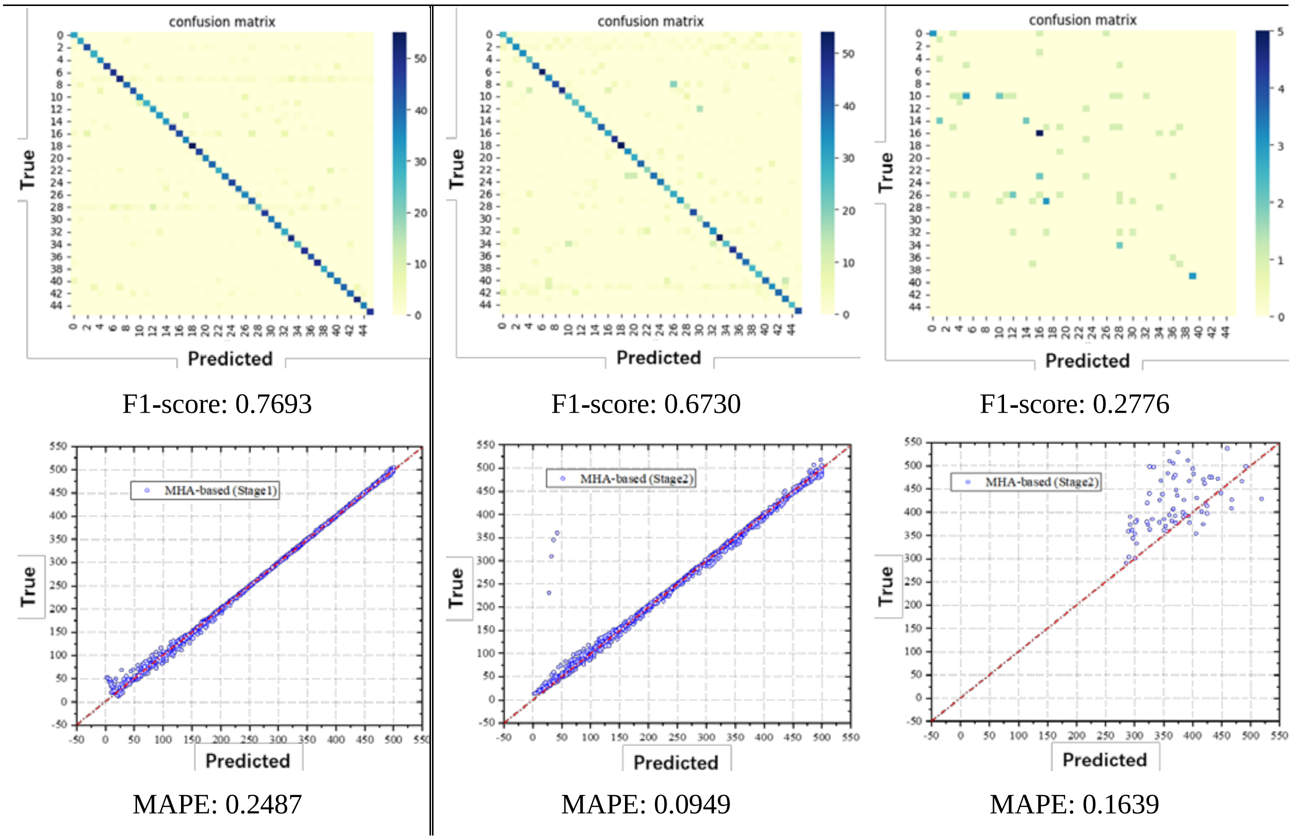
Then, following the methodology described, we established the Transfer-AE (mode 1) model. In constructing the Transfer-AE network, we experimented with three different feature extraction units: those based on CNN units, LSTM units, and multi-head attention (MHA) units, in order to compare the performance differences between these extraction methods. For the three Transfer-AE (mode 1) models, we first used the numerical experiment set for Stage 1 pre-training, and then proceeded to joint training on the physical experiment set and numerical experiment set in Stage 2. **Table 8** displays the performance of Transfer-AE (mode 1) on their respective test sets for both Stage 1 and Stage 2. Similarly, we use F1-score and MAPE as metrics.

Table 8. The performance of three different units of Transfer-AE (mode1) on test sets

Train Stage 1	Train Stage 2	
Numerical test set	**Numerical test set**	**Physical test set**
		
F1-score: 0.9464	F1-score: **0.9903**	F1-score: **0.8598**
		
MAPE: **0.0704**	MAPE: **0.0120**	MAPE: 0.1063
(a) CNN-based Transfer-AE (mode1)		
		



(b) LSTM-based Transfer-AE (mode1)



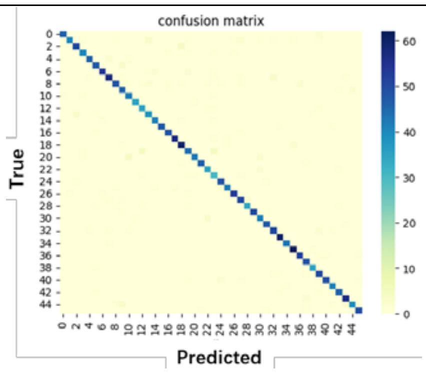
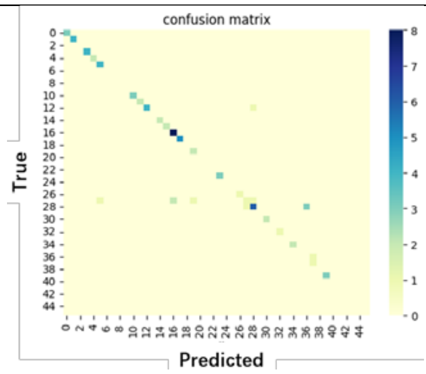
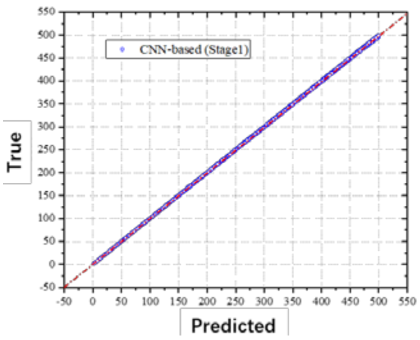
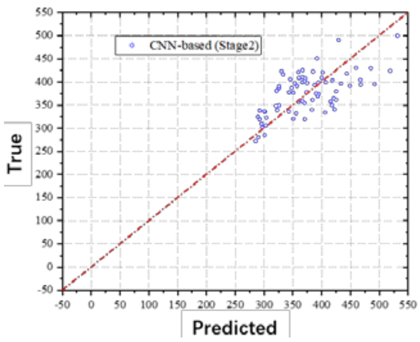
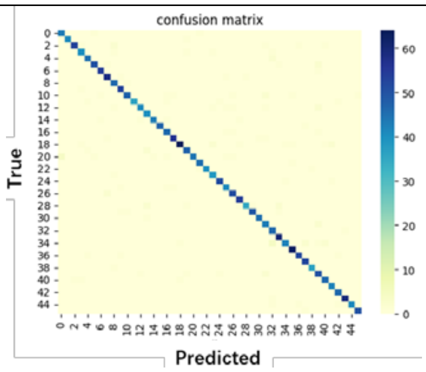
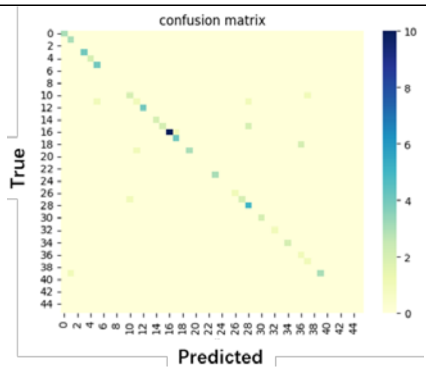
(c) MultiHeadAttention-based Transfer-AE (mode1)

