applied sciences

MDPI

Article

Orthogonal Matrix-Autoencoder-Based Encoding Method for Unordered Multi-Categorical Variables with Application to Neural Network Target Prediction Problems

Yiying Wang [1], Jinghua Li [2,3,*], Boxin Yang [2,*], Dening Song [2] and Lei Zhou [2]

[1] College of Shipbuilding Engineering, Harbin Engineering University, Harbin 150001, China; wyy0316@hrbeu.edu.cn

[2] College of Mechanical and Electrical Engineering, Harbin Engineering University, Harbin 150001, China; songdening@hrbeu.edu.cn (D.S.); zhoulei@hrbeu.edu.cn (L.Z.)

[3] Sanya Nanhai Innovation and Development Base of Harbin Engineering University, Harbin Engineering University, Sanya 572024, China

* Correspondence: lijinghua2024@126.com (J.L.); yangboxin@hrbeu.edu.cn (B.Y.)

Abstract: Neural network models, such as BP, LSTM, etc., support only numerical inputs, so data preprocessing needs to be carried out on the categorical variables to convert them into numerical data. For unordered multi-categorical variables, existing encoding methods may produce dimensional catastrophes and may also introduce additional order misrepresentation and distance bias in neural network computation. To solve the above problems, this paper proposes an unordered multi-categorical variable encoding method O-AE using orthogonal matrix for encoding and encoding representation learning and dimensionality reduction via an autoencoder. Bayesian optimization is used for hyperparameter optimization of the autoencoder. Finally, seven experiments were designed with the basic O-AE, Bayesian optimization of the hyperparameters of the autoencoder for O-AE, and other encoding methods to encode unordered multi-categorical variables in five datasets, and they were input into a BP neural network to carry out target prediction experiments. The results show that the experiments using O-AE and O-AE-b have better prediction results, proving that the method proposed in this paper is highly feasible and applicable and can be an optional method for the data processing of unordered multi-categorical variables.

Keywords: unordered multi-categorical variables; orthogonal matrix; autoencoder; encoding method; dimensionality reduction; Bayesian optimization; BP neural network; target prediction

check for updates

Citation: Wang, Y.; Li, J.; Yang, B.; Song, D.; Zhou, L. Orthogonal Matrix-Autoencoder-Based Encoding Method for Unordered Multi-Categorical Variables with Application to Neural Network Target Prediction Problems. *Appl. Sci.* **2024**, *14*, 7466. https://doi.org/10.3390/app14177466

Academic Editor: Douglas O'Shaughnessy

Received: 6 August 2024
Revised: 21 August 2024
Accepted: 21 August 2024
Published: 23 August 2024

1. Introduction

Target prediction is the prediction of a target associated with an input based on existing data by some method. Target prediction is increasingly important in finance [1,2], healthcare [3], manufacturing [4], power [5,6], weather [7], transportation [8], etc. Excellent prediction results can identify potential risks, provide credible prediction data, and improve users' risk management ability; it can develop executable future plans and provide decision-making support for enterprises.

Neural networks, decision trees, linear regression, polynomial regression, etc., are methods to achieve target prediction. Kim [9] used multiple target prediction methods, conducted experiments under different numbers of features, feature types, and numbers of samples and concluded that the performance of artificial neural networks with target prediction is better under multi-categorical variable input conditions.

Overall, neural networks are the dominant model for solving target prediction problems at this stage. It is important to note that neural network models, such as BP, LSTM, etc., support only numerical data inputs; however, a large portion of the dataset contains non-numerical variables, i.e., categorical variables. For example, the binary categorical variable "drug reaction" includes two categories of "positive" and "negative", and the "assessment result" is also a binary categorical variable, including "pass" and "fail"; another example is the multi-categorical variable "occupation", which includes "teacher, firefighter, construction worker, doctor..." and other categories. Numerical variables are easier to interpret, but when categorical variables dominate the dataset, it is not easy to view data trends and make predictions [10]. Therefore, a method must be taken to map categorical variables to numerical values before inputting them into the neural network [11].

Categorical variable coding methods include one-hot encoding, label encoding, target encoding, embedding, etc. In this, one-hot encoding is applicable to unordered categorical variables, which generates N-dimensional binary vectors using 0, 1 for N categories. Label encoding applies to ordered categorical variables, and it maps each category to an integer starting from 0. Target encoding uses the target mean corresponding to the category variables instead of the categorical variables. Embedding, on the other hand, randomly generates weight vectors according to the specified embedding dimensions and maps the text into word vectors according to the indexes of the categorical variables.

Unordered multi-categorical variables are categorical variables that contain several (usually more than three) categories with no differences in order or distance between categories.

Some existing encoding methods are not entirely suitable for unordered multi-categorical variables; for example, when the number of categories is large, applying one-hot encoding or applying embedding with a large embedding dimension creates the risk of dimensionality explosion, and the data are too sparse after one-hot encoding. Embedding and label encoding also introduce additional order and distance to an originally unordered categorical variable. Target encoding, on the other hand, affects the ability of the neural network model to extract information with the risk of overfitting.

In the current research on the target prediction problem, scholars are more concerned about how to design a better model structure to make better prediction results, and when carrying out the input data processing, the data either do not include the categorical variables [12–14] or use other traditional coding methods. For example, Bu et al. [15] used label encoding for both ordered and unordered categorical variables for data preprocessing when adopting selective integrated learning for ship painting man-hour prediction. Hur et al. [16] predicted ship construction man-hours using deployable data at different times during the manufacturing process and converted the categorical variables into dummy variables when processing them. Sasan et al. [17] converted binary categorical variables into 1 and 2 labeled coding. Wang et al. [18] set working time and non-working time as 0 and 1 coding, respectively. Carrizosa et al. [19] processed all the categorical variables using one-hot encoding in a binary categorical problem in the presence of categorical variables. All of the above studies adopted the existing conventional encoding methods to carry out the codes for unordered multi-categorical variables but ignored the fact that the encoded values did not maintain the characteristics of unordered multi-categorical variables and also additionally introduced the misleading order and distance bias.

In his research on encoding methods for categorical variables, Sebastian [20] applied five encoding methods to three regression models in his experiments and proved that the regression results varied according to the encoding methods of categorical variables; Hien et al. [21] also verified that the three encoding methods of categorical variables, namely label encoding, one-hot encoding, and embedding, had different effects on the performance of a deep dense neural network and long- and short-term memory neural network models. Li et al. [22] proposed an ordered log-linear model for ordered categorical variables that can convert ordered categorical data into model-describable multi-categorical lists; Meulemeester et al. [23] used unsupervised embedding to convert categorical variables into word vectors. Dahouda et al. [24] extended the word-embedding approach by proposing a deep learning method to codify categorical variables. All of the above methods are more suitable for ordered categorical variables or for categorical variables with dependencies. Namgil et al. [25] devised a Bayesian network based method for converting categorical variables into data variables, and Jung et al. [26] proposed a method for updating data points in the kernel space of a continuous variable by using SVMs to reflect the effect of each categorical variable; all of the above methods are only suitable for the target value of a binary categorical variable.

In general, there is no encoding method for transforming unordered multi-categorical variables into numerical codes with the characteristics of unordered variables and small dimensions. Based on this need, this paper carries out a research on encoding methods for unordered multi-categorical variables. This paper also extends the problem of predicting the working hours of cruise ship production design tasks by analyzing the relationship between task workload and task working hours through the previous working hours data and the design attribute data in the ship area, such as the task type, the number of model structures, the planned working hours, the feedback working hours, etc., so as to provide credible and standards-based working hours solutions for the planning and scheduling of the design tasks when designing and constructing a new ship in the future, which is also the key link to promote the standardization of ship design and construction. In the ship production design task working hour prediction problem, ship area and task type are categorical variables containing dozens of unordered categories, so the unordered multi-categorical variables cannot be ignored.

Autoencoder, as a classical tool for feature extraction and data dimensionality reduction, has continued to show its unique value in the field of data processing in recent years. For example, the research of Yang et al. [27] is a profound exploration, in which they cleverly integrate a deep autoencoder network into the Orthogonal Nonnegative Matrix Factorization (ONMF) framework, which achieves an accurate capture and hierarchical parsing of the intrinsic structure of complex data. This innovation not only highlights the ability of the autoencoder to automatically extract high-level features from data in an unsupervised learning environment but also effectively reduces the data dimensionality while preserving key information through its unique network structure.

Based on this core feature of autoencoder, this paper proposes a new encoding method, O-AE, for unordered multi-categorical variables, which first uses an orthogonal matrix to numerically encode the unordered multi-categorical variables, ensuring that the codes are independent of each other in terms of position, are equal distance, and are the same size; second, it uses autoencoder to carry out representation learning on the numerical codes and then performs the dimensionality reduction to ensure that the data inputted to the neural network has learned the relevant characteristics of the orthogonal matrix at the same time low dimensionality. This paper is structured as follows. Section 2 introduces four traditional methods for encoding multi-categorical variables; Section 3 describes the encoding mechanism of O-AE proposed in this paper; Section 4 introduces the Bayesian optimization of the hyperparameters of autoencoder for O-AE; Section 5 designs seven encoding experiments for six encoding methods (including O-AE), designs the Bayesian optimization of the hyperparameters of the autoencoder for O-AE, designs seven encoding experiments, introduces the specific information of five datasets, and proposes the validation metrics; and Section 6 carries out the example validation and demonstrates and discusses the experimental results.

2. Encoding of Categorical Variables

This section presents the theory of several types of encoding methods that are used frequently. First, the parametric representation of unordered multi-categorical variables is presented along with the parametric representation after encoding.

Definition 1. *Unordered multi-categorical variables are non-numerical variables with no size or positional differences between categories.*

Definition 2. *The sample size of unordered multi-categorical variables* CV *is set as* $\tilde{N}$*, and the variable contains* N *categories* $c_i, i = 1, 2, \cdots N, N \geq 3, N \leq \tilde{N}$*. There are multiple data points of the same category in* CV*, among which there are* m *data points in category* c_i*,* $c_{i1}, c_{i2}, \cdots c_{im}, m \geq 1$*.*

$$CV = \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_{i1} \\ c_{i2} \\ \cdots \\ c_N \end{bmatrix}_{\tilde{N} \times 1} \rightarrow CV : \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_i \\ \cdots \\ c_N \end{bmatrix}_{N \times 1}, c_i = (c_{i1}, c_{i2}, \cdots c_{im}), m \geq 1 \tag{1}$$

Using the encoding method f, the category c_i is mapped to the numerical value n_i such that the unordered multi-categorical variables CV are mapped to the numerical variables NV.

$$CV : \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_i \\ \cdots \\ c_N \end{bmatrix}_{N \times 1} \xrightarrow{f} NV : \begin{bmatrix} n_1 \\ n_2 \\ \cdots \\ n_i \\ \cdots \\ n_N \end{bmatrix}_{N \times 1} \tag{2}$$

The categorical data of the same category have the same numerical data under the same encoding method mapping.

