Data Reconstruction Based on a Multi-branch Deep Neural Network

Hongyun Tian[1,2], and Xiaomin Huang[1,2*]

[1] Faculty of Civil Engineering and Mechanics, Kunming University of Science and Technology, Kunming, 650500, China
[2] Intelligent Infrastructure Operation and Maintenance Technology Innovation Team of Yunnan Provincial, Department of Education, Kunming University of Science and Technology, Kunming 650500, China
huangxm.yn@kust.edu.cn

Abstract. Structural health monitoring systems measure structural deformations, vibrations, stress, etc., in real-time through sensor networks, comprehensively assessing the health of structures. The quality of monitoring data directly influences the accuracy of structural assessments and maintenance decisions. However, data loss is inevitable during the long-term monitoring of large structures. To address this issue, this paper proposes a Compressive Sensing-Convolutional Transformer Networks (C-CTNet) based multi-channel data reconstruction method. The neural network model comprises sampling, feature embedding and transformation, and reconstruction modules. The sampling module utilizes a mask matrix to sample the response matrix. The sampled matrix is then converted into a high-dimensional feature matrix in the feature embedding, and transformation module and segmented into submatrices. During the process, information is extracted, and dimensionality is reduced by learned convolution kernels. In the reconstruction phase, features are projected into a linear initialization module, CNN stem, and transformer stem, respectively. The initialization module mimics traditional compressive sensing reconstruction but generates initial reconstructions in a learnable and efficient manner. The CNN and transformer stem compute and effectively integrate local and long-range features. Additionally, a progressive strategy and window-based transformer model blocks reduce parameter count and computational complexity. Experimental results demonstrate that the proposed neural network model can effectively reconstruct data under random loss conditions. Comparisons with ablation models confirm the critical role of the three branches in the task.

Keywords: Structural health monitoring, compressive sensing, transformer, CNN, data reconstruction.

W. Guo et al. (eds.), *Proceedings of the 3rd International Conference on Green Building, Civil Engineering and Smart City (GBCESC 2024)*, Advances in Engineering Research 264,
https://doi.org/10.2991/978-94-6463-728-1_72

1 Introduction

In recent years, SHM has become a prominent topic across civil, mechanical, automotive, and aerospace engineering, especially rapidly advancing in civil engineering [1]. In practice, data loss in SHM systems is nearly inevitable due to sensor failures, system malfunctions, and network transmission losses [2]. To address this, efforts to reconstruct missing data to compensate for incomplete responses are increasingly emphasized, alongside improvements in hardware reliability.

In multi-channel data acquisition via compressive sensing, responses are encoded through linear random measurements for sampling and compression. Given the typical sparsity of responses in the transform domain, far fewer samples are required for reconstruction than mandated by the Shannon-Nyquist sampling theorem [3]. In 2011, Bao et al. used CS to compress accelerometer signals, significantly reducing data transmission bandwidth and storage needs [4]. Following this application, CS has been extensively employed for data loss recovery.

In compressive sensing theory, the response $Y \in \mathbb{R}^M$ from random linear measurements can be expressed as:

$$Y = \Phi X + e, \tag{1}$$

where $X \in \mathbb{R}^N$ is the original response, e is the measurement noise, $\Phi \in \mathbb{R}^{M \times N}$ is the measurement matrix, M/N is the measurement ratio, and $N \gg M$.

Due to measurement uncertainties, CS reconstruction is an ill-posed inverse problem. However, CS theory demonstrates that if the response s is sparse (i.e., the response has a sparse representation in a certain basis Ψ or redundant dictionary D , $X = \Psi s$ or $X = Ds$), and Φ satisfies the Restricted Isometry Property (RIP), the coefficients α can be reconstructed through an l_1 optimization problem:

$$\begin{aligned} Y &= \Phi \Psi s + e = \Theta s + e, \\ \hat{s} &= \min \|s\|_1 \Rightarrow \|X - \Theta \hat{s}\|_2 \le \varepsilon. \end{aligned} \tag{2}$$

where Θ is the sensing matrix, and ε is the boundary of the measurement error level, $\|e\|_2 \le \varepsilon$.