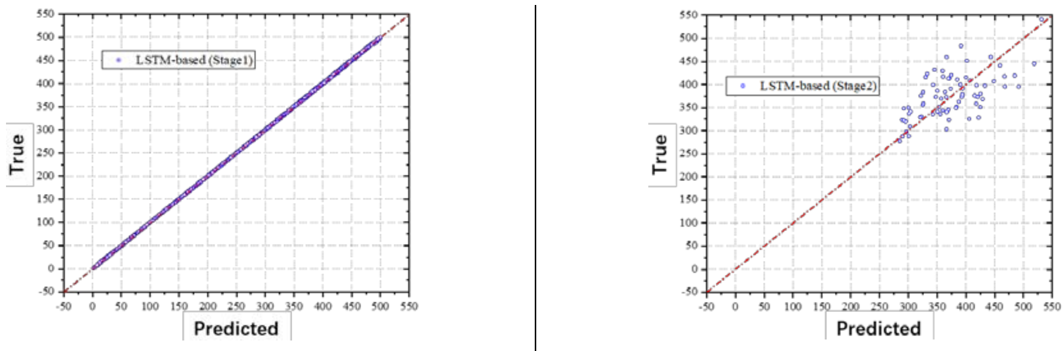
* The best results among the models in the table are underlined and bolded.

From **Table 8**, it can be observed that the all three Transfer-AE (mode 1) models, after training, achieved excellent results on both numerical and physical test sets, surpassing the performance of the baseline model. Notably, after introducing physical experimental data for training in Stage 2, Transfer-AE (mode 1) not only achieved effective recognition of impacts on the physical test set but also saw a significant improvement in the accuracy of recognizing impact magnitude and location on the numerical test set compared to the results from Stage 1. Moreover, a comparison of the three models reveals that the Transfer-AE (mode 1) based on CNN units achieved the best impact quantification performance. It achieves the highest accuracy in detecting the magnitude and location of impact among the three models, with an F1-score of 0.9903 and a MAPE of 0.0120 on the numerical test set, and it also achieved the highest impact location accuracy with an F1-score of 0.8598 on the physical test set. Moreover, on the physical test set, the best accuracy in impact quantification was achieved by the LSTM-based Transfer-AE (mode 1).

Similarly, we established three Transfer-AE (mode 2) models using the three types of feature extraction units. Each was trained on the numerical experiment set during Stage 1, followed by training on the physical experiment set during Stage 2. After training, the three models were then tested on both the numerical and physical test sets to evaluate their performance in detecting impacts, as shown in **Table 9.**

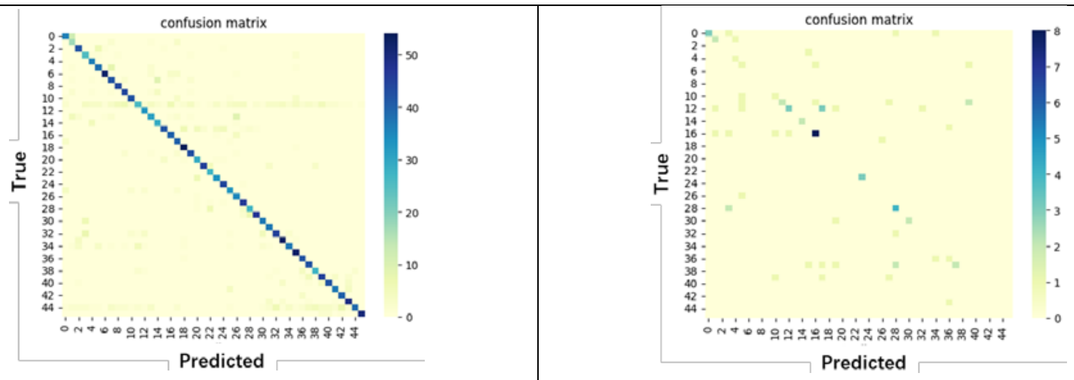
Table 9. The performance of three different units of Transfer-AE (mode2) on test sets

Train Stage 1	**Train Stage 2**
Numerical test set	**Physical test set**
 F1-score: 0.9208	 F1-score: **0.8492**
 MAPE: **0.0033**	 MAPE: **0.1016**
(a) CNN-based Transfer-AE mode2	
 F1-score: **0.9432**	 F1-score: 0.8459