The m identical categories c_i in CV, under f mapping, $c_{i1}, c_{i2}, \cdots c_{im}$, are transformed into numerical values $n_{i1}, n_{i2}, \cdots n_{im}$, respectively, and $n_{i1} = n_{i2} = \cdots = n_{im}$.

$$\begin{cases} CV : [c_i] \xrightarrow{f} NV : [n_i] \\ c_i = (c_{i1}, c_{i2}, \cdots c_{im}), n_i = (n_{i1}, n_{i2}, \cdots n_{im}), m \geq 1, n_{i1} = n_{i2} = \cdots = n_{im} \end{cases} \tag{3}$$

2.1. Label Encoding

The label encoding f_L maps the category c_i to integers from 0 to $N - 1$.

$$CV : \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_i \\ \cdots \\ c_N \end{bmatrix}_{N \times 1} \xrightarrow{f_L} NV : \begin{bmatrix} 0 \\ 1 \\ \cdots \\ i-1 \\ \cdots \\ N-1 \end{bmatrix}_{N \times 1} \tag{4}$$

2.2. Target Encoding

The target encoding f_T maps the category c_i to the mean of the target value corresponding to that category.

Assuming that the predicted target is y, the target value corresponding to category c_i is y_i. When c_i contains multiple similar data $c_{i1}, c_{i2}, \cdots c_{im}$ within c_i, the corresponding target values are $y_{i1}, y_{i2}, \cdots y_{im}$.

$$CV : \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_i \\ \cdots \\ c_N \end{bmatrix}_{N \times 1} \xrightarrow{f_T} NV : \begin{bmatrix} n_1 \\ n_2 \\ \cdots \\ n_i \\ \cdots \\ n_N \end{bmatrix}_{N \times 1}, n_i = \sum_{j=1}^{m} y_{ij} / m \tag{5}$$

2.3. One-Hot Encoding

One-hot encoding f_O maps the N categories c_i into N binary row vectors with N-dimensional size, where each binary vector has only one valid digit 1 and the rest of the positions are 0, and the positions of the valid digits are different between different vectors.

$$CV:\begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_i \\ \cdots \\ c_N \end{bmatrix}_{N\times 1} \overset{f_O}{\rightarrow} NV:\begin{bmatrix} \boldsymbol{n_1} \\ \boldsymbol{n_2} \\ \cdots \\ \boldsymbol{n_i} \\ \cdots \\ \boldsymbol{n_N} \end{bmatrix}_{N\times N} = \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 \\ & & & \cdots & & \\ 0 & 0 & \cdots & 1 & \cdots & 0 \\ & & & \cdots & & \\ 0 & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}_{N\times N} \tag{6}$$

For example, the categorical variable "Occupation" contains three categories "Teacher, Firefighter, Construction Worker", which are mapped to binary vectors [1,0,0], [0,1,0], [0,0,1] using one-hot encoding f_O.

From this encoding method, it is easy to see that each vector is a unit vector and the vectors are orthogonal to each other, i.e., the vectors are equal in distance and independent of each other, which is fully consistent with the characteristics of unordered categorical variables.

However, one-hot encoding also has some problems: the vectors are only composed of 0 and 1, and when the categorical variables are of large categories, the encoding obtained from the mapping will be sparse and of huge dimensions, which will make the computational cost large.

2.4. Embedding

The embedding encoding f_E maps each category to a point in a vector space, and vectors with similar categories are closer together in the vector space.

Setting embedding converts the categorical variable CV into a numerical variable NV with a specified embedding dimension d_E. The embedding dimension d_E denotes the dimension of the mapped n_i, i.e., $c_i \overset{f_E}{\rightarrow} \boldsymbol{n_i} = \begin{bmatrix} n_i^1 & n_i^2 & \cdots & n_i^{d_E} \end{bmatrix}$.

Embedding is roughly divided into four steps.

(1) Predefine the vector space, determine the number N of categories c_i contained within the categorical variables, specify the embedding dimension d_E, and construct a weight matrix $\boldsymbol{W}$ of N rows and d_E columns. The weight matrix is randomly generated by default from a standard normal distribution with mean 0 and standard deviation 1. That is, $\boldsymbol{W} \sim N(0,1)$.

(2) Assign an integer index starting from 0 to each category in the categorical variables.

(3) Map the integer indexes into a predefined vector space by means of a lookup table (this table is the weight matrix $\boldsymbol{W}$), where each index is associated with a vector corresponding to the number of the weight matrix.

(4) Return the mapped numeric variables for the categorical variables.

$$CV:\begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_i \\ \cdots \\ c_N \end{bmatrix}_{N\times 1} \overset{f_E}{\rightarrow} NV:\begin{bmatrix} \boldsymbol{n_1} \\ \boldsymbol{n_2} \\ \cdots \\ \boldsymbol{n_i} \\ \cdots \\ \boldsymbol{n_N} \end{bmatrix}_{N\times d_E} \rightarrow \boldsymbol{W} = \begin{bmatrix} n_1^1 & n_1^2 & \cdots & n_1^{d_E} \\ n_2^1 & n_2^2 & \cdots & n_2^{d_E} \\ & & \cdots & \\ n_i^1 & n_i^2 & \cdots & n_i^{d_E} \\ & & \cdots & \\ n_N^1 & n_N^2 & \cdots & n_N^{d_E} \end{bmatrix} \tag{7}$$

3. Orthogonal Matrix-Autoencoder-Based Encoding Method for Unordered Multi-Categorical Variables

Among the existing methods for encoding categorical variables, part of the methods is not applicable to unordered multi-categorical variables with no size, order, or distance requirements between categories, and the other part of the methods have large dimensionality after encoding. Therefore, this paper proposes the method O-AE based on orthogonal matrix-autoencoder for encoding and dimension reduction of unordered multi-categorical variables. O-AE for unordered multi-categorical variables encoding and dimensionality reduction process is shown in Figure 1.

$$CV : [c_i] \overset{f_{O-AE}}{\rightarrow} \boldsymbol{Z} \in \mathbb{Z} \Leftrightarrow \langle CV \rightarrow \boldsymbol{A} \overset{QR}{\rightarrow} \boldsymbol{Q} \rightarrow \boldsymbol{X} \in \mathbb{X}, \boldsymbol{X} \in \mathbb{X} \overset{encoder}{\rightarrow} \boldsymbol{Z} \in \mathbb{Z} \overset{decoder}{\rightarrow} \hat{\boldsymbol{X}} \in \mathbb{X} \rangle \tag{8}$$

The model is mainly divided into two parts, the original code generation part and the code reduction part.

The original code generation part is $CV \rightarrow \boldsymbol{A} \overset{QR}{\rightarrow} \boldsymbol{Q} \rightarrow \boldsymbol{X} \in \mathbb{X}$. The code dimensionality reduction part is $\boldsymbol{X} \in \mathbb{X} \overset{encoder}{\rightarrow} \boldsymbol{Z} \in \mathbb{Z} \overset{decoder}{\rightarrow} \hat{\boldsymbol{X}} \in \mathbb{X}$.

First, according to the number of categories N in the unordered multi-categorical variables, the N-order orthogonal matrix $\boldsymbol{Q}$ is constructed, and the orthogonal matrix is mapped one by one with the original unordered multi-categorical variables according to the positional indexes into an orthogonal numerical coding matrix $\boldsymbol{X}$ with the same data size, independent positions, and equal distances.

Second, the orthogonal matrix of numerical code matrix $\boldsymbol{X}$ is input to the autoencoder, and the autoencoder is trained step by step by compressed reconstruction coding through the encoder and decoder in the autoencoder; finally, the learned feature $\boldsymbol{Z}$ is obtained.