Traditional compressive sensing methods like OMP [5], gradient projection [6], and Lasso [7] address sparse regularization problems by exploring data sparsity. In the SHM field, Bao and others have utilized CS technology and l_1 optimization to restore data in wireless sensor networks, demonstrating its recovery accuracy [8]. Zou and others implemented data recovery on the Imote2 platform through compressive sensing, successfully applying it in bridge testing [9]. While these algorithms are theoretically sound and interpretable, their high iteration count increases computational costs.

As neural networks progressively reveal their advantages in feature extraction and nonlinear mapping, their application in compressive sensing has significantly improved data reconstruction efficiency. Notably, Bao and Tang have enhanced compressive sensing data reconstruction under low sampling ratios by integrating prior knowledge into neural networks[10-11]. However, while traditional CNNs excel at processing local image features, they struggle to capture global information. In

contrast, transformers excel at handling long-range dependencies through their multi-head self-attention mechanism, and are increasingly used in various computer vision tasks. Based on both neural networks' strengths, CSformer successfully combines CNNs and transformers, significantly enhancing image compression reconstruction efficiency [12]. Inspired by CSformer's outstanding performance and innovative approach to image processing, we modified the neural network structure and named it C-CTNet. This network not only suits multi-channel data compression reconstruction, but also retains efficient reconstruction capabilities and enhances feature extraction, demonstrating excellent reconstruction precision and generalization ability.

2 Network Architecture

Figure. 1 shows the network architecture of C-CTNet. The detailed descriptions of each module branch are as follows.

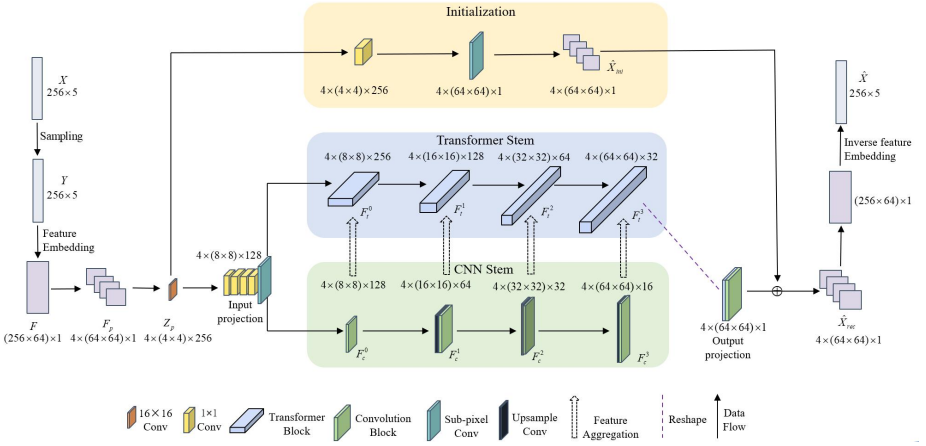

Fig. 1. The employed network architecture.

2.1 Sampling

Assume that K sensors (channels) are arranged on a structure, and each channel of the sensor uniformly samples N points over a duration T. Collecting all the data results in an $N\times K$ response matrix X. The sampling module is designed to sample the response matrix X using a mask matrix P, resulting in a sampled response matrix $Y = P \odot X$, where all elements in the mask matrix P are generated according to a Bernoulli distribution with a sampling ratio γ, taking values of 0 or 1. The formula is as follows:

$$f(p_{nk}) = \begin{cases} \gamma & p_{nk} = 1 \\ 1-\gamma & p_{nk} = 0 \end{cases}, p_{nk} \in P \in \mathbb{Z}^{N\times K}, \tag{3}$$