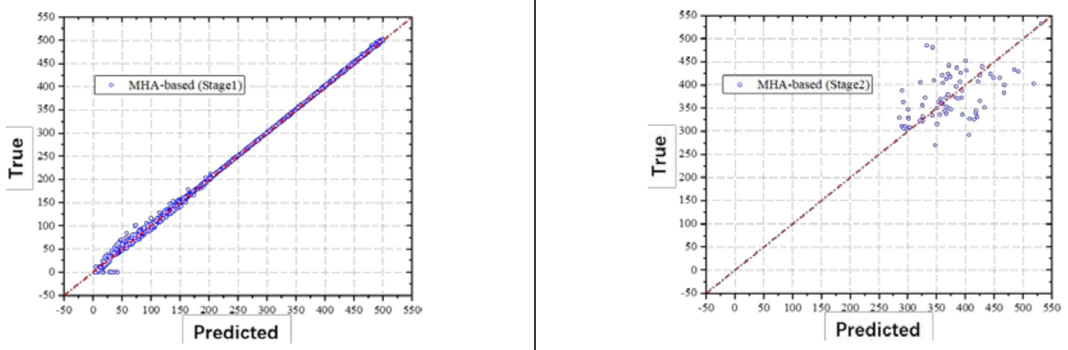


MAPE: 0.0051 MAPE: 0.1018

(b) LSTM-based Transfer-AE mode2



F1-score: 0.7660 F1-score: 0.5104



MAPE: 0.0391 MAPE: 0.1100

(c) MultiHeadAttention-based Transfer-AE mode3

* The best results among the models in the table are underlined and bolded.

From **Table 9**, it can be seen that all three Transfer-AE (mode 2) models surpassed the baseline performance on both test sets. Furthermore, similar to the Transfer-AE (mode 1) models, the CNN-based Transfer-AE (mode 2) achieved the highest accuracy in detecting impact magnitude on the numerical test set, with a MAPE of 0.0033, and exhibited the best performance among the three models on the physical test set, with an F1-score of 0.8492 and a MAPE of 0.1016. Moreover, the LSTM-based Transfer-AE (mode 2) achieved the best accuracy in detecting impact locations on the numerical test set, with an F1-score of 0.9432.

In **Table 10**, we summarized the performance of the baseline model, the deep learning-based DA models and the six Transfer-AE models in the task of impact quantification and location. In **Table 10**, we highlighted the best metrics for each item by using bold and underlining. From **Table 10**, it is evident that, compared to the baseline and the CycleWDCGAN-GP model, which has the best performance among the mainstream DA methods, both Transfer-AE (mode 1) and Transfer-AE (mode 2) achieved remarkable accuracy in detecting and locating impacts on both the numerical and physical test sets. This validates the effectiveness of the two proposed Transfer-AE models, as they successfully bridged the distribution differences between samples from the numerical space and those from the real physical space, which enabled one network to achieve consistent impact detection results across structural responses of varying distributions.

Table 10. Summary of models' performance in impact detection.

Model		**Location**				**Quantification**
		Accuracy	**Precision**	**Recall**	**F1-score**	**MAPE**
Baseline	Numerical	0.8304	0.8723	0.8304	0.8160	0.0586
	Physical	0.2105	0.4934	0.2105	0.2668	0.3132
WGAN-GP	Numerical	0.9156	0.9241	0.9156	0.9133	0.0401
	Physical	0.3684	0.5251	0.3684	0.3545	0.1402
CGAN	Numerical	0.9156	0.9241	0.9156	0.9133	0.0401
	Physical	0.8289	0.8294	0.8289	0.8023	0.2535
CycleGAN	Numerical	0.9156	0.9241	0.9156	0.9133	0.0401
	Physical	0.8157	0.8590	0.8157	0.8037	0.2152
CycleWDCGAN-GP	Numerical	0.9156	0.9241	0.9156	0.9133	0.0401
	Physical	0.8157	0.8573	0.8157	0.8116	0.2099
CNN Tansfer-AE (mode1)	Numerical	**0.9904**	**0.9906**	**0.9904**	**0.9903**	0.0120
	Physical	**0.8552**	**0.8731**	**0.8552**	**0.8598**	0.1063
LSTM Tansfer-AE (mode1)	Numerical	0.9413	0.9456	0.9413	0.9366	0.0186
	Physical	0.7500	0.8622	0.7500	0.7693	0.1044
MHA Tansfer-AE (mode1)	Numerical	0.6834	0.7079	0.6834	0.6730	0.0949
	Physical	0.2763	0.3703	0.2763	0.2776	0.1639
CNN Tansfer-AE (mode2)	Numerical	0.9208	0.9238	0.9208	0.9208	**0.0033**
	Physical	**0.8552**	0.8675	**0.8552**	0.8492	**0.1016**
LSTM Tansfer-AE (mode2)	Numerical	0.9434	0.9456	0.9434	0.9432	0.0051
	Physical	**0.8552**	0.8698	**0.8552**	0.8459	0.1018
MHA Tansfer-AE (mode2)	Numerical	0.7808	0.7935	0.7808	0.7660	0.0391
	Physical	0.4605	0.6846	0.4605	0.5104	0.1100

* The best results among the models in the table are underlined and bolded.

Moreover, when comparing the Transfer-AE models under the two modes with different units, it can be observed that for the task of locating impacts, Transfer-AE (mode 1) performs better, with the CNN-based Transfer-AE (mode 1) achieving the best performance among the six models on both datasets. On the other hand, for the quantification of impact magnitude, Transfer-AE (mode 2) shows superior performance, with the CNN-based Transfer-AE (mode 2) and the LSTM-based Transfer-AE (mode 2) achieving the lowest MAPE on the numerical and physical test sets, respectively. The analysis concludes that compared to LSTM and MHA, the broader global receptive field of CNN when processing time-domain signals may be the reason for its superior performance. Regarding this case, the mode 1 performs better than mode 2 because it makes fuller use of information fusion, which directly finds a universal mapping to the same latent space from two different distributions, whereas mode 2 calculates the residual between the two distributions directly. However, this situation may change when faced with more distributions. As more distributions are introduced, finding a common mapping for these distributions becomes increasingly difficult, whereas the difficulty for mode 2 remains constant when more distributions are introduced. Moreover, it should be noted that Transfer-AE (mode 1) has the advantage of faster inference speed because it does not add extra network structure when introducing new distribution samples for training. On the other hand, Transfer-AE (mode 2) benefits from faster training speed and lower training difficulty because it uses only the newly introduced distribution samples for training, additionally, there is no need to worry about the introduction of new distributions compromising or reducing the network's applicability to the old distribution data. **Table 11** summarizes the speeds of mode 1 and mode 2 for the three types of Transfer-AE models at different stages. From **Table 11**, it can be observed that for all three types of Transfer-AE, mode 1 has a smaller size and faster computational speed during inference compared to mode 2, but it is slower than mode 2 during the training phase.