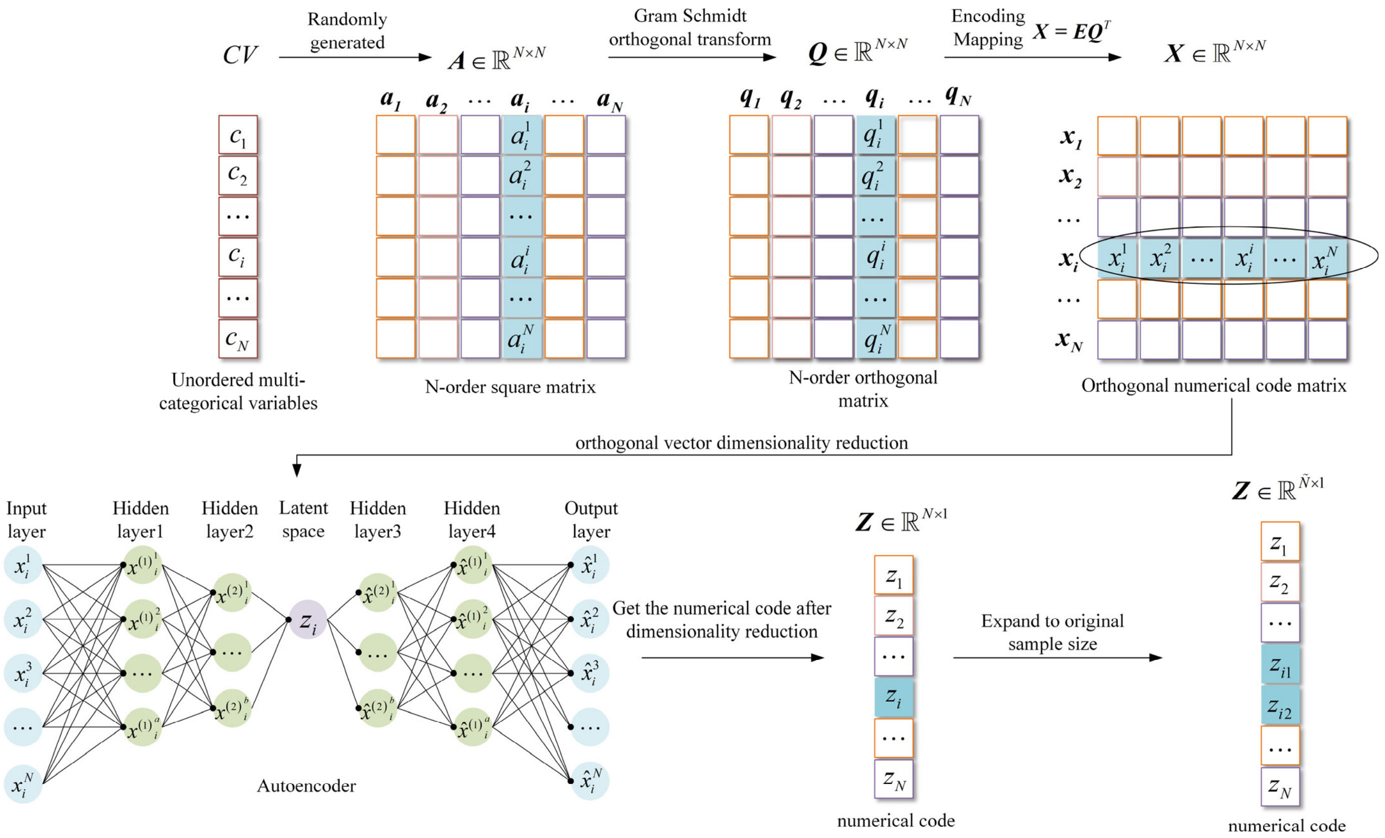

CV
Randomly generated
$A \in \mathbb{R}^{N\times N}$
Gram Schmidt orthogonal transform
$Q \in \mathbb{R}^{N\times N}$
Encoding Mapping
$X = EQ^T$
$X \in \mathbb{R}^{N\times N}$
a_1 a_2 ⋯ a_i ⋯ a_N
a_i^1 a_i^2 ⋯ a_i^i ⋯ a_i^N
q_1 q_2 ⋯ q_i ⋯ q_N
q_i^1 q_i^2 ⋯ q_i^i ⋯ q_i^N
x_1 x_2 ⋯ x_i ⋯ x_N
x_i^1 x_i^2 ⋯ x_i^i ⋯ x_i^N
c_1 c_2 ⋯ c_i ⋯ c_N
Unordered multi-categorical variables
N-order square matrix
N-order orthogonal matrix
Orthogonal numerical code matrix
orthogonal vector dimensionality reduction
Input layer
Hidden layer1
Hidden layer2
Latent space
Hidden layer3
Hidden layer4
Output layer
x_i^1 x_i^2 x_i^3 ⋯ x_i^N
$x^{(1)}{}_i^1$ $x^{(1)}{}_i^2$ ⋯ $x^{(1)}{}_i^a$
$x^{(2)}{}_i^1$ ⋯ $x^{(2)}{}_i^b$
z_i
$\hat{x}^{(2)}{}_i^1$ ⋯ $\hat{x}^{(2)}{}_i^b$
$\hat{x}^{(1)}{}_i^1$ $\hat{x}^{(1)}{}_i^2$ ⋯ $\hat{x}^{(1)}{}_i^a$
$\hat{x}_i^1$ $\hat{x}_i^2$ $\hat{x}_i^3$ ⋯ $\hat{x}_i^N$
Autoencoder
Get the numerical code after dimensionality reduction
$Z \in \mathbb{R}^{N\times 1}$
z_1 z_2 ⋯ z_i ⋯ z_N
numerical code
Expand to original sample size
$Z \in \mathbb{R}^{\tilde{N}\times 1}$
z_1 z_2 ⋯ z_{i1} z_{i2} ⋯ z_N
numerical code

Figure 1. O-AE for unordered multi-categorical variable encoding and dimensionality reduction process.

3.1. Code Generation

The N-order square matrix is set as $\boldsymbol{A} \in \mathbb{R}^{N\times N}$, $\boldsymbol{A} = [\boldsymbol{a_1}, \boldsymbol{a_2}, \cdots, \boldsymbol{a_i}, \cdots, \boldsymbol{a_N}]$, where $\boldsymbol{a_i}$ is the column vector of A and $\boldsymbol{a_i} = \left[a_i^1, a_i^2, \cdots, a_i^N\right]^T$.

There exists a series of transformations that decompose A into an orthogonal matrix $\boldsymbol{Q}$ and an upper triangular matrix $\boldsymbol{R}$, i.e., $\boldsymbol{A} = \boldsymbol{QR}$. These transformations are also called QR decomposition of $\boldsymbol{A}$.

The orthogonal matrix $\boldsymbol{Q}$ is set to consist of N column vectors, $\boldsymbol{Q} = [\boldsymbol{q_1}, \boldsymbol{q_2}, \cdots, \boldsymbol{q_i}, \cdots, \boldsymbol{q_N}]$, where $\boldsymbol{q_i} = \left[q_i^1, q_i^2, \cdots, q_i^N\right]^T$.

The row and column vectors of an orthogonal matrix are two-by-two orthogonal unit vectors.

$$\boldsymbol{QQ^T} = \boldsymbol{Q^T Q} = \boldsymbol{E} \tag{9}$$

$$\boldsymbol{q_i}^{\boldsymbol{T}}\boldsymbol{q_j} = \begin{cases} \boldsymbol{0}, \boldsymbol{i} \neq \boldsymbol{j} \\ \boldsymbol{1}, \boldsymbol{i} = \boldsymbol{j} \end{cases} \tag{10}$$

$$\|\boldsymbol{q_i}\|_2 = \boldsymbol{1} \tag{11}$$

Therefore, mapping unordered multi-categorical variables into orthogonal numerical codes can ensure that the different categories are independent of each other in terms of location, are equal distances, and are the same size, and at the same time increase the data richness to avoid underfitting in the subsequent prediction process brought about by data impoverishment.

The steps for encoding unordered multi-categorical variables using an orthogonal matrix are as follows.

(1) Construct N-order square matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ based on the number of categories N in the unordered multi-categorical variables.

(2) Perform the QR decomposition of $\boldsymbol{A}$ using the Gram–Schmidt orthogonal transformation to obtain the orthogonal matrix $\boldsymbol{Q} \in \mathbb{R}^{N \times N}$.

The Gram–Schmidt orthogonal transformation is able to convert non-orthogonal bases into orthogonal bases by first orthogonalizing the non-orthogonal bases and, second, unitizing them.

$$\begin{cases} \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}} = \boldsymbol{a_1} \\ \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{2}} = \boldsymbol{a_2} - \frac{\left\langle \boldsymbol{a_2}, \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}} \right\rangle}{\left\langle \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}}, \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}} \right\rangle} \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}} \\ \cdots \\ \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{N}} = \boldsymbol{a_N} - \frac{\left\langle \boldsymbol{a_N}, \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}} \right\rangle}{\left\langle \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}}, \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}} \right\rangle} \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}} - \frac{\left\langle \boldsymbol{a_N}, \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{2}} \right\rangle}{\left\langle \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{2}}, \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{2}} \right\rangle} \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{2}} \cdots - \frac{\left\langle \boldsymbol{a_N}, \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{N-1}} \right\rangle}{\left\langle \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{N-1}}, \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{N-1}} \right\rangle} \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{N-1}} \end{cases} \tag{12}$$

$$\begin{cases} \boldsymbol{q_1} = \frac{\overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}}}{\left\| \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{1}} \right\|} \\ \boldsymbol{q_2} = \frac{\overset{\sim}{\boldsymbol{q}}_{\boldsymbol{2}}}{\left\| \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{2}} \right\|} \\ \cdots \\ \boldsymbol{q_N} = \frac{\overset{\sim}{\boldsymbol{q}}_{\boldsymbol{N}}}{\left\| \overset{\sim}{\boldsymbol{q}}_{\boldsymbol{N}} \right\|} \end{cases} \tag{13}$$

$$\boldsymbol{Q} = [\boldsymbol{q_1}, \boldsymbol{q_2}, \cdots, \boldsymbol{q_i}, \cdots, \boldsymbol{q_N}] \tag{14}$$

(3) Construct the N-order identity matrix $\boldsymbol{E}$. Using matrix multiplication, map the orthogonal matrix to the original unordered multi-categorical variables one by one according to the position indexes to obtain the orthogonal numerical code matrix $\boldsymbol{X} \in \mathbb{R}^{N \times N}$, which consists of row vectors.

$$CV: \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_i \\ \cdots \\ c_N \end{bmatrix}_{N \times 1} \rightarrow \boldsymbol{X}: \begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \\ \cdots \\ \boldsymbol{x_i} \\ \cdots \\ \boldsymbol{x_N} \end{bmatrix}_{N \times N} = \boldsymbol{E}_{N \times N} \boldsymbol{Q}^T{}_{N \times N} = \begin{bmatrix} \boldsymbol{q_1}^{\boldsymbol{T}} \\ \boldsymbol{q_2}^{\boldsymbol{T}} \\ \cdots \\ \boldsymbol{q_i}^{\boldsymbol{T}} \\ \cdots \\ \boldsymbol{q_N}^{\boldsymbol{T}} \end{bmatrix}_{N \times N} = \begin{bmatrix} q_1^1 & q_1^2 & \cdots & q_1^N \\ q_2^1 & q_2^2 & \cdots & q_2^N \\ & & \cdots & \\ q_i^1 & q_i^2 & \cdots & q_i^N \\ & & \cdots & \\ q_N^1 & q_N^2 & \cdots & q_N^N \end{bmatrix}_{N \times N} \tag{15}$$

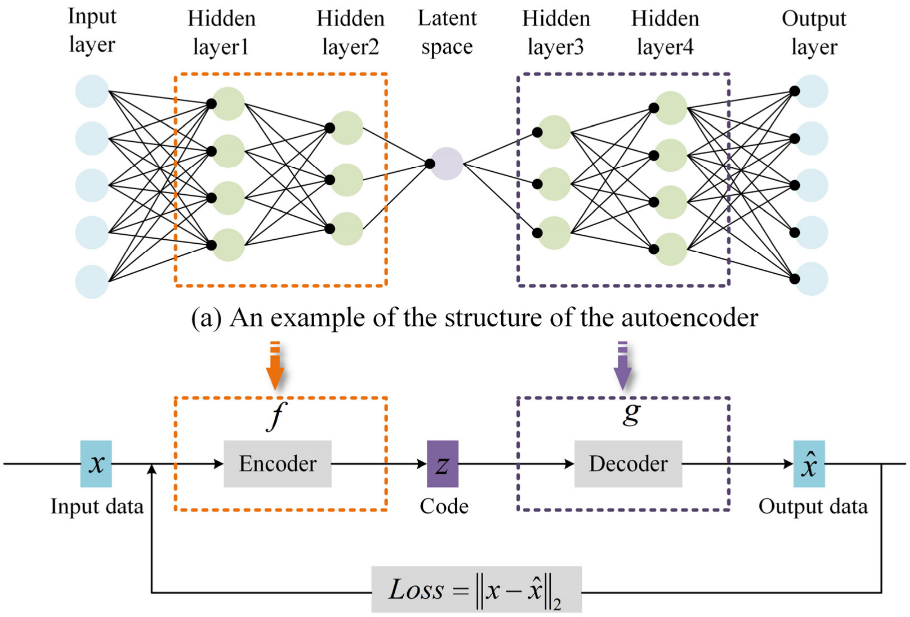
where $\boldsymbol{x_i} = \boldsymbol{q_i}^T$ and $\boldsymbol{x_i}$ is the row vector of $\boldsymbol{X}$.