2.2 Feature Embedding and Transformation

To integrate and extract multi-channel information, the data is embedded from a low-dimensional space into a higher-dimensional representation space, facilitating expansion in the channel dimension. Therefore, feature embedding is performed on Y. The Y matrix is convolved using M convolutional layers with kernel sizes of $3\times K$, a stride of 1, and wrap-around padding. This operation produces a higher-dimensional and more feature-rich feature matrix F, with a size of $N\times M$, where $M > K$. In this paper, the process is illustrated in Figure. 2. In this case, $N = 256$, $K = 5$, and $M = 64$. To reduce computational complexity and burden, F is divided into multiple small Submatrix sets $F_p = \{F_1, F_2, \ldots\ldots, F_i\}$. Each submatrix F_i has a size of $B\times B$, and the number of submatrices is $(N/B)\times(M/B)$. An unbiased convolutional layer consisting of m filters with sizes of $b\times b$ and a stride of b is applied to F_i. After applying the convolution operation, the final sub-transformation matrix Z_i is obtained, with a size of $m\times(B/b)\times(B/b)$. This study's parameter values are $B = 64$, $b = 16$, and $m = 256$.

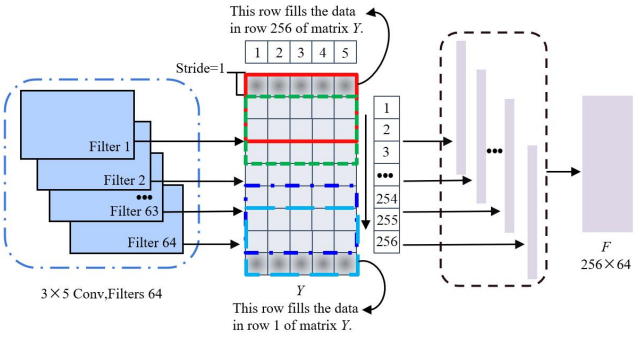

Fig. 2. Convolution-based feature embedding process.

2.3 Initialization

Given sampled values, traditional compressive sensing typically achieves initial reconstruction using the following method: $\hat{X} = \Phi^{+}Y$, where $\hat{X}$ is the reconstruction of X, and Φ^{+} is the pseudoinverse of Φ . During the initialization process, a $1\times1\times m$ convolution is used instead, increasing the spatial dimension of the Z_i space to B^2. Subsequently, a sub-pixel convolution layer is utilized to obtain the initial reconstruction $\hat{X}_{ini}$. In summary, convolution and sub-pixel convolution efficiently generate each initial reconstruction by leveraging residual information. This is a more effective method as the output is directly a tensor rather than a vector. The initial reconstruction is finally added to the recovery results of the main network, thereby supplementing and optimizing the main network.

2.4 Main Recovery Network

Z_i applies an input projection containing multiple layers of 1×1 convolutions to incrementally increase the dimensionality, followed by a sub-pixel convolution operation with a doubling upsampling ratio to generate the input feature F_{in}. The main recovery network is comprised of a CNN stem and a transformer stem, with the input feature F_{in} sizes for the CNN and transformer corresponding to $H_0 \times W_0 \times C_0$ and $(H_0 \times W_0) \times C_0$ respectively. Each stem consists of four blocks with upsampling layers, progressively reconstructing features until they align with the dimensions of F_i.

CNN Stem. Each convolutional block of the CNN stem consists of two 3×3 convolutional layers (with padding of 1, and the number of output channels equal to the number of input channels), a ReLU activation layer, and a batch norm layer, maintaining the same spatial dimensions and channel count. To expand the spatial dimensions, an upsampling module is added before the convolutional block, which utilizes bicubic upsampling to increase spatial dimensions, followed by a 1×1 convolutional layer that halves the number of channels. Thus, the output feature of the CNN stem is denoted as $F_c^i \in R^{H_i \times W_i \times C_i}$, where $H_i = 2^i \times H_0$, $W_i = 2^i \times W_0$, and $C_i = C_0/2^i$, with $i \geq 0$. Finally, the output features are fused with the transformer features, introducing a locality mechanism for the transformer stem, making full use of the detailed spatial information provided by the CNN. This not only enhances the model's ability to capture local features in time series but also significantly improves the efficiency and accuracy of the overall inference process.

