Journal of Artificial Intelligence Practice (2023)
Clausius Scientific Press, Canada

DOI: 10.23977/jaip.2023.060409
ISSN 2371-8412 Vol. 6 Num. 4

Study on Inversion of Damage Incentives of High Pile Wharf in Inland River Based on SEResNet

Li Jia[1], Cai Fenglin[1,*], Zhu Qitao[1], Yang Tongxin[1], Zhang Qian[1]

School of Intelligent Technology and Engineering, Chongqing University of Sicence and Technology, Chongqing, 401331, China
**Corresponding author*

Keywords: The SEResNet neural network algorithm; Inland high-pile wharf; Cause of injury; invert

Abstract: Based on SEResNet neural network algorithm, the inversion model of damage incentives of inland high-piled wharf is constructed. The stress data of pile foundation under the action of damage incentives of high-piled wharf are obtained by using solid element finite element model calculation and indoor model test methods. The parameterized finite element calculation model of high-piled wharf is established by using subprocess in Python program to call MANSYS module, and verified with solid element model. The parameterized simplified finite element model meets the needs of inversion calculation. Based on the stress data samples of the pile foundation of the high-piled wharf obtained from the model test, the inversion analysis of single and multiple damage incentives is carried out. The model can identify the location, size and type of injury causative agent with good generalization ability.

1. Introduction

With the rapid development of the Chinese economy and continuous advancements in transportation, inland waterway transportation plays a significant role in logistics transportation in China. However, as an integral component of inland waterway transportation, inland high-pile wharves inevitably experience structural damage[1] during their service life due to factors such as vessel berthing, cargo stacking, and slope sliding, which in turn can trigger safety accidents. Therefore, accurately identifying the factors of damage to inland high-pile wharves holds crucial significance in ensuring the safety and smoothness of inland waterway transportation.

In recent years, the rapid development of deep learning technology has provided powerful tools for addressing the inverse problem of damage factor identification[2]. However, there is still limited quantitative analysis research on the structural damage and its causes in inland high-pile wharves. Furthermore, inland high-pile wharves exhibit characteristics such as large amount of structural elements, high rigidity, and large pile diameters. Therefore, in the analysis of damage factor identification for inland high-pile wharves, it is necessary to select key structural elements as research targets and establish a reasonable computational analysis model. The required dataset for analysis can be achieved through physical model experiments or on-site measurements. These conditions serve as the foundation and prerequisites for conducting damage factor identification analysis[3]. Pile foundations, as important load-bearing structures of wharves, not only affect the overall structural

safety but also serve as windows for observing structural damage and as a means to identify damage factors[4].

Based on the SEResNet[5] neural network algorithm, an inverse model for inferring damage factors in inland high-pile wharves is constructed. This model utilizes a parameterized modeling approach to simplify numerical model calculations. The parameterized model is validated using physical unit models and is supported by data from indoor model experiments. This approach enables the identification of the location, size, and type of damage factors, and an analysis of the model's generalization ability.

2. SEResNet neural network building

The SEResNet is constructed by embedding the SE block into the ResNet architecture[6]. It addresses the gradient vanishing problem through residual connections in the ResNet. By incorporating the SE module, the model can adaptively adjust the weights of feature channels, allowing it to better focus on important feature information and improve the overall performance of the model.

2.1. ResNet

In traditional convolutional neural networks, information is transmitted between network layers through layer-by-layer stacking. However, as the number of network layers increases, issues such as gradient vanishing or exploding can arise, making the network difficult to train. ResNet solves this problem by introducing residual blocks into the network architecture. These residual blocks add the input features to the features obtained from an identity mapping through residual connections, resulting in the final output. This allows information to directly bypass several layers in the network, mitigating the problems of gradient vanishing and exploding. Consequently, ResNet enables the network to be trained deeper, overcoming the limitations of traditional convolutional neural networks.

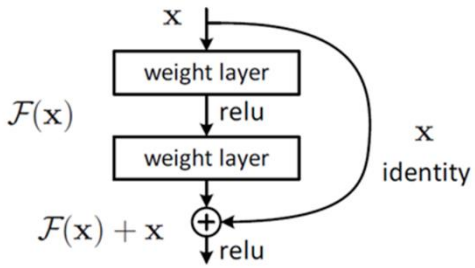


Figure 1: Structure diagram of Residual block.

The core idea of residual blocks is to introduce skip connections to bypass certain layers in the computation, allowing the network to directly learn residual functions instead of solely focusing on learning low-level features. Each residual block consists of two components: an identity mapping and a residual mapping, as shown in Figure 1. The identity mapping connects the input directly to the output, while the residual mapping applies non-linear transformations to the input by stacking multiple convolutional layers. The calculation of features from the l-th layer to the (l+1)-th layer is as follows:

$$x_{l+1} = h(x_l) + F(x_l, W_l) \tag{1}$$

where x_l and x_{l+1} represent the input and output of the residual unit, $h(x_1) = x$ represents the identity mapping, and F denotes the ReLU activation function. Due to the residual characteristics between any unit L and 1, the feature representation of a residual unit with a depth of L can be obtained through recursion:

$$x_L = x_0 + \sum_{i=0}^{L-1} F(x_i, W_i) \tag{2}$$

where x_L is the sum of all previous residual function outputs added to x_0.