After the encoding of unordered multi-categorical variables, input the orthogonal matrix of numerical codes $\boldsymbol{X}$ into the autoencoder for representation learning and dimension reduction.

3.2. Code Dimensionality Reduction

An autoencoder (AE) [28] is an unsupervised neural network model, which mainly consists of two parts, an encoder and decoder, and has a symmetric structure. Through neural network training, an autoencoder can achieve the tasks of data denoising, feature learning, or data dimensionality reduction [29].

An example of the structure of the autoencoder and the working principle of the autoencoder is shown in Figure 2. The autoencoder includes an input layer, a hidden layer, latent space, and an output layer, each with a different number of nodes. In this case, the structure from the input layer to the latent space is referred to as the encoder, and the structure from the latent space to the output layer is referred to as the decoder. The number of nodes in the input layer is the size of the original dimension of the data, the number of nodes in the latent space is the dimension in which the data needs to be compressed, and the number of nodes in the hidden layer is in between the original dimension and the compressed dimension.



(a) An example of the structure of the autoencoder

(b) Working principle of the autoencoder

Figure 2. The structure and working principle of the autoencoder.

The working principle of the autoencoder is as follows: the encoder compresses the input data into a predefined low-dimensional encoding through a series of transformations f, and the decoder tries to reconstruct the low-dimensional encoding into the original input data through a series of transformations g. Iterative training of the autoencoder is carried out by minimizing the reconstruction error between the input and the output, and ultimately, it is hoped that the autoencoder learns the abstract feature representation of the samples z.

The working principle of the autoencoder to carry out data dimensionality reduction is shown in Equation (16).

$$\begin{aligned} & f : \mathbb{R} \rightarrow \mathbb{Z} \\ & g : \mathbb{Z} \rightarrow \mathbb{R} \\ & \boldsymbol{z} = f(\boldsymbol{W_1 x} + \boldsymbol{b_1}) \\ & \hat{\boldsymbol{x}} = g(\boldsymbol{W_2 z} + \boldsymbol{b_2}) \\ & f, g = arg\ \underset{f,g}{min} \|\boldsymbol{x} - g(f(\boldsymbol{x}))\|_2 \end{aligned} \tag{16}$$

In this paper, multiple symmetric fully connected layers are designed in the encoder and decoder of the autoencoder to increase the complexity of the model and feature extraction capability by stacking multiple connected layers.

ReLU is used as the activation function; ReLU is simpler to compute and has relatively stable performance. ReLU can avoid the gradient disappearing during model training, increase the nonlinear fitting ability of the model, and have good performance on multi-class tasks and datasets. The ReLU formula is $f(x) = max(0, x)$.

Taking the autoencoder model containing four hidden layers and one-dimensional data compression dimension as an example in Figure 3, $\boldsymbol{X}_{N\times N}$ denotes the orthogonal numerical code matrix, and $\hat{\boldsymbol{X}}_{N\times N}$ denotes the numerical matrix after the reconstruction of the autoencoder. Let $\boldsymbol{X}^{(1)} \in \mathbb{R}^{N\times a}, \boldsymbol{X}^{(2)} \in \mathbb{R}^{N\times b}, a, b < N$ denote the process data compressed by the encoding layer. Let $\hat{\boldsymbol{X}}^{(2)} \in \mathbb{R}^{N\times b}, \hat{\boldsymbol{X}}^{(1)} \in \mathbb{R}^{N\times a}, a, b < N$ denote the process data that has been reconstructed by the decoding layer, and let $\boldsymbol{Z} \in \mathbb{Z}^{N\times 1}$ denote the coded data that has undergone dimensionality reduction.

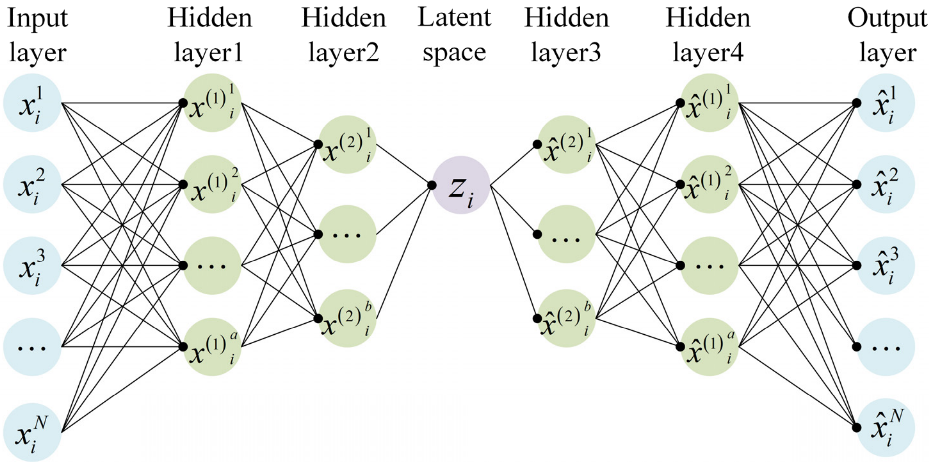


Figure 3. Example of dimensionality reduction using autoencoder.

$\boldsymbol{X}, \boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \hat{\boldsymbol{X}}^{(2)}, \hat{\boldsymbol{X}}^{(1)}, \hat{\boldsymbol{X}}$ all consist of row vectors.

The change in data dimensions during the encoding and decoding of the orthogonal numerical code matrix by AE is shown in Equation (17).

$$\begin{aligned}\boldsymbol{X}_{N\times N} = \begin{bmatrix} \boldsymbol{x_1} \\ \boldsymbol{x_2} \\ \cdots \\ \boldsymbol{x_i} \\ \cdots \\ \boldsymbol{x}_N \end{bmatrix}_{N\times N} &\overset{encoder}{\rightarrow} \boldsymbol{X^{(1)}}_{N\times a} = \begin{bmatrix} \boldsymbol{x^{(1)}}_{\boldsymbol{1}} \\ \boldsymbol{x^{(1)}}_{\boldsymbol{2}} \\ \cdots \\ \boldsymbol{x^{(1)}}_{\boldsymbol{i}} \\ \cdots \\ \boldsymbol{x^{(1)}}_{\boldsymbol{N}} \end{bmatrix}_{N\times a} \overset{encoder}{\rightarrow} \boldsymbol{X^{(2)}}_{N\times b} = \begin{bmatrix} \boldsymbol{x^{(2)}}_{\boldsymbol{1}} \\ \boldsymbol{x^{(2)}}_{\boldsymbol{2}} \\ \cdots \\ \boldsymbol{x^{(2)}}_{\boldsymbol{i}} \\ \cdots \\ \boldsymbol{x^{(2)}}_{\boldsymbol{N}} \end{bmatrix}_{N\times b} \overset{encoder}{\rightarrow} \boldsymbol{Z}_{N\times 1} \\ &= \begin{bmatrix} z_1 \\ z_2 \\ \cdots \\ z_i \\ \cdots \\ z_N \end{bmatrix}_{N\times 1} \overset{decoder}{\rightarrow} \hat{\boldsymbol{X}}^{(\boldsymbol{2})}{}_{N\times b} = \begin{bmatrix} \hat{\boldsymbol{x}}^{(\boldsymbol{2})}{}_{\boldsymbol{1}} \\ \hat{\boldsymbol{x}}^{(\boldsymbol{2})}{}_{\boldsymbol{2}} \\ \cdots \\ \hat{\boldsymbol{x}}^{(\boldsymbol{2})}{}_{\boldsymbol{i}} \\ \cdots \\ \hat{\boldsymbol{x}}^{(\boldsymbol{2})}{}_{\boldsymbol{N}} \end{bmatrix}_{N\times b} \overset{decoder}{\rightarrow} \hat{\boldsymbol{X}}^{(1)}{}_{N\times a} = \begin{bmatrix} \hat{\boldsymbol{x}}^{(\boldsymbol{1})}{}_{\boldsymbol{1}} \\ \hat{\boldsymbol{x}}^{(\boldsymbol{1})}{}_{\boldsymbol{2}} \\ \cdots \\ \hat{\boldsymbol{x}}^{(\boldsymbol{1})}{}_{\boldsymbol{i}} \\ \cdots \\ \hat{\boldsymbol{x}}^{(\boldsymbol{1})}{}_{\boldsymbol{N}} \end{bmatrix}_{N\times a} \overset{decoder}{\rightarrow} \hat{\boldsymbol{X}}_{N\times N} = \begin{bmatrix} \hat{\boldsymbol{x}}_{\boldsymbol{1}} \\ \hat{\boldsymbol{x}}_{\boldsymbol{2}} \\ \cdots \\ \hat{\boldsymbol{x}}_{\boldsymbol{i}} \\ \cdots \\ \hat{\boldsymbol{x}}_N \end{bmatrix}_{N\times N}\end{aligned} \tag{17}$$

More specifically, the steps for the dimensionality reduction of the orthogonal vector $\boldsymbol{x_i}$ in the orthogonal numerical coding matrix are as follows.