Transformer Stem. The input features for each transformer block result from the fusion of local features from the CNN and global features from the transformer through concatenation. The only difference is that the input fusion feature F_a^0 for the first transformer block is created by concatenating CNN local features F_c^0 and transformation features F_{in}. The feature fusion process is expressed as follows: $F_a^j \in \mathbb{R}^{(H_j \times W_j) \times C_j}$, where $H_j = 2^j \times H_0$, $W_j = 2^j \times W_0$, $C_j = 2C_0/2^j$, $j \geq 0$. The transformer block consists of a batch norm layer, multi-head self-attention (MSA), multi-layer perceptron (MLP), and their residual connections, with multiple transformer networks stacked together. To address computational complexity, a window-based transformer strategy is employed. First, the aggregated features are added to the learnable positional encodings $E \in \mathbb{R}^{(H_j \times W_j) \times C_j}$, and then split into $L \times L$ non-overlapping small windows. The dimensions of these windows are $(H_j W_j / L^2) \times L^2 \times C_j$, where $H_j W_j / L^2$ represents the total number of windows. Finally, in each L^2 window, features $F_t^{win} \in \mathbb{R}^{L^2 \times (C_j/h)}$ are computed through the Multi-Head Self-Attention mechanism, where h denotes the number of heads in the Multi-Head Self-Attention. During this process, the query, key, and value matrices are computed first:

$$Q = F_t^{win} \times W_Q, K = F_t^{win} \times W_K, V = F_t^{win} \times W_V, \tag{4}$$

Here, the sizes of W_Q, W_K, and W_V are $(C_j/h) \times d$. The self-attention mechanism can be expressed as:

$$O\left(F_t^{win}\right) = \left(\sigma\left(\frac{QK^T}{\sqrt[2]{d}} + E_r\right)\right)V, \tag{5}$$

where $O(\bullet)$ represents the self-attention operation, $\sigma(\bullet)$ is the softmax function, and E_r is the learnable relative positional encoding. To obtain the output, the multi-head self-attention operation is performed in parallel h times, and all results are concatenated. The window-based MSA scheme significantly reduces computational and GPU memory costs. Subsequently, the output is passed to an MLP consisting of two fully connected layers and is transformed non-linearly using the GELU activation function. The insertion of Layer Normalization is $\tau(\bullet)$. The entire transformer process can be described as follows:

$$\begin{aligned} F_a^j &= F_a^j + E, \\ F_t^j &= MSA\left(\tau\left(F_a^j\right)\right) + F_a^j, \\ F_t^j &= MLP\left(\tau\left(F_t^j\right)\right) + F_t^j. \end{aligned} \tag{6}$$

The transformer stem restores global feature information by fusing CNN and transformer features, along with a window-based self-attention mechanism. This fusion enhances the performance and efficiency of the data reconstruction task. When the transformer features reach the size of F_i, an output projection first maps the features to a single channel to facilitate main recovery. Subsequently, the initial reconstruction is added to the main recovery to generate the final submatrix $\hat{X}_{rec}$. Finally, the ultimate reconstruction response matrix $\hat{X}$ is obtained through merging and feature restoration operations.

2.5 Loss Function and Evaluation Metrics

The goal of this study is to reconstruct missing responses. Given an incomplete response matrix as input, the model regresses to a complete reconstruction response matrix. The *MAE* is chosen as the loss function. It optimizes the parameters of C-CTNet by minimizing the average absolute error between the output reconstruction values and the true values:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{X}_i - X_i\right|, \tag{7}$$

where N is the number of samples with missing data, $\hat{X}_i$ and X_i are the reconstructed values and true values, respectively.

To quantify the reconstruction performance of the model, *RMSE* and R^2 are used as performance evaluation metrics. *RMSE* measures the difference between the model's reconstructed values and the true values, reflecting the magnitude of the prediction error. R^2 is a statistical measure representing the variation between the model's reconstructed values and the true values, thereby reflecting the model's explanatory power. Specifically, the smaller the *RMSE* and the larger the R^2, the better the model's reconstruction performance. *RMSE* and R^2 can be represented as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}\left(\hat{X}_i - X_i\right)^2}{N}},$$
$$R^2 = 1 - \frac{\sum_{i=1}^{N}\left(X_i - \hat{X}_i\right)^2}{\sum_{i=1}^{N}\left(X_i - \overline{X}_i\right)^2}. \tag{8}$$

where N is the number of missing data, X_i are the true values, $\hat{X}_i$ are the reconstructed values, and $\overline{X}_i$ is the average value.