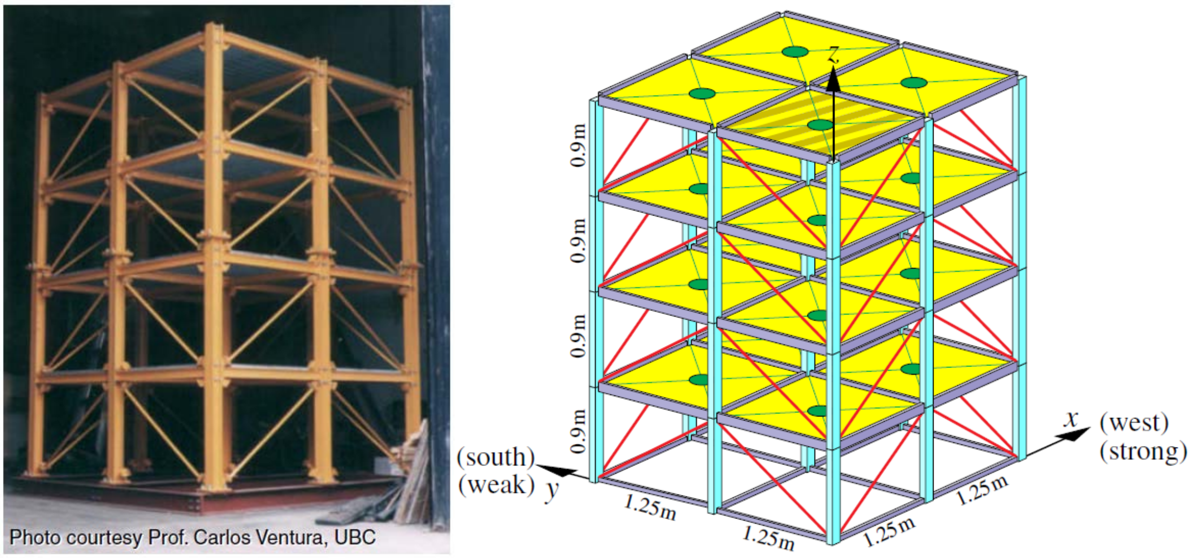
Table 11. Comparison of training and inference speeds between Mode 1 and Mode 2.

Model	Training period	Inference period
CNN Tansfer-AE (mode 1)	▫ 187ms/step ▫ 16s/epoch	▫ Total params: 7,492,538 ▫ 0.18s/step
CNN Tansfer-AE (mode 2)	▫ 89ms/step ▫ 5s/epoch	▫ Total params: 7,900,506 ▫ 0.22s/step
LSTM Tansfer-AE (mode 1)	▫ 141ms/step ▫ 9s/epoch	▫ Total params: 7,511,642 ▫ 0.25s/step
LSTM Tansfer-AE (mode 2)	▫ 197ms/step ▫ 12s/epoch	▫ Total params: 7,938,714 ▫ 0.29s/step
MHA Tansfer-AE (mode 1)	▫ 170ms/step ▫ 11s/epoch	▫ Total params: 7,141,835 ▫ 0.21s/step
MHA Tansfer-AE (mode 2)	▫ 97ms/step ▫ 6s/epoch	▫ Total params: 7,199,100 ▫ 0.29s/step

Finally, from the perspective of feature extraction units, it is apparent that Transfer-AE is more suited to using CNN-based units to construct its Encoder, and this is true for both mode 1 and mode 2. For the task of detecting and locating impacts on the geodesic dome in this case, taking all metrics into consideration, we recommend using the CNN-based Transfer-AE (mode 1) for digital twin.

4.2 Impact localization and quantification test on the IASC–ACSE benchmark structure

To further validate the generalizability of the proposed model in the impact localization and quantification tasks, we conducted tests on the IASC-ASCE benchmark [54,55] which is a benchmark framework used to evaluate the performance of structural health monitoring methods and control systems. The structure of the benchmark is shown in **Figure 10**. Phase II experiments of the IASC-ASCE benchmark include impact hammer tests while testing for structural damage, and the benchmark also provides multiple open-source finite element numerical models. Due to the limited number of impact hammer hits in the measured data (only 6-7 hits in a healthy state), it is challenging to establish a training set with enough samples for training. Therefore, our experiments are based on the finite element models provided by the IASC-ASCE benchmark. The IASC-ASCE benchmark offers two finite element analysis models with 12 DOF and 120 DOF [56]. The 120 DOF model is very close to reality and can serve as a substitute for the real structure after model updating, while the 12 DOF model simulation is quite rough. Hence, we do not perform model updating on the 12 DOF model. Instead, we use the domain-adaptive approach, which includes our proposed Transfer-AE as well as mainstream models, to verify whether it is possible to establish impact localization and quantification models. This approach aims to simultaneously identify structurally responsive data with distribution differences from the 12 DOF and 120 DOF models to achieve consistent detection results.



(a) Steel-frame scaled structure (b) Diagram of analytical model

Figure 10. Real structure and analytical model of benchmark [55].

Acceleration sensors are installed at 16 positions on the framework structure to collect structural response data, and the finite element model also has acceleration monitoring units at corresponding positions. According to the benchmark specifications, the impact hammer can be applied at four locations, corresponding to the four floors of the framework structure. The sampling rate of the finite element model is 1 kHz, and the benchmark indicates that the force from the hammer dissipates within 4 seconds, so we sample the structural response for 4 seconds in the model. We sampled a total of 100 different forces in the range of [9000(lbs.), 11000(lbs.)] at four different impact locations on the 12 DOF model, creating a time series dataset of shape (4×100, 4000, 16), referred to as $\boldsymbol{T}_{12}$. Similarly, in the 120 DOF model, we sampled 20 different forces within the same range, creating a time series dataset of shape (4×20, 4000, 16), referred to as $\boldsymbol{T}_{120}$. After shuffling $\boldsymbol{T}_{12}$ and $\boldsymbol{T}_{120}$, we split them into training and testing sets in an 8:2 ratio. **Table 12** shows the specific format and description of the benchmark data used.

Table 12. The size and description of used benchmark data.