(1) By means of the encoder, the orthogonal vector $\boldsymbol{x_i}$ is mapped from a high-dimensional space to a low-dimensional space to achieve data dimensionality reduction.

$$\begin{aligned} \boldsymbol{x^{(1)}}_{\boldsymbol{i}} &= max\left(0, \boldsymbol{W_1^{(1)}} \boldsymbol{x_i} + \boldsymbol{b_1^{(1)}}\right) \\ \boldsymbol{x^{(2)}}_{\boldsymbol{i}} &= max\left(0, \boldsymbol{W_1^{(2)}} \boldsymbol{x^{(1)}}_{\boldsymbol{i}} + \boldsymbol{b_1^{(2)}}\right) \end{aligned} \tag{18}$$

The coded data z_i after feature extraction of the model is obtained by linearly transforming the output of the previous layer to avoid introducing additional nonlinear relationships to enhance the complexity of the coded data, as in Equation (19).

$$z_i = \boldsymbol{W_1^{(3)}} \boldsymbol{x}^{(2)}{}_{\boldsymbol{i}} + \boldsymbol{b_1^{(3)}} \tag{19}$$

(2) The original data $\boldsymbol{x_i}$ is reconstructed by mapping the low-dimensional space back to the high-dimensional space through the decoder.

$$\begin{aligned} \boldsymbol{\hat{x}_i^{(2)}} &= max\left(0, \boldsymbol{W_2^{(1)}} z_i + \boldsymbol{b_2^{(1)}}\right) \\ \boldsymbol{\hat{x}_i^{(1)}} &= max\left(0, \boldsymbol{W_2^{(2)}} \boldsymbol{\hat{x}_i^{(2)}} + \boldsymbol{b_2^{(2)}}\right) \end{aligned} \tag{20}$$

The final output of the decoding layer, $\boldsymbol{\hat{X}}$, is obtained by linearly transforming the output of the previous layer to satisfy the purpose of output data reconstruction, as in Equation (21).

$$\boldsymbol{\hat{x}_i} = \boldsymbol{W_2^{(3)}} \boldsymbol{\hat{x}_i^{(1)}} + \boldsymbol{b_2^{(3)}} \tag{21}$$

(3) The error is calculated between the reconstructed data $\boldsymbol{\hat{X}_{N\times N}}$ and the original data $\boldsymbol{X}_{N\times N}$, generally using the mean-square error as a loss function L.

$$L = \|\boldsymbol{x_i} - \boldsymbol{\hat{x}_i}\|_2 \tag{22}$$

(4) With the goal of minimizing the reconstruction error, the weights $\boldsymbol{W}$ and bias $\boldsymbol{b}$ in the encoder and decoder are optimised by error back propagation using a gradient-based method for the purpose of training the autoencoder.

$$f, g_{(\boldsymbol{W},\boldsymbol{b})} = arg \underset{f,g}{min} L(\boldsymbol{x_i}, \boldsymbol{\hat{x}_i}) \tag{23}$$

(5) After autoencoder training, the autoencoder learns the feature z_i of the data.

$$\boldsymbol{x_i} \xrightarrow{f_{O-AE}} z_i \tag{24}$$

The same is true for the other orthogonal vectors, which ultimately results in the reduced dimensionality of the orthogonal numerical code matrix with numerical code $\boldsymbol{Z} \in \mathbb{R}^{N\times 1}$.

$$\boldsymbol{X}_{N\times N} \xrightarrow{f_{O-AE}} \boldsymbol{Z}_{N\times 1} = \begin{bmatrix} z_1 \\ z_2 \\ \cdots \\ z_i \\ \cdots \\ z_N \end{bmatrix}_{N\times 1} \tag{25}$$

From Equation (3), the values are the same after encoding in the same category. Therefore, the coded values after dimensionality reduction are expanded to the original sample size $\boldsymbol{Z} \in \mathbb{R}^{\tilde{N}\times 1}$ to obtain the coded expressions for all unordered multi-categorical variables.

$$CV : \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_i \\ \cdots \\ c_N \end{bmatrix}_{N\times 1} \to CV = \begin{bmatrix} c_1 \\ c_2 \\ \cdots \\ c_{i1} \\ c_{i2} \\ \cdots \\ c_N \end{bmatrix}_{\tilde{N}\times 1} \overset{f_{O-AE}}{\to} \boldsymbol{Z}_{\tilde{N}\times 1} = \begin{bmatrix} z_1 \\ z_2 \\ \cdots \\ z_{i1} \\ z_{i2} \\ \cdots \\ z_N \end{bmatrix}_{\tilde{N}\times 1} \tag{26}$$

4. Bayesian Optimization of Autoencoder Hyperparameters

The structure of the autoencoder model is not fixed. Since a dataset may include multiple unordered multi-categorical variables with varying numbers of categories, the number of hidden layers and nodes in the autoencoder adapts according to the dataset's size and the required level of data compression. Therefore, parameters such as the number of layers and nodes in the hidden layer of the autoencoder are the key influencing factors that affect the performance of the autoencoder and the effect of data dimensionality reduction.

Bayesian optimization is suitable for black-box optimization problems where the target function is complex and has no analytical expression and the computational cost is high. The core idea is to use a probabilistic agent model (e.g., Gaussian process or random forest) to approximate the objective function and use this model to select the next evaluation point so that the objective function can converge to the optimal value as soon as possible, which is a kind of optimization method based on a priori information. There are two core components in this process: the probabilistic model and the acquisition strategy.

Bayes' theorem is shown in Equation (27).

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} \tag{27}$$

This paper uses the TPE (Tree-structured Parzen Estimator) algorithm in the Optuna framework to specify the implementation of the Bayesian optimization technique. The TPE is a tree-structured Bayesian optimization method. In the TPE, two density functions are used to define $p(x|y)$.

$$p(x|y) = \begin{cases} l(x), y < y^* \\ g(x), y \geq y^* \end{cases} \tag{28}$$

The TPE uses Kernel Density Estimation (KDE) to model the probability distribution of objective function values. Specifically, the TPE divides the objective function value into two components: a probability distribution $l(x)$ for the better objective value and a probability distribution $g(x)$ for the worse objective value.

The TPE uses Expected Improvement (EI) as a collection function.

$$EI_{y^*}(x) = \int_{-\infty}^{+\infty} max(y^* - y, 0) p_M(y|x) dy = \frac{\int_{-\infty}^{y^*} (y^* - y, 0) p(y) dy}{\gamma + (1-\gamma)\frac{g(x)}{l(x)}} \tag{29}$$

This leads to Equation (30).

$$EI_{y^*}(x) \propto \left(\gamma + (1-\gamma)\frac{g(x)}{l(x)}\right)^{-1} \tag{30}$$

The value of EI is proportional to $\left(\gamma + (1-\gamma)\frac{g(x)}{l(x)}\right)^{-1}$ and depends on the ratio of the two probabilities, so it is necessary to find the x that makes the ratio $\frac{l(x)}{g(x)}$ maximal and, in each iteration, return the maximum EI.

5. Experimental Design

5.1. Dataset

In order to validate the feasibility and effectiveness of the proposed method, this section carried out example validation on five datasets and compared the results with those of one-hot encoding, embedding, label encoding, and target encoding.

In this paper, the working-hour dataset of two typical majors in the cruise ship production design process, namely piping majors and hull structure majors, were selected from the major cruise shipbuilding yards in domestic market, and three public datasets containing unordered multi-categorical variables from UCI and Kaggle were also selected. The sample sizes of the five datasets as well as the number of categories in the unordered multi-categorical variables differed from each other, which made the method of this paper feasible to validate the universality of the method.

In particular, the sample size in the Gait dataset was so large that we randomly selected 1000 pieces of data to form a new dataset for the experiment.

The sample size of the data set, the number of variables, the number of categorical variables, the number of categories in the categorical variables, and the targets used for prediction are shown in the following table, Table 1.

Table 1. Specific information on the dataset.

Dataset	Sample Size ($\tilde{N}$)	Total Number of Variables	Number of Continuous Variables	Categorical Variable			Target
				Number of Categorical Variables	Name	Number of Categories (N)	
Piping	163	8	6	2	area task type	16 19	Feedback working hours
Hull structure	144	4	2	2	area task type	67 4	Feedback working hours
Gait [30]	1000	6	2	4	subject condition joint leg	10 3 3 2	angle
Restaurant [31]	1000	7	5	2	Cuisine_Type Promotions	4 2	Monthly_Revenue
Fish [32]	159	6	5	1	Species	7	Weight

5.2. Detailed Experimental Design

The dimension of the values obtained by different encoding methods is not the same. In this paper, we set the dimension of autoencoder compression to be one-dimensional.

Therefore, O-AE, label encoding, and target encoding received one-dimensional codes, one-hot encoding received the same dimensionality as the number of categories in the categorical variables, and embedding methods were free to set the dimensionality size.

We were concerned with the effect of different encoding methods on the input dimensions of the dataset as well as on the target prediction results of the neural network, so we designed 2 experiments with different embedding dimensions for embedding and for O-AE, one-hot encoding, embedding, label encoding, target encoding, and Bayesian optimization of the hyperparameters of the autoencoder for the O-AE-b method, respectively, in 1 experiment.

The encoded dataset was input to BP neural network to carry out target prediction.

In this paper, only the multi-categorical variables "subject" in the Gait dataset were encoded; the categorical variables "condition" and "joint" in the Gait dataset contained a small number of categories, and "leg" was a binary categorical variable, so the experiments were conducted using the labeled data that came with the original dataset.

The multi-categorical variables "Cuisine_Type" in the Restaurant dataset were encoded and the binary categorical variables "Promotions" were experimented with using the labeled data that came with the original data.

Continuous variables in the dataset were mapped to a standard normal distribution with a mean of 0 and standard deviation of 1 by Z-Score standardization.

The specific methods of the seven experiments and the total dimensions of the five datasets input to the BP neural network under each of the seven experimental codes are shown in Table 2.

Table 2. The experimental design and the total input dimensions after encoding.

No.	Encoding Method	Code Dimension	Total Input Dimensions					Experiments Abbreviation
			Piping	Hull Structure	Gait	Restaurant	Fish	
1	Orthogonal matrix-autoencoder-based encoding method	1	8	4	6	7	6	O-AE
2	Embedding	1	8	4	6	7	6	1-EE
3		5	16	12	10	11	10	5-EE
4	One-hot encoding	Number of categories (N)	31	73	15	10	12	OHE
5	Label encoding	1	8	4	6	7	6	LE
6	Target encoding	1	8	4	6	7	6	TE
7	Bayesian optimization of the hyperparameters of the autoencoder for O-AE	1	8	4	6	7	6	O-AE-b

5.3. Evaluation Metrics

In this paper, three classical evaluation metrics were used to assess the performance of different encoding methods in the BP neural network target prediction task, including Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and coefficient of determination (R^2).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - y_i|,\ MAE \in [0,\ +\infty] \tag{31}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2},\ RMSE \in [0, +\infty] \tag{32}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2},\ \overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i,\ R^2 \in [0, 1] \tag{33}$$

where n is the number of samples, y_i is the actual value, $\hat{y}_i$ is the predicted value, and $\overline{y}$ is the mean of the actual value.