3 Experimental Validation

To evaluate the effectiveness and robustness of C-CTNet in reconstructing multi-channel response data from structures, this study conducted random loss experiments. These experiments utilized five-channel accelerometer data from a three-span continuous beam. Additionally, ablation experiments were carried out, comparing with ablation models to validate the importance of each branch of the C-CTNet. These experiments are designed to ensure the network's reliability and overall performance under data loss conditions.

3.1 Data Description

The dataset originates from Project 3 of the 3rd International Competition for Structural Health Monitoring. It documents the structural responses of a three-span continuous bridge under various stimuli, including moving vehicle traffic, Gaussian noise, and pulse loads [13-14]. As shown in Figure. 3, sourced from the competition's official website, the model was equipped with 12 sensors, including 5 accelerometers and 7 strain gauges. Only measurements from accelerometers at positions 1 through 5 (left to right) were used for this experiment. These measurements spanned 1000 seconds with a sampling rate of 100Hz, creating an accelerometer response matrix of 100,000 × 5. Effective data preprocessing, such as cleaning and normalization, is crucial for enhancing the efficiency and capability of deep learning tasks. Thus, measurements were preprocessed with a low-pass filter at a cutoff frequency of 35Hz. The input and output for the deep learning model are not the complete time series of the original samples but rather a set of equal-length short sequences. A sliding window method with a step size of 1 was used to divide the original sequence into multiple short sequences, ensuring sufficient samples for experimentation, with each channel containing 256 data points over 2.56 seconds. Each sample was normalized:

$$X'(j) = \frac{X(j) - \overline{X}(j)}{\sigma(j)}. \tag{9}$$

where $X(j)$, $\overline{X}(j)$ and $\sigma(j)$ represent the j-th sample, its mean, and its standard deviation, respectively. A total of 99745 samples were obtained, with 70% used for training, 15% for validation, and the remaining 15% for testing.

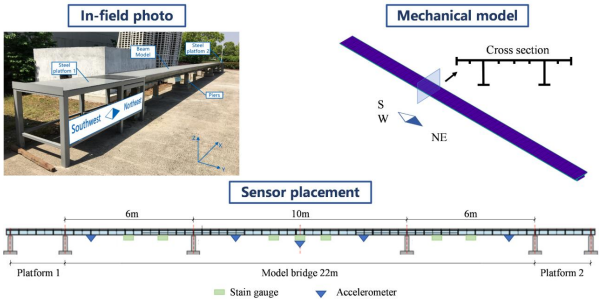

Fig. 3. The bridge model layout and sensor placement.

3.2 Training Details

The output feature dimension C_0 for the input projection was set to 128. For all transformer blocks, the window size for the window-based multi-head self-attention was set to $L \times L = 8 \times 8$, with each block stacking five transformer networks. The model was trained using an Nvidia 4060Ti graphics card on the PyTorch framework, employing the Adam optimizer for parameter optimization. During training, the random sampling ratio was 0.25, the batch size was set to 6, and the initial learning rate 2×10^{-4} was gradually reduced to 1×10^{-6} using a cosine decay strategy. A total of 65 iterations were completed over approximately 40 hours, with the model fully converging and optimally learning the mapping between input and output. The training and validation loss curves, as shown in Figure. 4, displayed a smooth decline until they stabilized after 60 iterations, with no overfitting issues observed. The best training and validation mean absolute errors were 0.0711 and 0.0723, respectively. Notably, all subsequent test results on random losses were conducted using this fully trained neural network model.

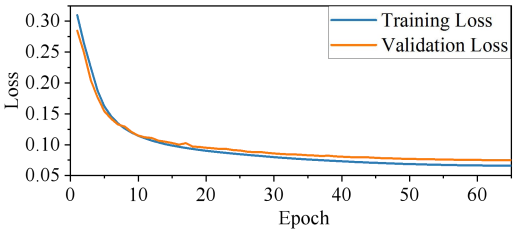

Fig. 4. Training and validation loss.