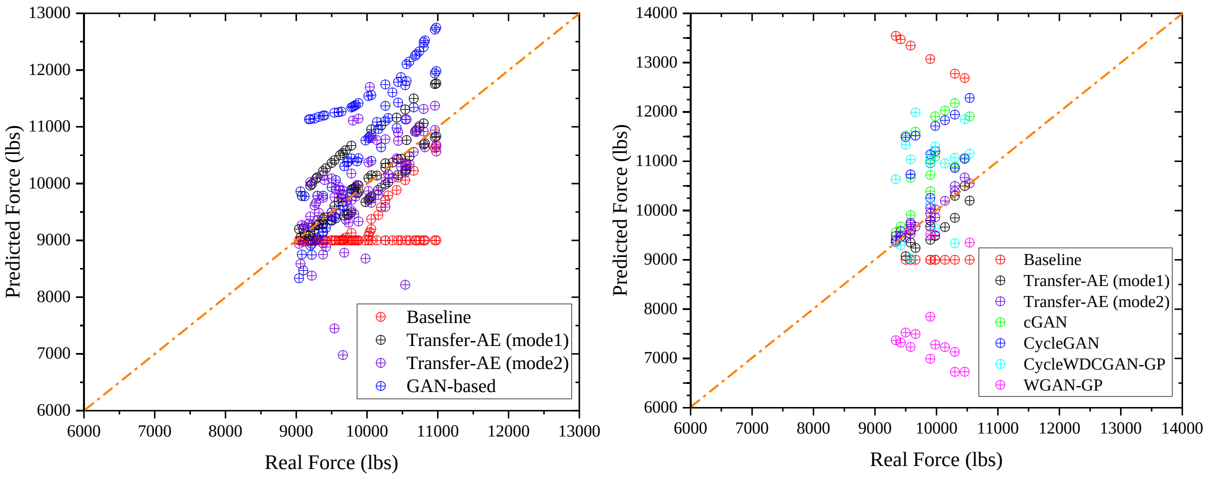
Dataset	Input	Ground Truth	
	16 channels of acceleration signals	Impact Force (lbs.)	Impact position (one-hot type)
Training set			
12 DOF model	(320, 4000, 16)	(320, 1)	(320, 4)
120 DOF model	(64, 4000, 16)	(64, 1)	(64, 4)
Test set			
12 DOF model	(80, 4000, 16)	(80, 1)	(80, 4)
120 DOF model	(16, 4000, 16)	(16, 1)	(16, 4)

Similar to Case 1, we established the baseline model, WGAN-GP, CGAN, CycleGAN, CycleWDCGAN-GP, as well as mode1 and mode2 of Transfer-AE. For the baseline model, we combine the training sets of 12 DOF and 120 DOF to train the baseline model. For the GAN-based model, We use the training set of 12 DOF to train the task model and train the GAN-based model to transfer signals from the 120 DOF domain to the 12 DOF domain. Then, we test the trained task model on the 120 DOF test set. Since the same task model is used, the performance of the GAN-based model on the 12 DOF test set is the same. For the proposed Transfer-AE (both mode1 and mode2), we use a CNN-based structure with optimal hyperparameters to perform stage 1 and stage 2 training using the training sets of 12 DOF and 120 DOF, respectively. **Table 13** summarizes the impact localization and quantification metrics of each model on the 12DOF and 120DOF test sets.

Table 13. The metrics of models on the test sets of 12DOF and 120DOF.

Model		Location	Quantification	
		F1-score	MAE	MAPE
Baseline	12 DOF	1.000	736.196	0.0725
	120 DOF	1.000	1835.970	0.1799
WGAN-GP	12 DOF	1.000	951.485	0.0951
	120 DOF	0.496	2106.264	0.2115
CGAN	12 DOF	1.000	951.485	0.0951
	120 DOF	1.000	1103.237	0.1109
CycleGAN	12 DOF	1.000	951.485	0.0951
	120 DOF	1.000	1065.395	0.1069
CycleWDCGAN-GP	12 DOF	1.000	951.485	0.0951
	120 DOF	1.000	1018.316	0.0815
Tansfer-AE (mode1)	12 DOF	1.000	**309.165**	**0.0312**
	120 DOF	1.000	269.638	0.0271
Tansfer-AE (mode2)	12 DOF	1.000	453.334	0.0459
	120 DOF	1.000	**99.226**	**0.0100**

From **Table 11**, it can be observed that all models, except WGAN-GP, achieved 100% impact localization accuracy on both the 12 DOF and 120 DOF test sets. This is because the benchmark has only four impact locations, making the localization task simpler compared to Case 1's dome model, which includes 46 possible impact locations. Therefore, we focus the comparison of network performance on impact quantification accuracy. **In Table 13**, we have bolded and underlined the best impact quantification performances. Additionally, we also present the impact quantification accuracy of each model on the 12 DOF and 120 DOF test sets in **Figure 11**. The x-axis represents the actual impact size, while the y-axis represents the predicted impact size by each network. The orange dashed line in the subplots represents $y=x$, indicating better goodness of fit as the points get closer to the orange line. Combining **Table 13** and **Figure 11,** we can see that Transfer-AE (mode 1) achieved the best performance on the 12 DOF test set with MAE/MAPE values of 309.165/0.0312, while Transfer-AE (mode 2) achieved the best performance on the 120 DOF test set with MAE/MAPE values of 99.226/0.0100. While from **Figure 11**, it can be observed that on the 12 DOF and 120 DOF test sets, the black and purple points representing Transfer-AE are closer to the orange line compared to other models. In the IASC-ASCE benchmark structure, we obtained analysis results consistent with those on the Dome structure in Case 1. Specifically, the two proposed modes of Transfer-AE achieved state-of-the-art (SOTA) performance in structural impact localization and quantification tasks compared to current mainstream DA-based transfer learning models.



(a) 12 DOF test set (b) 120 DOF test set

Figure 11. The impact quantification results of models on the test sets of 12DOF and 120DOF.

5. Discussion

Based on the results of the case study, this section will discuss two issues regarding proposed Transfer-AE: the further transfer modes for structural digital twins and how Transfer-AE can be further applied in digital twins.