For backpropagation, we can obtain the following equations assuming the loss function is ε based on the chain rule of backpropagation:

$$\frac{\partial \varepsilon}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\left(1 + \frac{\partial}{\partial x_l}\sum_{i=1}^{L-1} F(x_i, w_i)\right) \tag{3}$$

where $\frac{\partial \varepsilon}{\partial x_L}$ indicates that the gradient is calculated without going through the weight layer, and $\frac{\partial \varepsilon}{\partial x_L}\frac{\partial}{\partial x_l}\sum_{i=1}^{L-1} F(x_i, w_i)$ represents the gradient is calculated going through the weight layer. Additionally, $\frac{\partial}{\partial x_l}\sum_{i=l}^{L-1} F(x_i, w_i)$ cannot be equal to -1, which solves the problem of gradient vanishing.

ResNet is composed of multiple residual blocks that can be stacked together to create a deeper network. In addition, global average pooling layers and fully connected layers are incorporated for the final classification or regression tasks.

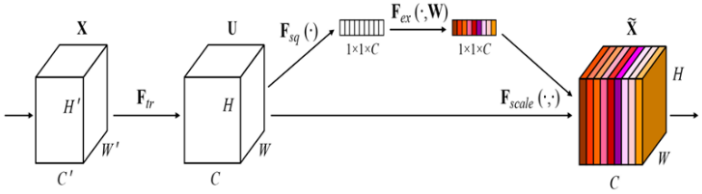


Figure 2: Structure diagram of SE block.

2.2. SE Block

SENet is a neural network architecture based on SE (Squeeze and Excitation) modules. The key component of this architecture is the squeeze-and-excitation module, which enhances the expressive power of the model by performing squeeze and excitation operations on the channels of the feature map. The structure of the module is illustrated in Figure 2.

The SE module consists of two steps: squeeze and excitation. In the squeeze step, the channels of the feature map are compressed by performing global average pooling, resulting in compressed features for each channel. In the excitation step, two fully connected layers are used to excite each channel, obtaining excitation weights for each channel. Finally, the excitation weights are multiplied with the original feature map, resulting in a feature map that has been enhanced by the SE module.

To enable the input feature map X to be processed by the SE module, it is first transformed or mapped to a feature map U using a transformation function F_{tr}. F_{tr} can be seen as a convolutional operator, and we can represent the convolutional kernel as$V = [v_1, v_2, \ldots, v_C]$, and the output as $U = [u_1, u_2, \ldots, u_C]$. The specific transformation process is as follows:

$$u_c = v_c * X = \sum_{s=1}^{C'} v_c^s \times x^s \tag{4}$$

where $*$ indicates convolution operation， v_c denotes the parameters of the c-th convolutional kernel, x^s represents the s-th input, $v_c = [v_c^1, v_c^2, \ldots, v_c^{C'}]$， $X = [x^1, x^2, \ldots, x^{C'}]$， $u_c \in \mathrm{R}^{H \times W}$， v_c^s

is a 2D spatial kernel that represents the individual channel of v_c， and operates on the corresponding channel of X.

To overcome the issue of utilizing channel dependencies, the processed U is further subjected to global average pooling to generate channel-level statistics. The statistic $z \in R^C$ is obtained by reducing U along its spatial dimensions H×W, The calculation formula for the c-th element of z is as follows:

$$z_c = F_{sq}(u_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{w} u_c(i,j) \tag{5}$$

where F_{sq} indicates the squeezing operation, which leverages global average pooling to compress each channel in the feature maps, resulting in compressed features for each channel.

In the squeeze step, the channel dimension of the input features is compressed by global average pooling to get a global feature vector. The compression is typically achieved through global average pooling, where the feature values of each channel are averaged.

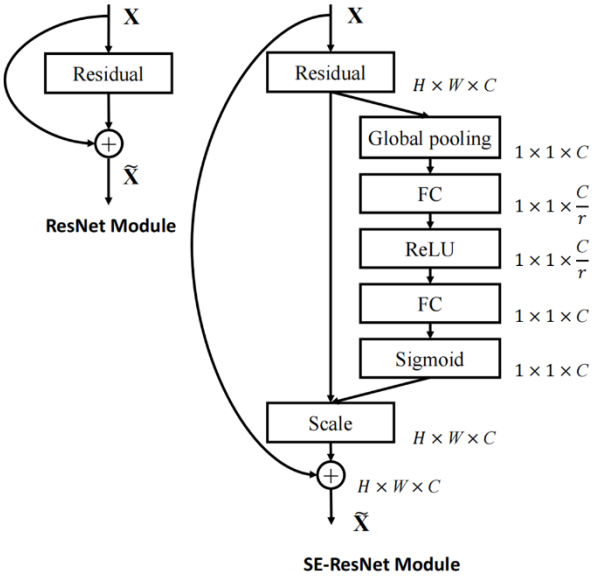


Figure 3: Structure comparison diagram of ResNet and SEResNet module.