3.3 Data Reconstruction

In practical SHM systems, data loss is usually random. Thus, data was segmented and randomly discarded in this experiment, considering five different sampling ratios ranging from 0.1 to 0.9, increasing by 0.2 each time, specifically 0.1, 0.3, 0.5, 0.7, and 0.9. For these segmented losses, the number of data points lost per segment analyzed included 1, 2, 4, 8, and 16. Once the sampling ratio and segment length are

determined, the number of missing segments can be directly calculated. For example, with a 256×5 response matrix, a sampling ratio of 0.3, and a segment length of 4, the matrix will contain approximately [(1−0.3) ×256×5] /4=224 consecutive missing segments. If the number of segments is not an integer, it will be rounded.

As shown in Table 1, the method demonstrated excellent overall performance across five different segment lengths and corresponding sampling ratios. As the sampling ratio increased, *RMSE* linearly decreased, and R^2 gradually increased. Moreover, at the same sampling ratio, longer missing segments had poorer *RMSE* and R^2 compared to shorter segments. For instance, with segment lengths of 1, at sampling ratios of 0.1 and 0.3, the average *RMSE* decreased from 0.3320 to 0.0885, while the average R^2 increased from 0.8647 to 0.9887. At a constant sampling ratio of 0.3, when segment lengths were 1 and 2, the average *RMSE* increased from 0.0885 to 0.1226, and the average R^2 decreased from 0.9887 to 0.9800. These results are within acceptable ranges, indicating that the well-trained network performs well under various loss lengths and sampling ratios, showcasing its excellent applicability.

To clearly demonstrate the recovery performance of the C-CTNet during random data loss across different channels, a segment of length 4 was selected. Among the five channels of accelerometer responses, one was chosen to visualize the original and reconstructed responses over 2.56 seconds. Channel 1 was selected as its reconstruction metrics were closest to the overall average, making it representative. A sample was randomly drawn from the test set, and comparison charts of the original and reconstructed responses of Channel 1 at different sampling ratios were illustrated. Figures. 5(a), (c), and (e) show that at sampling ratios of 0.5 and 0.9, the reconstructed responses of Channel 1 almost completely overlap with the original responses and are hard to distinguish. At a sampling ratio of only 0.1, differences exist in peak points and some segments between the reconstructed and original responses, but the overall trends are still generally consistent. The scatter plots in Figures. 5(b), (d), and (f) also confirm these findings. Generally, the C-CTNet achieves a low root mean square error of 0.0536 under typical loss conditions (e.g., a sampling ratio of 0.5). Even in extreme cases, such as a sampling ratio of only 0.1, the root mean square error can reach 0.485.

Table 1. Reconstruction results of accelerometer responses under different segment lengths and sampling ratios.

Missing length	Type	Sampling ratio 0.1	0.3	0.5	0.7	0.9
1	*RMSE*	0.3320	0.0885	0.0303	0.0167	0.0232
	R^2	0.8647	0.9887	0.9986	0.9997	0.9994
2	*RMSE*	0.3517	0.1226	0.0479	0.0275	0.0246
	R^2	0.8500	0.9800	0.9969	0.9991	0.9994
4	*RMSE*	0.5317	0.1583	0.0835	0.0497	0.0335
	R^2	0.6976	0.9691	0.9908	0.9966	0.9987
8	*RMSE*	0.5370	0.2513	0.1428	0.0732	0.0451
	R^2	0.6751	0.9152	0.9709	0.9929	0.9976
16	*RMSE*	0.6988	0.3002	0.1800	0.1049	0.0637
	R^2	0.4928	0.8795	0.9603	0.9859	0.9943