Firstly, the two Transfer-AE modes proposed in the study validate the feasibility of a transfer paradigm for digital twin tasks based on autoencoders, which relies on the application of latent spaces. This approach is an extrapolation and application of generative paradigms and unsupervised anomaly detection paradigms. Therefore, besides autoencoders, the other two generative paradigms, namely Generative Adversarial Networks (GANs) and Diffusion, can theoretically also be designed to establish task transfer models for digital twins. In further research, we plan to explore how to update the Transfer-AE for structural digital twins based on other generative AI technologies. However, the Variational Autoencoders (VAEs) based on autoencoders are also applicable, but this study did not employ VAE-based models. This is because, compared to AEs, VAEs introduce an additional constraint that the latent space follows a normal distribution. This is beneficial for increasing the complexity of mappings in generative tasks, thereby enhancing diversity. However, in transfer tasks, this diversity can increase the difficulty of the network of finding a fixed mapping to bridge the distribution differences between new and old data. In fact, a Transfer-VAE model on the basis of Transfer-AE was attempted in case study, but the results were poor, hence it was not included in the case study.

Additionally, it should be noted that the study proposes two Transfer-AE modes. Mode 1 has a fixed structure with fast inference speed, but training becomes difficult and costly when new distributions are introduced. Conversely, Mode 2 introduces new structures with the introduction of new distributions, making training simpler and faster, but at the cost of reduced inference speed. The research currently only involves two distributions; however, when more distributions are involved, Modes 1 and 2 can be combined to form Mode 3. In this hybrid mode, a portion of the dominant distribution data is trained using Mode 1, and on this basis, additional small-scale distributions are trained using Mode 2 by adding extra Encoders. The effects and applications of this Mode 3 are directions for our further research.

Furthermore, the Transfer-AE model proposed in this study focuses on establishing a consistent link between numerical models and physical structures in impact detection tasks. This means we can attempt a wider variety of impact types in the numerical model and train the Transfer-AE model to replace corresponding experiments on real physical structures. Consequently, the updated Transfer-AE model can be directly used for impact detection on real structures. Moreover, besides impact detection tasks, the Transfer-AE model can also be used to bridge the distribution differences between numerical models and real physical space models in other physical information aspects, such as structural modulus damage, missing structural components, and other anomalies. Transfer-AE can accomplish the task of detecting these kinds of structural anomalies like structural damage localization and quantification for such scenarios. Furthermore, Transfer-AE can be used as a foundational model in conjunction with inversion models such as genetic algorithms. It can equate the structural responses obtained from real structure monitoring to responses in numerical space, facilitating the update of numerical models. This continuous association between numerical models and real physical structures enables the realization of digital twins' roles in construction process monitoring, construction progress control, and long-term structural monitoring.

6. Conclusions

The study proposes a novel autoencoder-based impact detection model Transfer-AE for structural digital twin, the conclusions of the study are as follows:

- The study introduces a transfer learning paradigm for structural digital twins based on an autoencoder architecture and designs an artificial neural network structure named Transfer-AE. This structure takes structural responses as input and synchronously outputs predicted impact magnitude and location, adaptively compensating for distribution differences between numerical models and real physical structures, enabling consistent estimation for the same impact event.
- Based on the proposed Transfer-AE structure, Transfer-AE (mode 1) was designed with a fixed network structure. When updating with new distribution data, it requires alternating training with samples from both old and new distributions. Its fixed structure offers fast inference speed but comes with high training costs and difficulty. In the 3D dome structure case study, CNN-based Transfer-AE (mode 1) achieved the best accuracy in detecting and locating impacts on both numerical and physical test sets, with F1-scores of 0.9903/0.8598 and MAPEs of 0.0120/0.1063. In the IASC-ASCE benchmark structure case study, Transfer-AE (mode 1) achieved the best impact quantification accuracy on the 12 DOF model with MAE/MAPE of 309.165/0.0312.
- Transfer-AE (mode 2) was designed with a variable network structure. When updating with new distribution data, it freezes the parameters of the previously trained Transfer-AE and duplicates its Encoder module, using only new data to train new parameters. This simplifies and speeds up training with new samples but increases model complexity and decreases inference speed as more modules are added. In the 3D dome structure case study, CNN-based Transfer-AE (mode 2) showed better performance in impact quantification on both numerical and physical test sets, achieving MAPEs of 0.0033/0.1016, while LSTM-based Transfer-AE (mode 2) had better accuracy in impact location, with F1-scores of 0.9432/0.8459. In the IASC-ASCE benchmark structure case study, Transfer-AE (mode 2) achieved the best impact quantification accuracy on the 120 DOF model with MAE/MAPE of 99.226/0.0100.

The study still has some shortcomings. Firstly, the structural response signals from the real physical structure contain significant noise, making it difficult for the network to obtain accurate results in impact detection and localization using only these signals. In further research, we will optimize the physical structure and employ experimental methods and data processing techniques to minimize the noise as much as possible. Additionally, it should be noted that this study only investigated the scenario of impacts applied to the nodes of the geodesic dome. Therefore, in the future, it will be necessary to expand the application scenarios of the model to determine whether impacts applied to any location on the dome can still be accurately detected. Moreover, the study focuses solely on the impact localization and quantification of undamaged structures and does not consider the complex scenarios of structural damage. In the future, the quantification and transfer of structural damage will also be a key area for further research. Furthermore, future research will explore the continuous learning of Transfer-AE in the face of multi-distribution data, such as combining mode 1 and mode 2 into mode 3 as discussed in section 5, and investigating its performance and ways to enhance it.

Acknowledgements

The authors gratefully acknowledge the support of this research by a Space Technology Research Institutes Grant (number 80NSSC19K1076) from NASA's Space Technology Research Grants Program, the start-up grant at Nanyang Technological University, Singapore (03INS001210C120), and the Ministry of Education Tier 1 Grants, Singapore (No. RG121/21 and No. RS04/21).

References

[1] V. R. Gharehbaghi, E. Noroozinejad Farsangi, M. Noori, T. Y. Yang, S. Li, A. Nguyen, C. Málaga-Chuquitaype, P. Gardoni, & S. Mirjalili. (2022). A critical review on structural health monitoring: Definitions, methods, and perspectives. Archives of computational methods in engineering, vol.29, no.4, pp.2209-2235.

[2] C. Dutta, J. Kumar, T. K. Das, &S. P. Sagar. (2021). Recent advancements in the development of sensors for the structural health monitoring (SHM) at high-temperature environment: A review. IEEE Sensors Journal, vol.21, no.14,pp.15904-15916.