To utilize the aggregated information from the squeezing operation and fully capture channel correlations, each channel is stimulated using two fully connected layers. A simple gating mechanism is employed, and the sigmoid function is used for activation. The computation formula is as follows:

$$s = F_{ex}(z, W) = \sigma\big(g(z, W)\big) = \sigma\big(W_2 \delta(W_1 z)\big) \tag{6}$$

Where δ denotes Relu activation function，σ indicates sigmoid activation function，and $W_1 \in R^{\frac{C}{r} \times C}$, $W_2 \in R^{C \times \frac{C}{r}}$.

After obtaining the output s, two fully connected layers are added around the non-linearity. The first layer is a dimension reduction layer with a reduction ratio of r, followed by a ReLU activation. Then, an expansion layer is applied to increase the channel dimension back to the transformed output U. This helps limit the complexity of the model and improve its generalization. The gating mechanism is parameterized as follows:

$$\widetilde{x_c} = F_{scale}(u_c, s_c) = s_c u_c \tag{7}$$

where $F_{scale}(u_c, s_c)$ represents the channel-wise multiplication between the scalar s_c and the

feature map $u_c \in R^{H \times W}$.

In the excitation process, the global feature vector undergoes a nonlinear transformation to generate an excitation vector. This excitation vector is then used as weights to multiply the original features, ultimately obtaining the enhanced features through the SE module.

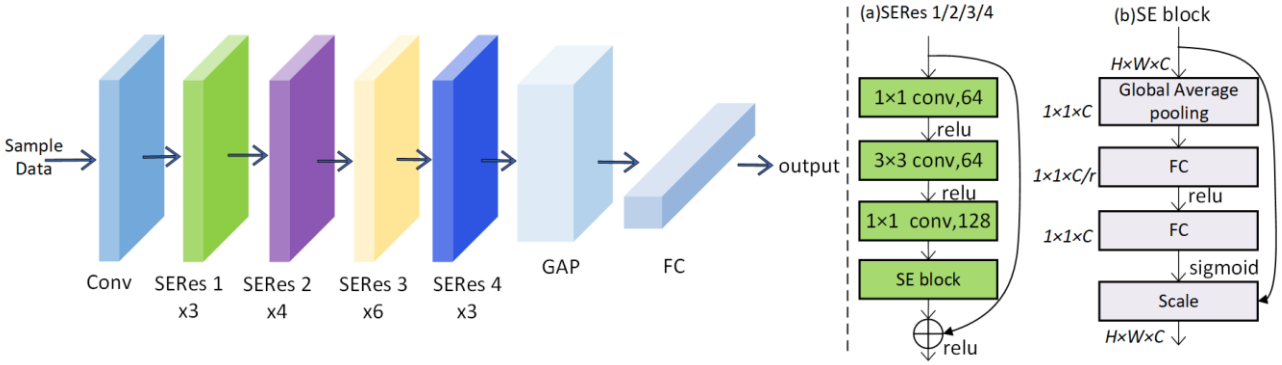


Figure 4: Structure diagram of SEResNet model.

2.3. SEResNet construction

Traditional ResNet network is constructed by stacking residual blocks, where each block contains a residual connection that allows for fast information propagation within the network. However, even with the presence of residual connections, there can still be challenges in information flow between different residual blocks. Especially in deep networks, due to the abstraction of features at multiple levels, the importance of features can become imbalanced, leading to the neglect of some important features and potentially affecting network performance.

To solve this issue, the SE module is embedded between residual connection and bottleneck in the resnet residual. The structure of the SE module is depicted in Figure 3. By incorporating the SE block, the network can adaptively learn the relationships between feature channels and weight the features based on their importance. This enables the network to enhance important features and improve performance.

In the SE block, the squeeze step performs spatial compression of features by applying global average pooling, resulting in a global feature descriptor. Then, two fully connected layers are utilized for non-linear transformations, mapping the global feature descriptor to a channel-wise weight vector. This enables the network to adaptively adjust the importance of each channel based on the channel-wise weight vector. The channel weights are multiplied with the input features through the scale and excitation steps, resulting in enhanced features after the SE block. This allows the network to better focus on important feature channels, enhancing the representation capability of the features and improving both the performance and generalization capability of the network.

In the SEResNet50, SE blocks are introduced to enhance the ResNet50 architecture. The structure of SEResNet50 is depicted in Figure 4. Among the figure, SERes means SE residual block, GAP represents global average pooling layer and the part on the right side of the dashed line in the figure represents the detailed structure of the left SERes. The SE residual blocks 2, 3, 4 have the same structure as SE residual block 1, but their dimensions are different and correspond to the dimensions in ResNet50, which are 128, 256, and 512, respectively. The SE block helps the network adaptively select and strengthen the task-relevant feature channels using channel-wise weights. This enhances the model's discriminative power and generalization ability.

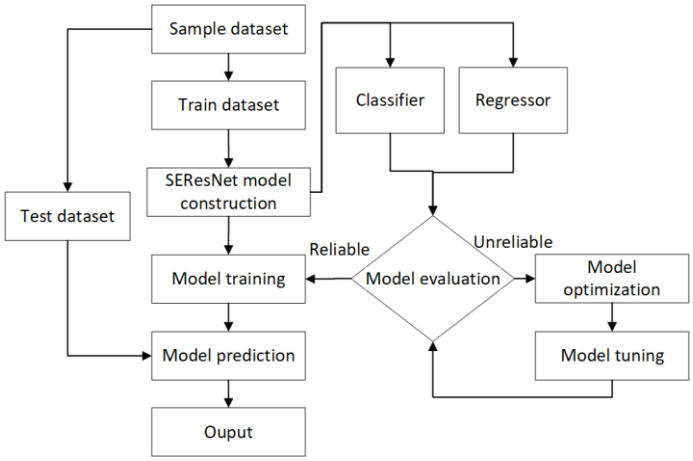


Figure 5: Structure diagram of SEResNet neural network inversion model.