MAE is the mean of the absolute errors, and RMSE is the square root of the mean of the sum of squares of the errors. MAE and RMSE are both measurements of the deviation of the predicted values from the actual values, with smaller values indicating better final predictions. R^2 is used to assess the linear relationship between the actual values and the predicted values, with values closer to 1 indicating a better fit of the model.

6. Example Validation

In this section, we analyze the performance of seven experiments. We implemented all the algorithms in Python and conducted all the experiments on a computer with an i7-11390H processor, 3.40 GHz CPU, and 16.0 GB RAM.

6.1. Parameter Setting

The relevant parameters of the autoencoder when carrying out Experiment 1 and Experiment 7 are shown in Table 3 below. The batch size is set as the sample size of the dataset due to the small sample size of the dataset for the experimental design.

Table 3. The parameter setting of the autoencoder in O-AE and O-AE-b.

Experiment	Loss Function	Activation Function	Optimizer	Learning Rate	Epoch	Batch Size	Patience
O-AE	MSE	Relu	Adam	0.001	2000	Sample size $(\widetilde{N})$	-
O-AE-b	MSE	Relu	Adam	0.001	2000	Sample size $(\widetilde{N})$	100

In particular, in Experiment 7, the unordered multi-categorical variables are divided into training and validation sets according to 7:3. The early-stopping strategy is introduced, and the patience parameter for early stopping is set to 100. The number of AE hidden layers and the number of nodes per layer are obtained with Bayesian optimization. The number of encoder hidden layers is set to be no more than five, and the decoder and encoder structures are symmetric.

The number of hidden layers of the autoencoder and the number of nodes in each layer in Experiment 1 are manually and dynamically adjusted according to different datasets. The epoch of Experiment 1 was kept the same as Experiment 7. Also, since there is no additional autoencoder hyperparameter optimization procedure for Experiment 1 and in order to compare with other experiments to demonstrate the applicability of Experiment 1, the early-stopping strategy is not used.

To carry out target prediction using a BP neural network, the dataset is divided into a training set, validation set, and test set in the ratio of 7:2:1.

The relevant parameters are shown in Table 4. The batch size is set as the sample size of the dataset due to the small sample size of the dataset for the experimental design. In order to quickly conduct experiments and obtain a better prediction result, the hyperparameter optimization method of a 5-fold cross-validation grid search is used, and the number of nodes in each hidden layer is set to be the same; the number of optional hidden layers and the number of optional nodes are shown in Table 4. In order to improve the model performance and avoid overfitting, we introduce L2 Regularization and early-stopping strategy and set the patience parameter of early stopping to 100.

Table 4. The parameter setting of the BP neural network.

Loss Function	Activation Function	Optimizer	Learning Rate	Epoch
MSE	Relu	Adam	0.001	1000
L2 Regularization	**Batch size**	**Patience**	**Optional hidden layers**	**Optional nodes**
0.0001	Sample size $(\widetilde{N})$	100	10, 20, 30	1, 2, 3

Meanwhile, the original code generation step randomly generates N-order square matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ based on the number of categories N in the unordered multi-categorical variables and fixes the random seed to ensure that the experimental data are the same.

6.2. Target Prediction Results

Under the above parameter settings, several calculations are carried out to obtain the results of the evaluation metrics for the five datasets in seven experiments. The number of nodes per layer of the encoding layer of the autoencoder (the decoding layer is symmetrical to the structure of the encoding layer, so the description is omitted) and the number of layers and nodes in the hidden layer of the BP neural network after hyperparameter optimization are shown in Table 5.

Table 5. Optimal hyperparameter results for autoencoder and BP neural network.

Dataset	Hyperparameter		O-AE	1-EE	5-EE	OHE	LE	TE	O-AE-b
Piping	AE	area	16-6-1			-			16-13-11-9-5-1
		task type	19-6-1						19-18-16-13-12-1
	BP	layers	3	3	3	3	3	3	3
		nodes	30	10	20	30	30	20	30
Hull structure	AE	area	67-30-6-1			-			67-26-17-1
		task type	4-2-1						4-2-1
	BP	layers	3	3	3	3	3	3	3
		nodes	30	20	30	30	20	30	30
Gait	AE	subject	10-4-1			-			10-9-8-5-1
	BP	layers	3	3	3	3	3	3	3
		nodes	30	30	30	30	30	30	30
Restaurant	AE	Cuisine_Type	4-3-1			-			4-2-1
	BP	layers	3	3	3	3	3	3	3
		nodes	30	30	10	20	30	20	30
Fish	AE	Species	7-3-1			-			7-6-5-1
	BP	layers	3	3	3	3	3	3	3
		nodes	30	30	30	30	30	30	30

In this paper, we denote the model structure of the autoencoder by unordered multi-categorical variables' original dimension − intermediate compression dimension − 1.

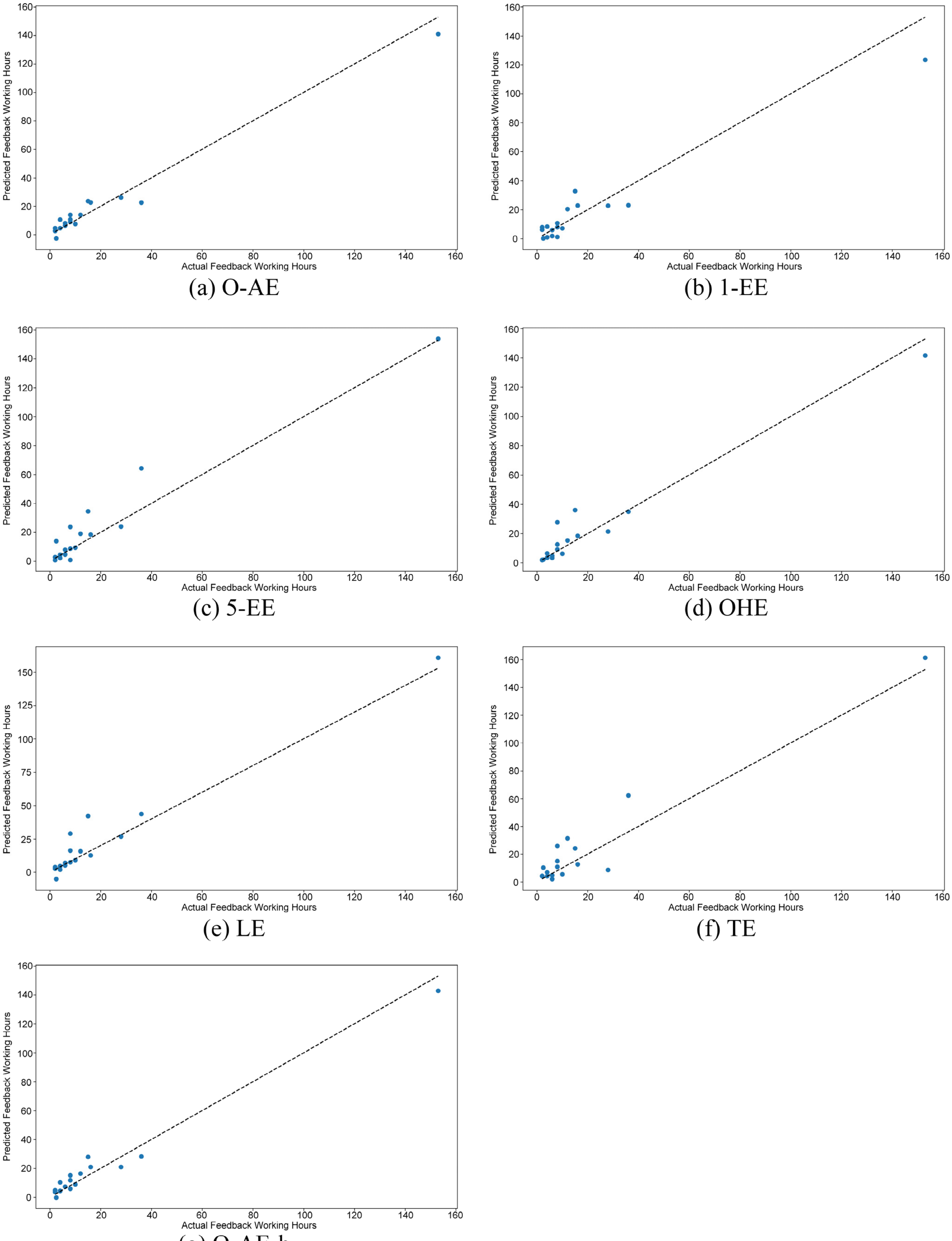
The results of the evaluation metrics for the five datasets in seven experiments are shown in Table 6.

Table 6. The results of the evaluation metrics.

		O-AE	1-EE	5-EE	OHE	LE	TE	O-AE-b
Piping	MAE	**4.3681**	6.8860	6.1809	4.8380	5.6480	**8.1285**	4.5841
	RMSE	5.8296	9.8875	9.9672	7.9449	9.2587	**11.1007**	**5.7244**
	R^2	0.9718	0.9189	0.9175	0.9476	0.9288	**0.8977**	**0.9728**
Hull structure	MAE	8.4987	12.3219	10.6569	6.4534	9.9077	**14.3765**	**5.8390**
	RMSE	10.2222	17.5980	15.5944	12.5951	15.5421	**19.2587**	**7.6556**
	R^2	0.9628	0.8896	0.9133	0.9435	0.9139	**0.8678**	**0.9791**
Gait	MAE	3.4295	3.4297	3.1289	3.6434	3.8893	**4.2187**	**3.3678**
	RMSE	4.5314	4.7570	4.6198	5.1868	5.1943	**5.7029**	**4.2098**
	R^2	0.9319	0.9250	0.9292	0.9108	0.9105	**0.8921**	**0.9412**
Restaurant	MAE	**40.8338**	**42.3132**	42.3016	41.4832	41.1913	41.6640	40.9350
	RMSE	52.6461	**54.2528**	53.0498	52.8340	53.3646	52.6894	**51.9348**
	R^2	0.7559	**0.7407**	0.7521	0.7541	0.7492	0.7555	**0.7624**
Fish	MAE	21.7449	24.5180	22.9351	**20.7503**	23.8476	**56.8344**	20.8339
	RMSE	32.1668	34.4302	34.8801	33.4807	33.7599	**72.4618**	**31.8064**
	R^2	0.9867	0.9848	0.9844	0.9856	0.9854	**0.9327**	**0.9870**

Bold indicates the best and worst values of the evaluation metrics.