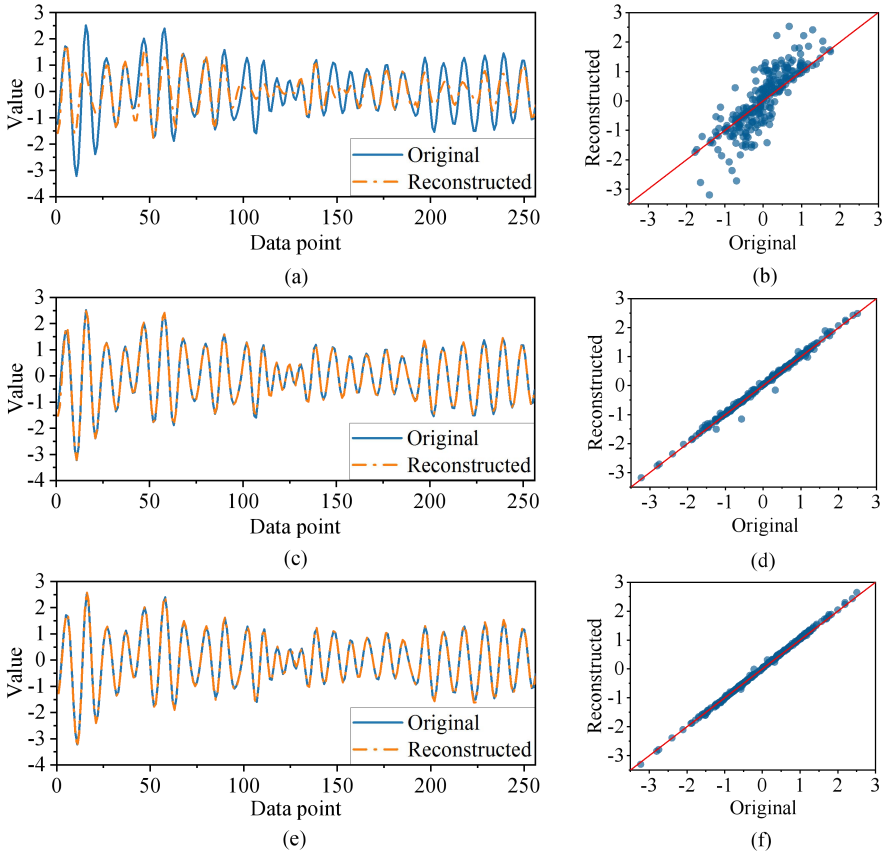

Fig. 5. Comparison of true and reconstructed responses for channel 1: (a)-(b) sampling ratio of 0.1, (c)-(d) sampling ratio of 0.5, (e)-(f) sampling ratio of 0.9.

3.4 Ablation Experiment

In this section, C-CTNet is compared with three neural network models to demonstrate the performance improvements of the adopted network. The first is the CTNet model, which is created by removing the initialization reconstruction branch from the original C-CTNet model, and using only the remaining two branches for reconstruction. Next is the C-TNet model, which is developed by removing the CNN stem and adding a 1×1 convolution before each transformer block, setting C_0 to 256, 128, 64, and 32, respectively, to maintain consistency in the channel dimensions within the transformer blocks. Finally, the C-CNet model is constructed by removing the transformer stem while keeping all other structures unchanged.

Specific results from experiments comparing the four neural network models under random loss with a segment length of 1 are presented in Table 2. The experiments show that only at a sampling ratio of 0.1 does the C-CTNet model slightly underperform compared to the comparative model C-TNet. However, C-CTNet

outperforms the other three comparison models at other sampling ratios. In summary, ablation studies demonstrate the importance of each component of the C-CTNet model in enhancing reconstruction performance, proving that each part is indispensable.

Table 2. Performance comparison of the adopted model with three ablation models at a loss length of 1, with the best results highlighted in bold.

Model	Type	Sampling ratio				
		0.1	0.3	0.5	0.7	0.9
C-CTNet	*RMSE*	0.3320	**0.0885**	**0.0303**	**0.0167**	**0.0232**
	R^2	0.8647	**0.9887**	**0.9986**	**0.9997**	**0.9994**
CTNet	*RMSE*	0.3323	0.0945	0.0381	0.0243	0.0239
	R^2	0.8655	0.9878	0.9981	0.9994	**0.9994**
C-TNet	*RMSE*	**0.3274**	0.0930	0.0366	0.0219	0.0247
	R^2	**0.8686**	0.9879	0.9981	0.9994	**0.9994**
C-CNet	*RMSE*	0.4199	0.1711	0.1057	0.1049	0.1638
	R^2	0.7952	0.9626	0.9871	0.9881	0.9689