3. Construction of damage cause inversion model

The damage factors inversion model is constructed as shown in Figure 5. Building the damage factors inversion model for the pier using Python language involves the utilization of various tools, including the numpy and sklearn libraries. The numpy library is a scientific computing library in Python that provides functionality for matrix operations. The sklearn library, on the other hand, is a third-party library for machine learning in Python that offers integrated tools for simple machine learning tasks.The steps involved in building the model are as follows:

(a) Collecting finite element model data of pile stress characteristics under different operating conditions.

(b) Performing data fusion and reconstruction on the stress sample data. Calculate the range, mean, variance, and coefficient of variation for each pile's stress features. Use these calculations as supplementary features for the sample data. Construct the sample dataset D, which includes the feature matrix X and the corresponding labels y. where $X \in R_{m\times n}$, $X \in R_{m\times q}$, m is the number of samples, n is the number of features, and q is the number of labels, including categories and sizes.

(c) The sample dataset was subjected to information fusion using normalization method, and the top 'a' principal components were extracted, and the resulting matrix $X_{pca} \in R_{m\times a}$.

(d) The dataset was randomly split using a holdout method into a training set S and a testing set T, with a ratio of 3:1.

(e) The SEResNet model was constructed with the optimization algorithm being Adam, and the activation functions used are sigmoid, logistic, tanh, and relu. For the classification learning classifier, the loss function chosen is cross-entropy loss[7], while for the regression learning classifier, the loss function used is mean squared error. The penalty factor α is set within the range of 0.0001 to 10. The maximum number of iterations and model convergence accuracy are set to train the model. K-fold cross-validation is performed by splitting the training set S and loading it into the SEResNet model for adaptive training. For the classification problem and regression problem, separate training is conducted. After training, cross-validation is performed to obtain model scores and evaluate model performance. The performance metrics mainly include accuracy (for classification) and mean absolute error (MAE, for regression). A low score indicates an unreliable model, prompting further optimization and parameter tuning. Methods such as adjusting the learning rate, adding or removing regularization terms, and optimizing hyperparameters are employed to optimize the inversion model.

(f) After training and validating the model on the training set S, the next step is to load the test set T into the model to further evaluate its generalization ability.

4. Construction of parameterized finite element models

4.1. Model construction

Building a parameterized finite element model using APDL[8] (ANSYS Parametric Design Language). In the analysis of damage factor inversion, the mooring force of the ship and the sliding thrust on the shore slope are both horizontal forces, while the load on the dock pile is vertical. To simplify the calculations, an elastic foundation beam approach can be chosen to model the soil constraints. The riprap foundation and the underlying soil are treated as elastic bodies, while the dock pile foundation is treated as a beam structure.

To facilitate the inversion calculation of damage factors, the three factors of ship mooring, shore slope sliding, and dock pile loading can be parameterized.

In the calculation of ship impact force, the range of ship tonnage is determined to be 2000t to 6000t. The water flow velocity is taken as 0.2m/s based on the average normalized flow velocity of the Three Gorges Reservoir. The impact time is set to 1s. Based on these values, the range of ship impact force can be calculated as 220 KN to 550 KN.

According to the "Code for Design of Building Foundation" (GB50007-2011), the maximum allowable displacement for a pile foundation is 40mm. The sliding action of the slope is considered equivalent to a uniformly distributed load applied to the pile foundation. Based on calculations, the maximum displacement of the upper beam of the pile foundation is 0.04m, and the combined load on a single pile is F = 1244kN. Therefore, the load range can be taken as 0-1244kN. Additionally, the range of increased loading intensity can be set from 0 to 1300kN.

To account for extreme conditions in the consideration of damage factors, the range for the dock load can be set slightly higher than the specified values in the code. Therefore, the range for the dock load can be set as 10 kPa to 60 kPa.

4.2. Model evaluation

Parametric model validation is shown in Figure 6. To validate the correctness of the established parameterized model, forward calculations can be performed by applying the same loads at corresponding positions and comparing the results. The calculations can be conducted according to three load application schemes.