[3] M. Abdulkarem, K. Samsudin, F. Z. Rokhani, & M. F. A Rasid. (2020). Wireless sensor network for structural health monitoring: A contemporary review of technologies, challenges, and future direction. Structural health monitoring, vol.19, no.3, pp.693-735.

[4] P. F. Giordano, S. Quqa, & M. P. Limongelli. (2023). The value of monitoring a structural health monitoring system. Structural safety, 100, 102280.

[5] P. Negi, R. Kromanis, A. G. Dorée, & K. M. Wijnberg. (2024). Structural health monitoring of inland navigation structures and ports: a review on developments and challenges. Structural Health Monitoring, 23(1), 605-645.

[6] F. Zonzini, M. M. Malatesta, D. Bogomolov, N. Testoni, A. Marzani, & L. De Marchi. (2020). Vibration-based SHM with upscalable and low-cost sensor networks. IEEE Transactions on Instrumentation and Measurement, vol.69, no.10, pp.7990-7998.

[7] M. A. Rahman, H. Taheri, F. Dababneh, S. S. Karganroudi, & S. Arhamnamazi. (2024). A review of distributed acoustic sensing applications for railroad condition monitoring. Mechanical Systems and Signal Processing, vol.208, pp.110983.

[8] B. Torres, I. Payá-Zaforteza, P. A. Calderón, & J. M. Adam. (2011). Analysis of the strain transfer

in a new FBG sensor for structural health monitoring. Engineering Structures, vol.33, no.2, pp.539-548.
[9] J. R. LeClerc, K. Worden, W. J. Staszewski, & J. Haywood. (2007). Impact detection in an aircraft composite panel—A neural-network approach. Journal of sound and vibration, vol.299, no.3, pp.672-682.
[10] S. N. Razavi, & C. T. Haas. (2010). Multisensor data fusion for on-site materials tracking in construction. Automation in construction, vol.19, no.8, pp.1037-1046.
[11] H. Batool, J. Xu, A. U. Rehman, & H. Hamam. (2024). Design Pattern and Challenges of Federated Learning with Applications in Industrial Control System. Journal on Artificial Intelligence.
[12] G. Xu, Q. Zhang, Z. Song, & B. Ai. (2023). Relay-Assisted Deep Space Optical Communication System Over Coronal Fading Channels. IEEE Transactions on Aerospace and Electronic Systems, 59(6), 8297-8312. doi: 10.1109/TAES.2023.3301463
[13] S. M. Saqib, M. Z. Asghar, M. Iqbal, A. Al-Rasheed, M. A. Khan, Y. Ghadi, & T. Mazhar. (2024). DenseHillNet: a lightweight CNN for accurate classification of natural images. PeerJ Computer Science, 10, e1995.
[14] F. Dipietrangelo, F. Nicassio, & G. Scarselli. (2023). Structural Health Monitoring for impact localisation via machine learning. Mechanical Systems and Signal Processing, 183, 109621.
[15] Y. Ye, C. Huang, J. Zeng, Y. Zhou, & F. Li. (2023). Shock detection of rotating machinery based on activated time-domain images and deep learning: An application to railway wheel flat detection. Mechanical Systems and Signal Processing, 186, 109856.
[16] M. Chiachío, M. Megía, J. Chiachío, J. Fernandez, & M. L. Jalón. (2022). Structural digital twin framework: Formulation and technology integration. Automation in Construction, vol.140, pp.104333.
[17] Y. Gao, H. Li, G. Xiong, & H. Song. (2023). AIoT-informed digital twin communication for bridge maintenance. Automation in Construction, vol.150, pp.104835.
[18] F. Jiang, L. Ma, T. Broyd, & K. Chen. (2021). Digital twin and its implementations in the civil engineering sector. Automation in Construction, vol.130, pp.103838.
[19] B. G. Pantoja-Rosero, R. Achanta, & K. Beyer. (2023). Damage-augmented digital twins towards the automated inspection of buildings. Automation in Construction, vol.150, pp.104842.
[20] M. Torzoni, M. Tezzele, S. Mariani, A. Manzoni, & K. E. Willcox. (2024). A digital twin framework for civil engineering structures. Computer Methods in Applied Mechanics and Engineering, 418, 116584.
[21] W. AlBalkhy, D. Karmaoui, L. Ducoulombier, Z. Lafhaj, & T. Linner. (2024). Digital twins in the built environment: Definition, applications, and challenges. Automation in Construction, 162, 105368.

[22] P. Gardner, X. Liu, & K. Worden. (2020). On the application of domain adaptation in structural health monitoring. Mechanical Systems and Signal Processing, 138, 106550.

[23] D. Sun, Z. Meng, Y. Guan, J. Liu, W. Cao, & F. Fan. (2023). Intelligent fault diagnosis scheme for rolling bearing based on domain adaptation in one dimensional feature matching. Applied Soft Computing, 146, 110669. "

[24] F. Shang, F. Sun, & J. Wen. (2024). An adaptive fault detection model based on variational auto-encoders and unsupervised transfer learning. Applied Soft Computing, 157, 111515.

[25] Q. Xu. (2014). Impact detection and location for a plate structure using least squares support vector machines. Structural Health Monitoring, vol.13, no.1, pp.5-18.

[26] I. Tabian, H. Fu, & Z. Sharif Khodaei. (2019). A convolutional neural network for impact detection and characterization of complex composite structures. Sensors, vol.19, no.22, pp.4933.

[27] Y. Sai, X. Zhao, L. Wang, &D. Hou. (2020). Impact localization of CFRP structure based on FBG sensor network. Photonic Sensors, vol.10, no.1, pp.88-96.

[28] Y. Tan, &L. Zhang. (2020). Computational methodologies for optimal sensor placement in structural health monitoring: A review. Structural Health Monitoring, vol.19, no.4, pp.1287-1308.

[29] C. Huang, C. Tao, H. Ji, & J. Qiu. (2023). Impact force reconstruction and localization using Distance-assisted Graph Neural Network. Mechanical Systems and Signal Processing, vol.200, pp.110606.

[30] P. Seventekidis, & D. Giagopoulos. (2021). A combined finite element and hierarchical Deep learning approach for structural health monitoring: Test on a pin-joint composite truss structure. Mechanical Systems and Signal Processing, vol.157, pp.107735.