This paper plots the results of the target prediction experiments for each dataset, including the fitted regression plots between the predicted and actual values in the test set, as shown in Figures 4–8, and the line charts between the predicted and actual values in the test set, as shown in Figures 9–13.



(a) O-AE

(b) 1-EE

(c) 5-EE

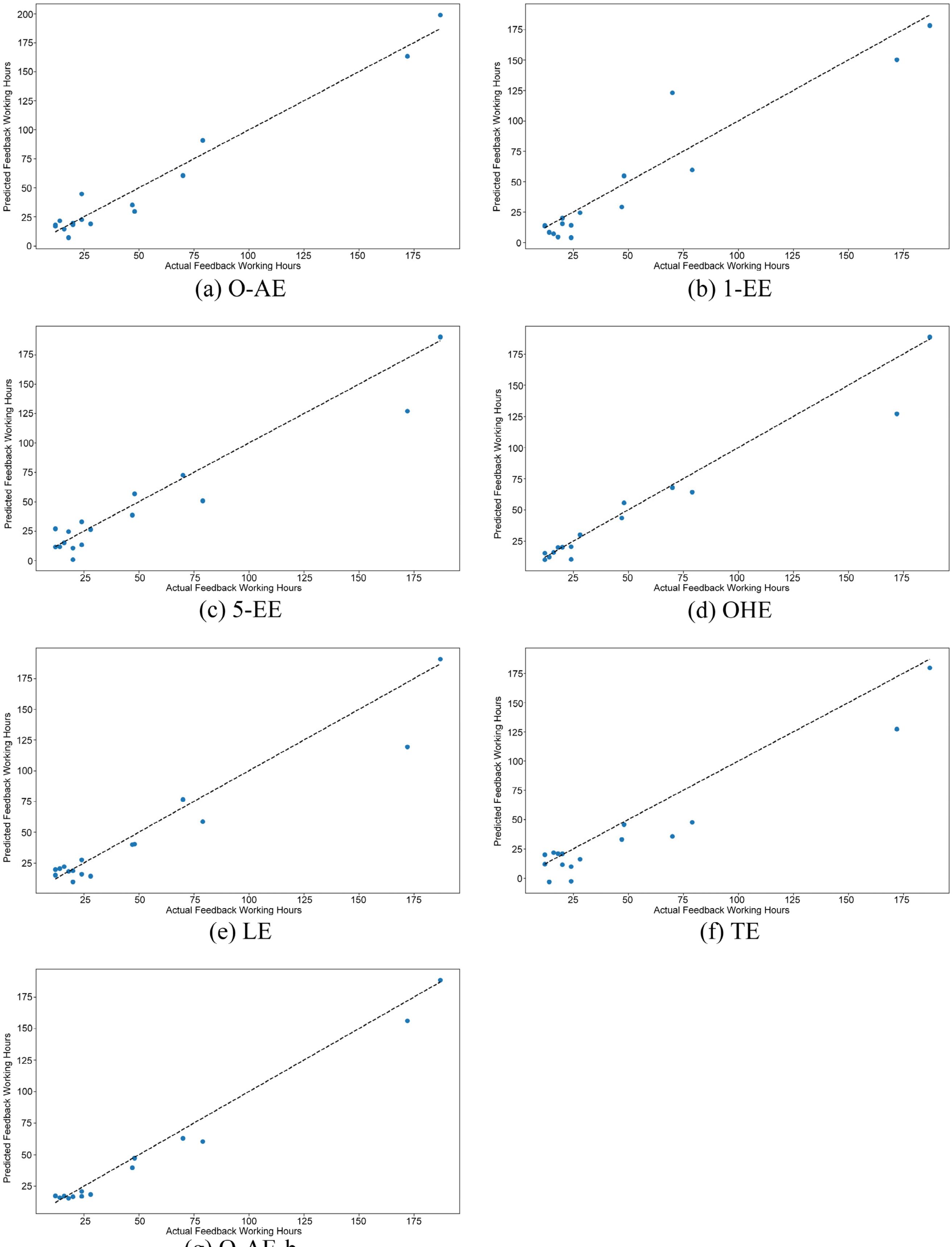
(d) OHE

(e) LE

(f) TE

(g) O-AE-b

Figure 4. Piping dataset–fitted regression plots.



(a) O-AE

(b) 1-EE

(c) 5-EE

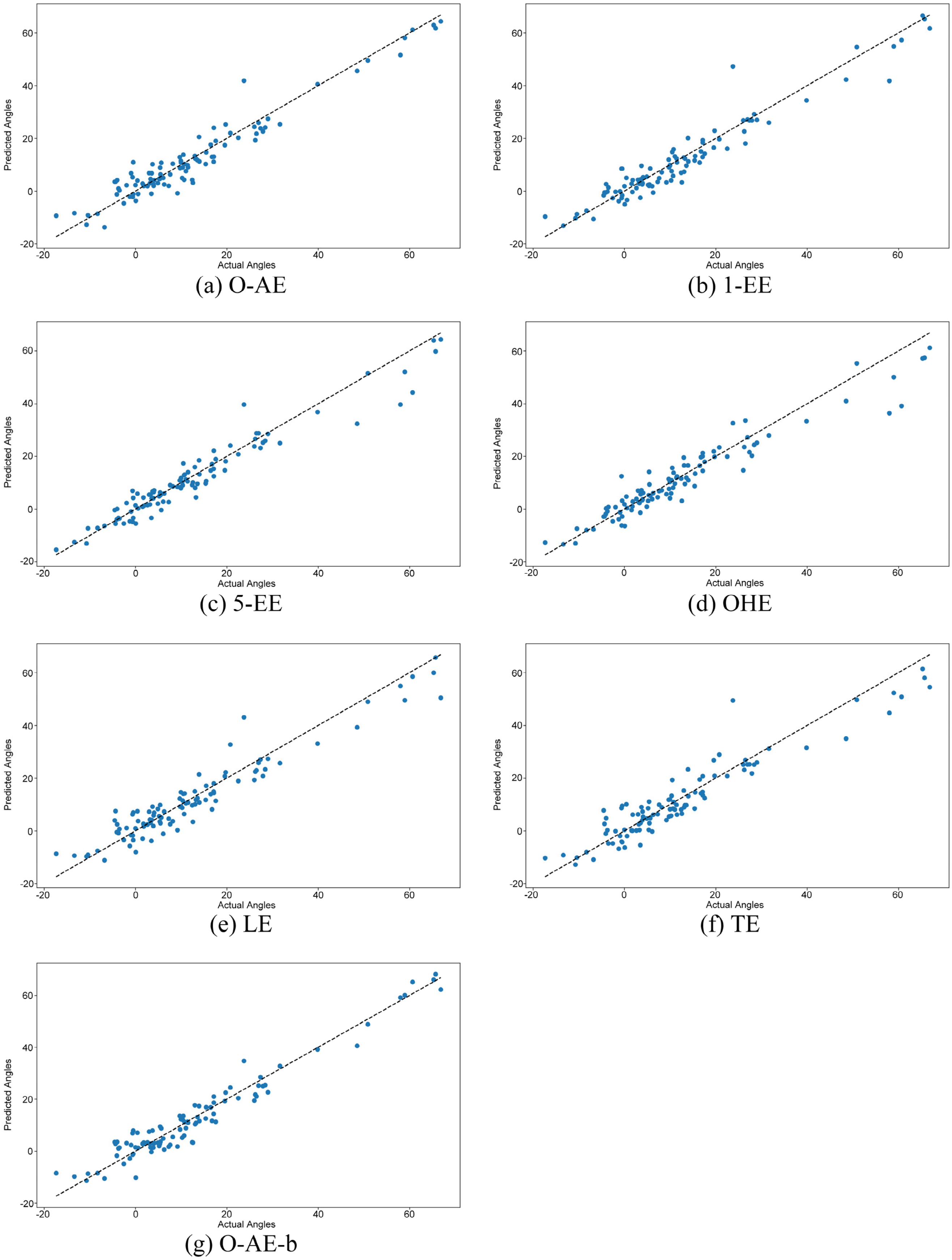
(d) OHE

(e) LE

(f) TE

(g) O-AE-b

Figure 5. Hull structure dataset–fitted regression plots.