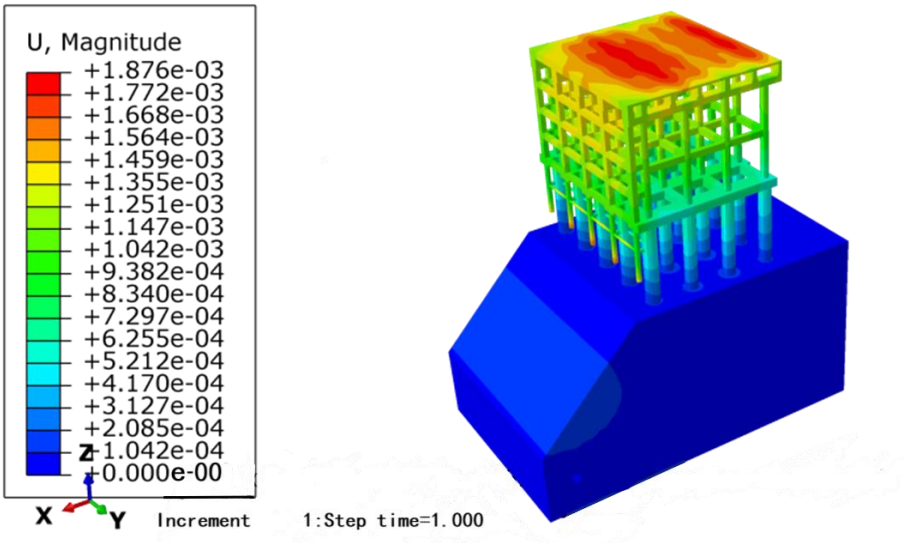


Figure 6: Finite element calculation model of solid element of high-piled wharf.

Table 1 Calculation result of vertical displacement of pile (mm)

Load / PF	Self-weight		Self-weight+ Heap Load		Self-weight+ ship collision	
	SIMP	ENT	SIMP	ENT	SIMP	ENT
No.1	3.1	2.7	3.5	3.4	3.1	2.8
No.2	2.8	2	3.4	3	2.8	2.2
No.3	2.7	2.4	3	2.9	2.7	2.3
No.4	2.4	2	2.5	2.3	2.5	2.1

According to the calculation results shown in Table 1, the average difference in vertical displacement values at the pile head is 14.5%. Among the table, PF represents pile foundation, SIMP denotes simplified model, and ENT indicates entity unit model. The displacement comparison curve, as shown in Figure 7, illustrates that the simplified model generally has larger vertical displacements compared to the solid element model. This is because the solid element model includes the presence of bedrock and fill layers at the bottom, which provide a confining effect on the pile foundation and reduce the vertical displacement at the pile head.

Table 2 Calculation results of vertical stress ratio of pile (MPa)

Load / PF	Self-weight		Self-weight + Heap Load		Self-weight + ship collision	
	SIMP	ENT	SIMP	ENT	SIMP	ENT
No.1	0.789	1.010	0.960	1.060	0.867	0.960
No.2	0.827	0.580	1.110	0.830	1.015	0.730
No.3	0.788	0.788	1.010	1.090	0.993	0.919
No.4	0.789	0.833	0.840	0.900	0.953	0.948

Table 2 presents the comparison of vertical stress values at the pile head, indicating an average difference of 5.3%. Figure 8 shows the comparison of vertical stress at the pile head, where the simplified model and the solid element model exhibit relatively small differences in numerical values overall.

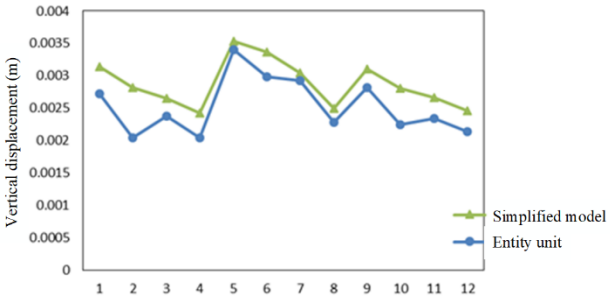


Figure 7: Vertical displacement curve of pile

Based on the above comparative analysis, the parameterized simplified finite element model shows little difference in computed results compared to the solid element model. Therefore, it can be used as the model for the inverse analysis. To proceed with the inverse analysis, the completed simplified finite element model is set as the invoked model. Quantified parameters related to adverse effects are randomly sampled and loaded into the solution file. The solution file, which is an ANSYS command stream file, contains commands for invoking the finite element model, defining all the loads, post-processing stress extraction commands, and exporting data using a macro file. The Python subprocess library is used to enter the ANSYS module and import the solution file, allowing it to automatically execute the finite element model, perform calculations, collect data, and export results [9-11].

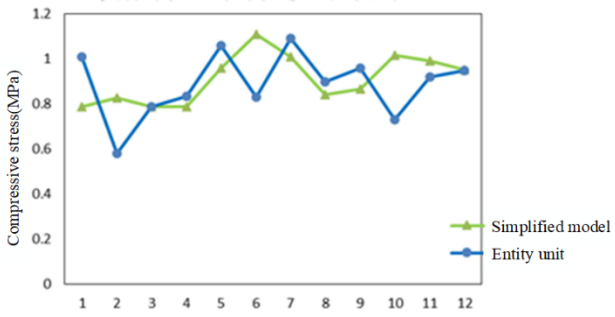


Figure 8: Comparison diagram of vertical stress on pile

Figure 9: Model for Damage Inversion of High Pile Wharf

5. Inversion of Damage Incentives

To test the adaptability of the SEResNet network model to complex stress features, we mainly use a sampling method to extract samples from indoor model tests to obtain single-factor sample conditions. These samples are then used in the simplified model for calculations to extract batches of stress feature data, as shown in Figure 9.