[31] Z. Zhang, &C. Sun. (2021). Structural damage identification via physics-guided machine learning: a methodology integrating pattern recognition with finite element model updating. Structural Health Monitoring, vol.20, no.4, pp.1675-1688.

[32] L. Colombo, D. Oboe, C. Sbarufatti, F. Cadini, S. Russo, & M. Giglio. (2021). Shape sensing and damage identification with iFEM on a composite structure subjected to impact damage and non-trivial boundary conditions. Mechanical Systems and Signal Processing, vol.148, pp.107163.

[33] L. Massari, E. Schena, C. Massaroni, P. Saccomandi, A. Menciassi, E. Sinibaldi, & C. M. Oddo. (2020). A machine-learning-based approach to solve both contact location and force in soft material tactile sensors. Soft robotics, vol.7, no.4, pp.409-420.

[34] J. Poole, P. Gardner, N. Dervilis, L. Bull, & K. Worden. (2023). On statistic alignment for domain adaptation in structural health monitoring. Structural Health Monitoring, 22(3), 1581-1600.

[35] V. Giglioni, J. Poole, I. Venanzi, F. Ubertini, & K. Worden. (2024). A domain adaptation approach to damage classification with an application to bridge monitoring. Mechanical Systems and Signal Processing, 209, 111135.

[36] L. Ge, &A. Sadhu. (2024). Domain adaptation for structural health monitoring via physics-informed and self-attention-enhanced generative adversarial learning. Mechanical Systems and Signal Processing, 211, 111236.
[37] F. Luleci, F. N. Catbas, & O. Avci. (2023). CycleGAN for undamaged-to-damaged domain translation for structural health monitoring and damage detection. Mechanical Systems and Signal Processing, 197, 110370.
[38] C. Han, H. Yang, T. Ma, S. Wang, C. Zhao, & Y. Yang. (2024). CrackDiffusion: A two-stage semantic segmentation framework for pavement crack combining unsupervised and supervised processes. Automation in Construction, vol.160, pp.105332.
[39] W. H. L. Pinaya, S. Vieira, R. Garcia-Dias, &A. Mechelli. (2020). Autoencoders. Machine learning. Academic Press. pp. 193-208.
[40] V. Bucinskas, A. Dzedzickis, N. Sesok, I. Iljin, E. Sutinys, M. Sumanas, & I. Morkvenaite-Vilkonciene. (2022). Automatic quality detection system for structural objects using dynamic output method: Case study Vilnius bridges. Structural Health Monitoring, 21(6), 2505-2517.
[41] C. Han, T. Ma, J. Huyan, X. Huang, & Y. Zhang. (2021). CrackW-Net: A novel pavement crack image segmentation convolutional neural network. IEEE Transactions on Intelligent Transportation Systems, 23(11), 22135-22144.
[42] F. Bayram, P. Aupke, B. S. Ahmed, A. Kassler, A. Theocharis, & J. Forsman. (2023). DA-LSTM: A dynamic drift-adaptive learning framework for interval load forecasting with LSTM networks. Engineering Applications of Artificial Intelligence, 123, 106480.
[43] C. Han, T. Ma, L. Gu, J. Cao, X. Shi, W. Huang, & Z. Tong. (2022). Asphalt pavement health prediction based on improved transformer network. IEEE Transactions on Intelligent Transportation Systems, 24(4), 4482-4493.
[44] S. Hochreiter, & J. Schmidhuber. (1997). Long short-term memory. Neural computation, vol.9, no.8, pp.1735-1780.
[45] Y. Zhang, Y. Miyamori, S. Mikami, & T. Saito. (2019). Vibration-based structural state identification by a 1-dimensional convolutional neural network. Computer-Aided Civil and Infrastructure Engineering, vol.34, no.9, pp.822-839.
[46] M. Ribeiro, A. E. Lazzaretti, & H. S. Lopes. (2018). A study of deep convolutional auto-encoders for anomaly detection in videos. Pattern Recognition Letters, vol.105, pp.13-22.
[47] J. Xin, Y. Jiang, J. Zhou, L. Peng, S. Liu, & Q. Tang. (2022). Bridge deformation prediction based on SHM data using improved VMD and conditional KDE. Engineering Structures, vol.261, pp.114285.
[48] C. Han, T. Ma, J. Huyan, Z. Tong, H. Yang, & Y. Yang. (2024). Multi-stage generative adversarial networks for generating pavement crack images. Engineering Applications of Artificial Intelligence,

vol.131, pp.107767.
[49] M. Ghifary, D. Balduzzi, W. B. Kleijn, & M. Zhang. (2016). Scatter component analysis: A unified framework for domain adaptation and domain generalization. IEEE transactions on pattern analysis and machine intelligence, 39(7), 1414-1430.
[50] M. P. Saka. (2007). Optimum geometry design of geodesic domes using harmony search algorithm. Advances in Structural Engineering, vol.10, no.6, pp.595-606.
[51] J. Ching, & Y. C. Chen. (2007). Transitional Markov chain Monte Carlo method for Bayesian model updating, model class selection, and model averaging. Journal of engineering mechanics, 133(7), 816-832.
[52] Y. Bao, & H. Li. (2021). Machine learning paradigm for structural health monitoring. Structural health monitoring, 20(4), 1353-1372.
[53] F. Baek, D. Kim, S. Park, H. Kim, & S. Lee. (2022). Conditional generative adversarial networks with adversarial attack and defense for generative data augmentation. Journal of Computing in Civil Engineering, 36(3), 04022001.
[54] J. M. Caicedo, S. J. Dyke, & E. A. Johnson. (2004). Natural excitation technique and eigensystem realization algorithm for phase I of the IASC-ASCE benchmark problem: Simulated data. Journal of Engineering Mechanics, 130(1), 49-60.
[55] D. Bernal, S. J. Dyke, H. F. Lam, & J. L. Beck. (2002). Phase II of the ASCE benchmark study on SHM. In Proceedings of the 15th ASCE engineering mechanics conference. Columbia University.
[56] Y. He, L. Zhang, Z. Chen, & C. Y. Li. (2023). A framework of structural damage detection for civil structures using a combined multi-scale convolutional neural network and echo state network. Engineering with Computers, 39(3), 1771-1789.