4 Conclusion

This study employed a composite neural network architecture termed C-CTNet, which integrates compressive sensing, CNN, and transformer for data reconstruction. Integrating CNN and Transformer effectively leverages both complementary strengths, enhancing the reconstruction performance of compressive sensing. During the sampling phase, a mask matrix is used for sampling, followed by the feature embedding and transformation phase, where convolution operations are employed to embed features into the sampled response matrix and further transform it by splitting into multiple sub-matrices for additional convolutional operation. In the reconstruction phase, the Transformer network serves as the backbone, linked with the CNN features to progressively increase the spatial dimensions to reduce memory costs and computational complexity, and an initialization branch is introduced for initial reconstruction. C-CTNet demonstrated robust reconstruction capabilities in scenarios of different missing lengths and loss ratios, outperforming ablation models. Overall, C-CTNet exhibits excellent performance and potential in compressive sensing reconstruction, offering new insights and methodologies for data reconstruction in structural health monitoring.

Acknowledgement

This research was supported by the Yunnan Fundamental Research Projects (Grant No. 202301AT070394, 202301BE070001-037).

References

1. Zhang C, Mousavi A A, Masri S F, et al.: Vibration feature extraction using signal rocessing techniques for structural health monitoring: A review[J]. Mechanical Systems nd Signal Processing, 2022, 177: 109175.
2. Zheng W, Li J, Li Q, et al.: Multi‐channel response reconstruction using transformer based generative adversarial network[J]. Earthquake Engineering & Structural Dynamics, 2023, 52(11): 3369-3391.
3. Donoho D L.: Compressed sensing[J]. IEEE Transactions on information theory, 2006, 52(4): 1289-1306.
4. Bao Y, Beck J L, Li H.: Compressive sampling for accelerometer signals in structural health monitoring[J]. Structural Health Monitoring, 2011, 10(3): 235-246.
5. Tropp J A, Gilbert A C.: Signal recovery from random measurements via orthogonal matching pursuit[J]. IEEE Transactions on information theory, 2007, 53(12): 4655-4666.
6. Figueiredo M A T, Nowak R D, Wright S J.: Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems[J]. IEEE Journal of selected topics in signal processing, 2007, 1(4): 586-597.
7. Tibshirani R.: Regression shrinkage and selection via the lasso[J]. Journal of the Royal Statistical Society Series B: Statistical Methodology, 1996, 58(1):267-288.
8. Bao Y, Li H, Sun X, et al.: Compressive sampling–based data loss recovery for wireless sensor networks used in civil structural health monitoring[J]. Structural Health Monitoring, 2013, 12(1): 78-95.
9. Zou Z, Bao Y, Li H, et al.: Embedding compressive sensing-based data loss recovery algorithm into wireless smart sensors for structural health monitoring[J]. IEEE Sensors Journal, 2014, 15(2): 797-808.
10. Bao Y, Tang Z, Li H.: Compressive-sensing data reconstruction for structural health monitoring: a machine-learning approach[J]. Structural Health Monitoring, 2020, 19(1): 293-304.
11. Tang Z, Bao Y, Li H.: Group sparsity-aware convolutional neural network for continuous missing data recovery of structural health monitoring[J]. Structural Health Monitoring, 2021, 20(4): 1738-1759.
12. Ye D, Ni Z, Wang H, et al.: CSformer: Bridging convolution and transformer for compressive sensing[J]. IEEE Transactions on Image Processing, 2023.
13. Shang Z, Sun L, Xia Y, et al.: Vibration-based damage detection for bridges by deep convolutional denoising autoencoder[J]. Structural Health Monitoring, 2021, 20(4):1880-1903.
14. Jian X, Sun L, Xia Y.: Modal parameter identification for regular bridges using vehicle sensing technique: Simulation and experiment[M]//Bridge Maintenance, Safety, Management, Life-Cycle Sustainability and Innovations. CRC Press, 2021: 2182-2187.