5.1. Model construction

During the parameter optimization of the inversion model, the model is optimized by setting the initial learning rate and adjusting the learning strategy, which includes strategies such as learning rate decay and adaptive learning rate adjustment. During the training process, the model is evaluated using the test dataset to calculate its performance metrics on the validation set. Based on the evaluation results, the model's hyperparameters, such as learning rate and regularization terms, can be adjusted to further optimize the model's performance. Cross-validation accuracy is used to validate the performance of the optimized network model. The model primarily addresses two recognition tasks:

(a) position recognition

Position recognition belongs to a classification problem, and the classification learner adopts cross-entropy loss as the model's loss metric. From Figure 10, it can be observed that the network model converges after 117 iterations.

To assess the generalization ability of the model for different positions of the loading factor, the training and testing sets are grouped and analyzed based on their positions. The trained model is then

used to evaluate the training and testing sets separately, resulting in specific metrics as shown in Table 3.

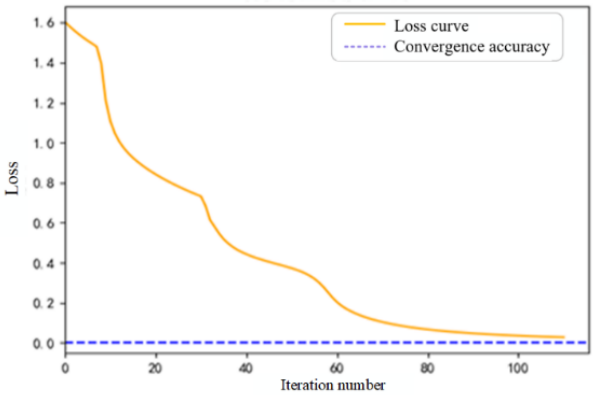


Figure 10: Loss curve of classification learner under surcharge load

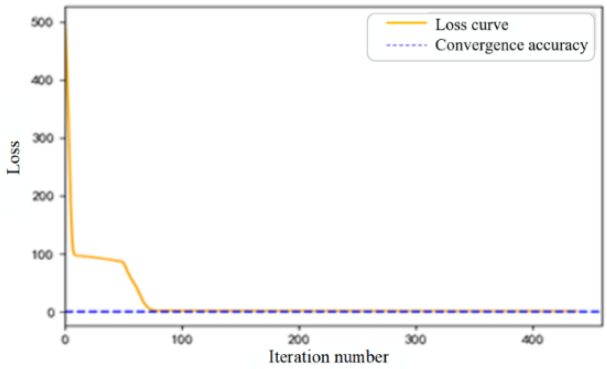


Figure 11: Loss curve of regression learner under surcharge load

From Table 3, it can be observed that the position recognition accuracy for the four regions in the training set is 0.989, and the recognition accuracy for the testing samples is 0.978. The F1 score and Hamming loss are also quite satisfactory. Additionally, the metrics and scores for the training and testing sets are relatively close, indicating that the neural network model has been well tuned and exhibits strong generalization ability. It is capable of accurately identifying the positions of the loading factor.

Table 3: Location recognition result

Position	Train numbers	Train accuracy	F1	Hamming loss	Test numbers	Test accuracy	F1	Hamming loss
1	1921	0.998	0.998	0.001	646	0.988	0.985	0.002
2	1849	0.996	0.989	0.001	628	0.978	0.977	0.002
3	1800	0.989	0.989	0.002	624	0.988	0.990	0.002
4	1930	0.991	0.990	0.001	602	0.991	0.990	0.001
Summary	7500	0.994	0.992	0.001	2500	0.986	0.986	0.002

(b)Size Recognition

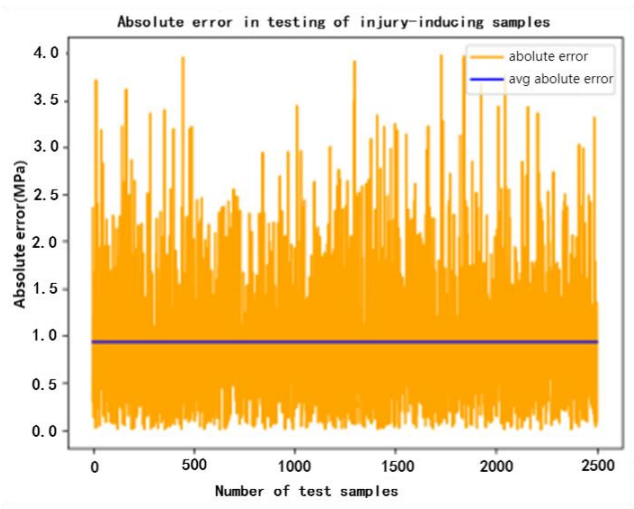
Size recognition belongs to a regression problem. The regression learner adopts variance as the model's loss metric. From Figure 11, it can be observed that when the neural network model iterates for 75 times, the loss error reaches around 0.001, indicating that the model has converged.

The training and testing sets were analyzed, and the size recognition results of the model were

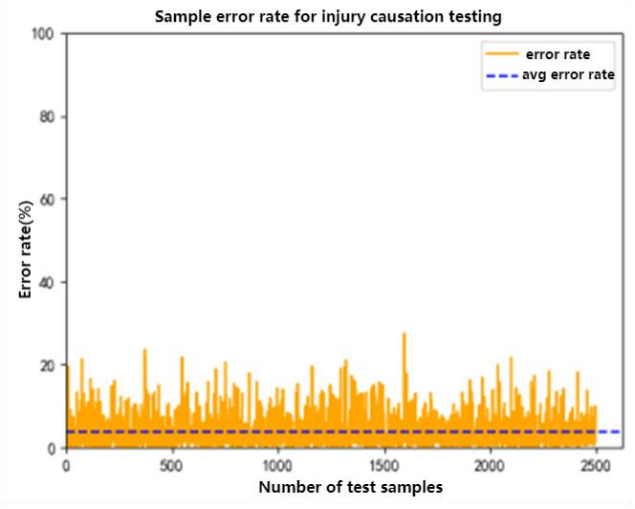
obtained, as shown in Table 4.

Table 4: Size recognition result

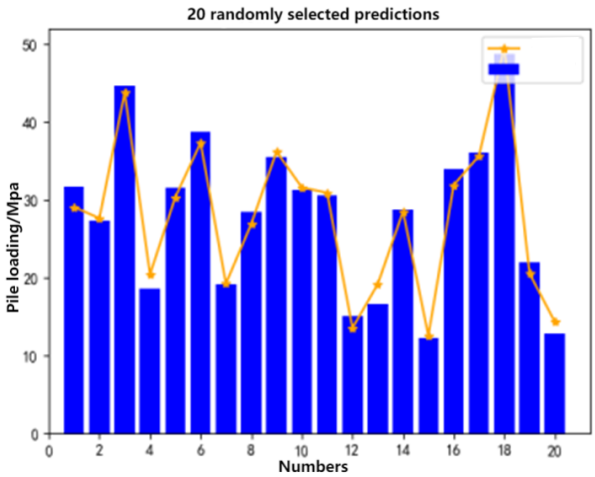
Position	Train numbers	R2	MSE	MAE	Test numbers	R2	MSE	MAE
1	1921	0.990	1.320	0.920	646	0.990	1.270	0.888
2	1849	0.989	1.330	0.911	628	0.988	1.420	0.974
3	1800	0.991	1.090	0.839	624	0.991	1.189	0.857
4	1930	0.989	1.310	0.909	602	0.991	1.143	0.850
Summary	7500	0.990	1.263	0.895	2500	0.990	1.256	0.892



a) Absolute error of test sample



b) Relative error of test sample



c) Comparison of predicted values for 20 randomly sampled samples.

Figure 12: Schematic diagram of mean absolute error

From Figure 12, it can be observed that the absolute errors are all within 4.0 kPa, with an average absolute error of approximately 0.892 kPa. The average relative error is 2.97%. These results indicate that the model performs well in identifying the magnitude of the pile load. For the 10,000 samples of pile load effects, the metrics and scores of the training and testing sets show minimal differences. This indicates that the constructed neural network inversion model has good predictive capabilities and does not suffer from underfitting or overfitting issues. It demonstrates strong generalization ability and can accurately and efficiently predict the position and magnitude of pile load effects, meeting the requirements for identifying damage factors under typical conditions.

5.2. Inversion analysis of ship berthing and shore slope thrust effects

Due to space limitations, the detailed process of damage factor inversion for ship berthing and shore slope thrust, as described in section 5.1, will not be further elaborated. The analysis shows that the methods employed in this study yield good inversion results for both damage factors.

5.3. Inversion analysis of mixed damage factor effects.

Under the same type of adverse effects, the stress characteristics of the wharf exhibit similarity. Inputting samples with similar characteristics into the model will inevitably lead to an improvement in the model's recognition accuracy. The SEResNet network inversion model is highly sensitive to the data features of the samples. By integrating the datasets of the three types of damage factors, it serves as the input for the inversion model. The adaptability and recognition accuracy of the inversion model for the type, location, and magnitude of damage factors can be observed.

Due to space limitations, this paper directly presents the analysis results as shown in Tables 5 to 7.

From Table 5, it represents that the model can accurately determine the type of all samples belonging to the different damage causation effects. This indicates that there are clear boundaries in the distribution of pile foundation stress under different damage causation types, and the model can accurately identify them based on sample features.

From Table 6, it represents the position recognition accuracy under the pile load effect reaches 0.98, while the position recognition accuracy under the ship collision effect reaches 0.9. Compared to the accuracy of individual ship collision samples, there is a slight decrease, but it still exhibits good recognition performance. As for the shore slope effect, there is only one position, so there is no need for recognition.

Table 5: Type recognition result

Damage factor	Train numbers	Train accuracy	F1	Hamming loss	Test numbers	Test accuracy	F1	Hamming loss
Pile loading	7500	1.0	1.0	0.0	2500	0.999	1.0	0.0
Ship berthing	7500	1.0	1.0	0.0	2500	1.0	1.0	0.0
Shore slope thrust	750	1.0	1.0	0.0	250	1.0	1.0	0.0

Table 6: Location recognition result

Damage factor	Train numbers	Train accuracy	F1	Hamming loss	Test numbers	Test accuracy	F1	Hamming loss
Pile loading	7500	0.998	0.999	0.0001	2500	0.985	0.98	0.021
Ship berthing	7500	0.908	0.888	0.0912	2500	0.902	0.9	0.09
Shore slope thrust	750	-	-	-	250	-	-	-
Summary	15750	0.954	0.95	0.0551	5250	0.943	0.94	0.056

Table 7: Size recognition result

Damage factor	Train numbers	R2	MSE	MAE	Test numbers	R2	MSE	MAE
Pile loading	7500	0.977	2.998	1.369	2500	0.965	3.223	1.587
Ship berthing	7500	0.997	38.406	4.93	2500	0.997	37.65	4.9.5
Shore slope thrust	750	0.997	35.624	4.786	250	0.995	34.951	4.955
Summary	15750	0.99	21.735	3.24	5250	0.985	25.274	2.18

From Table 7, it denotes the position recognition accuracy under the pile load effect reaches 0.965, which is slightly lower compared to the accuracy of individual ship collision samples. The position

recognition accuracy under the ship collision effect reaches 0.99.

Overall, the inversion model performs well on the mixed samples. Compared to samples with a single type of effect, the position recognition accuracy for pile load remains largely unchanged, while the size recognition accuracy slightly decreases. For ship collision effect, the position recognition accuracy decreases by 5%, while the size recognition accuracy remains unchanged. As for the effect of bank slope, the accuracy remains the same.

6. Conclusion

Based on the SEResNet model, a damage factor inversion model was constructed, and the MANSYS module was invoked using the subprocess module in Python for analysis, demonstrating the feasibility of batch calculations. By comparing the stress characteristics with the solid element model, it was found that there is little difference between the two models, indicating that the simplified model data can be used as a substitute for calculations.

The inversion model was optimized by adjusting the learning rate, adding or removing regularization terms, and tuning hyperparameters. Cross-validation was performed to obtain the model evaluation scores, and the damage factors of pile loading, ship collision during berthing, and shore slope thrust were analyzed separately. The analysis showed that the position recognition accuracy for pile loading was 0.98, with an average absolute error of 0.892 MPa for size recognition. For ship collision, the position recognition accuracy was 0.94, with an average absolute error of 5.5 kN for size recognition. The average absolute error for position and size recognition of pile loading was 1.2 kN. The inversion model demonstrated excellent generalization ability for individual damage factor samples.

By integrating all the samples and conducting a mixed damage factor analysis, the performance of the SEResNet inversion model was evaluated. The analysis revealed that for the mixed sample set, the accuracy of type recognition was 1.0, the accuracy of position recognition was 0.94, and the average absolute error in size recognition was 2.2. Although there was a decrease in accuracy and scores in certain damage factor recognition indicators, the overall prediction accuracy of the model still met the general requirements for inversion recognition.

Acknowledgements

This paper is one of the results of the scientific and technological research project of Chongqing Municipal Education Commission, "Research on Multi-domain Information Fusion Model for Small Aircraft Detection Based on Deep Learning" (KJQN202101507), and the project of Master's Graduate Student Innovation Plan of Chongqing University of Science and Technology (ZNYKJCX2022023).

References

[1] Fang Jing, Zhang Yanchi, Zhu Yaxian, Pan Deqiang, Wang Shengnian. Damage situation and durability countermeasures and suggestions of high-piled wharf in seaport [C]. Proceedings of the Science and Technology Forum on Durability and Design Methods of Concrete Structures in Coastal Areas and the 6th National Symposium on Concrete Durability. 2004: 387-400.

[2] Zuo Liangdong, et al. "Inversion of damage inducement of wharf pile foundation under heaped load based on parametric model." SN Applied Sciences 4. 3 (2022): 69.

[3] Wang Yuanzhan, Li Da. Analysis of overall safety degree of high-piled wharf structure under stacking load [J]. Waterway Port, 2013, 34(5): 430-435.

[4] Zhang Junhong, Li Shihai, Xu Likai, Hu Han. Experimental test and inversion study on mechanical properties of large pipe piles in harbor wharf [J]. Port Engineering Technology, 2006, 4: 21-23.

[5] Jie Hu, Li Shen, Gang Sun; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 7132-7141

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778
[7] Zhang Zhilu, and Mert Sabuncu. "Generalized cross entropy loss for training deep neural networks with noisy labels. " Advances in neural information processing systems 31 (2018).
[8] Sun Xiaochao, Zhang Yinghui, Lu Qi, Cui Di, He Weidong. Simulation analysis of small-period dynamic meshing of cycloidal pinwheel drive based on APDL [J]. Journal of Dalian Jiaotong University, 2020, 18 (05): 12-16.
[9] A. Hakan AKTAS. Development a Spectrophoto- metric of Fe(III), Al(III)and Cu(II) Using Eriochrome Cyanine R Ligand and Assessment of the Obtained Data by Partial Least-Squares and Artificial Neural Network Method—Application to Natural Waters[J]. Spectroscopy and Spectral Analysis，2018, 18(08): 102-106.
[10] Hao Cheng, Dorian J. Garrick, Rohan L. Fernando. Efficient strategies for leave-one-out cross validation for genomic best linear unbiased prediction [J]. Journal of Animal Science and Biotechnology, 2017, 18(03): 352-356.
[11] Tianxiang Zhang, Jinya Su, Cunjia Liu, Wenhua Chen. Potential Bands of Sentinel-2A Satellite for Classification Problems in Precision Agriculture [J]. International Journal of utomation and Computing, 2019, 18(01): 182-188.