(a) O-AE

(b) 1-EE

(c) 5-EE

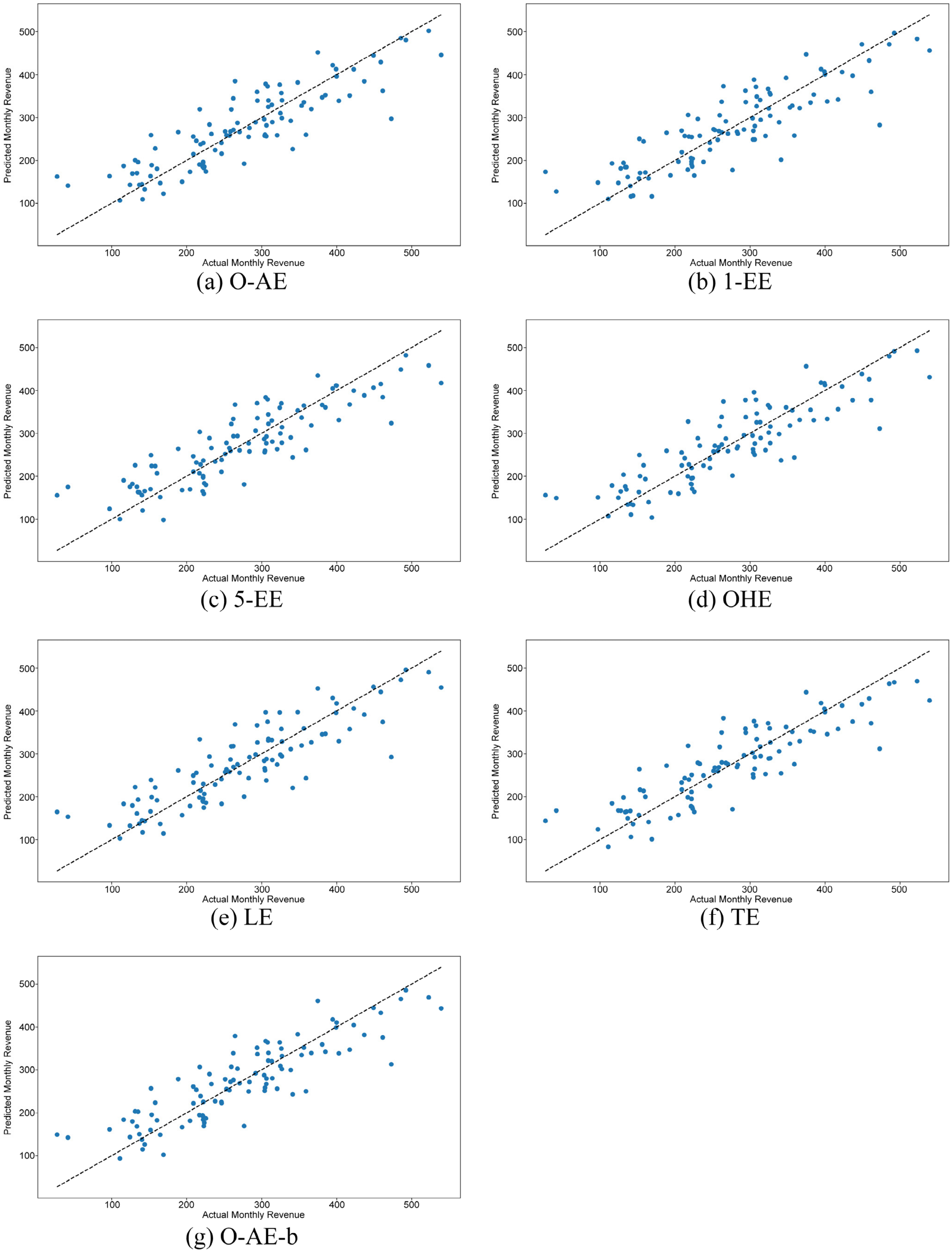
(d) OHE

(e) LE

(f) TE

(g) O-AE-b

Figure 6. Gait dataset–fitted regression plots.



(a) O-AE

(b) 1-EE

(c) 5-EE

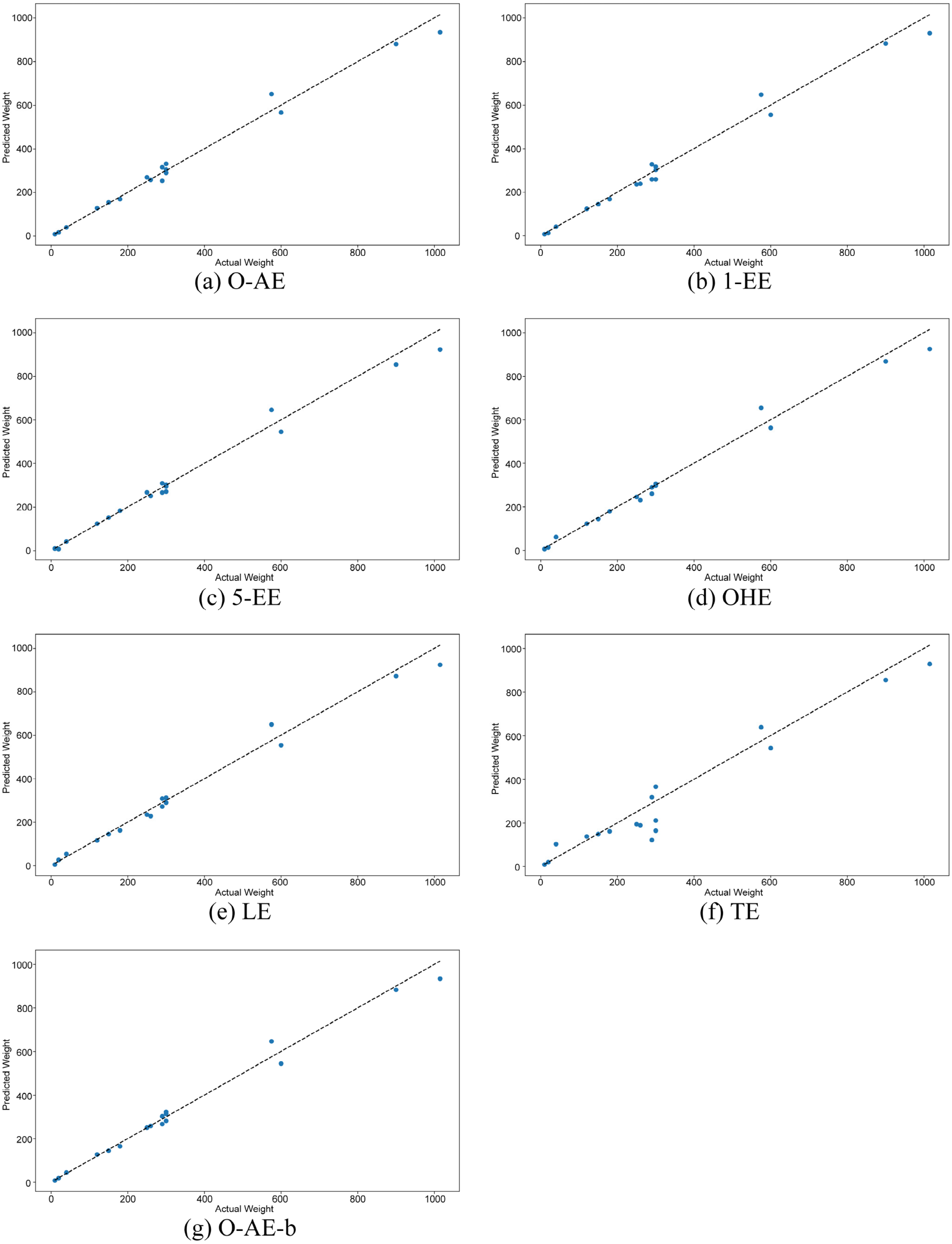
(d) OHE

(e) LE

(f) TE

(g) O-AE-b

Figure 7. Restaurant dataset–fitted regression plots.



(a) O-AE

(b) 1-EE

(c) 5-EE

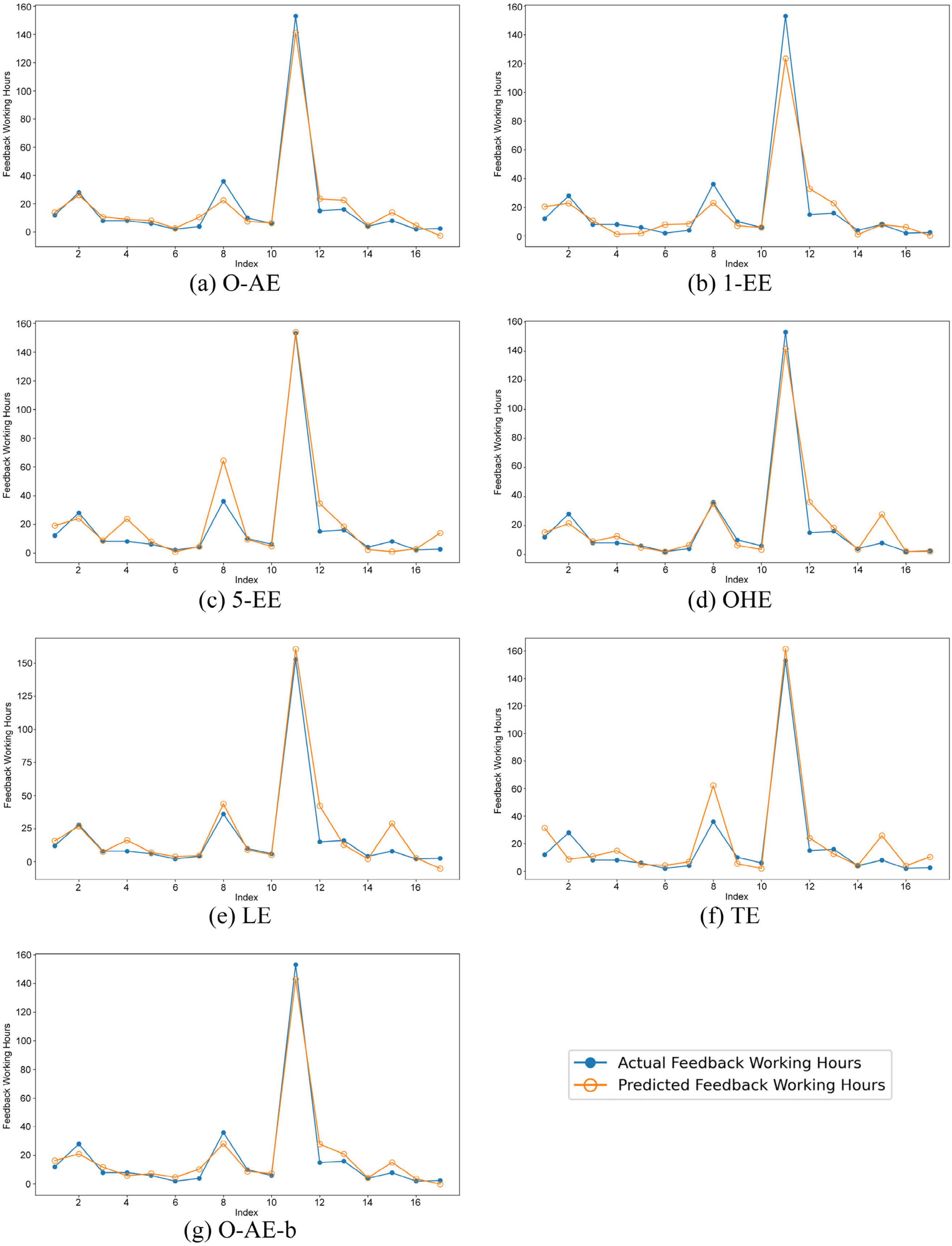
(d) OHE

(e) LE

(f) TE

(g) O-AE-b

Figure 8. Fish dataset–fitted regression plots.



(a) O-AE

(b) 1-EE

(c) 5-EE

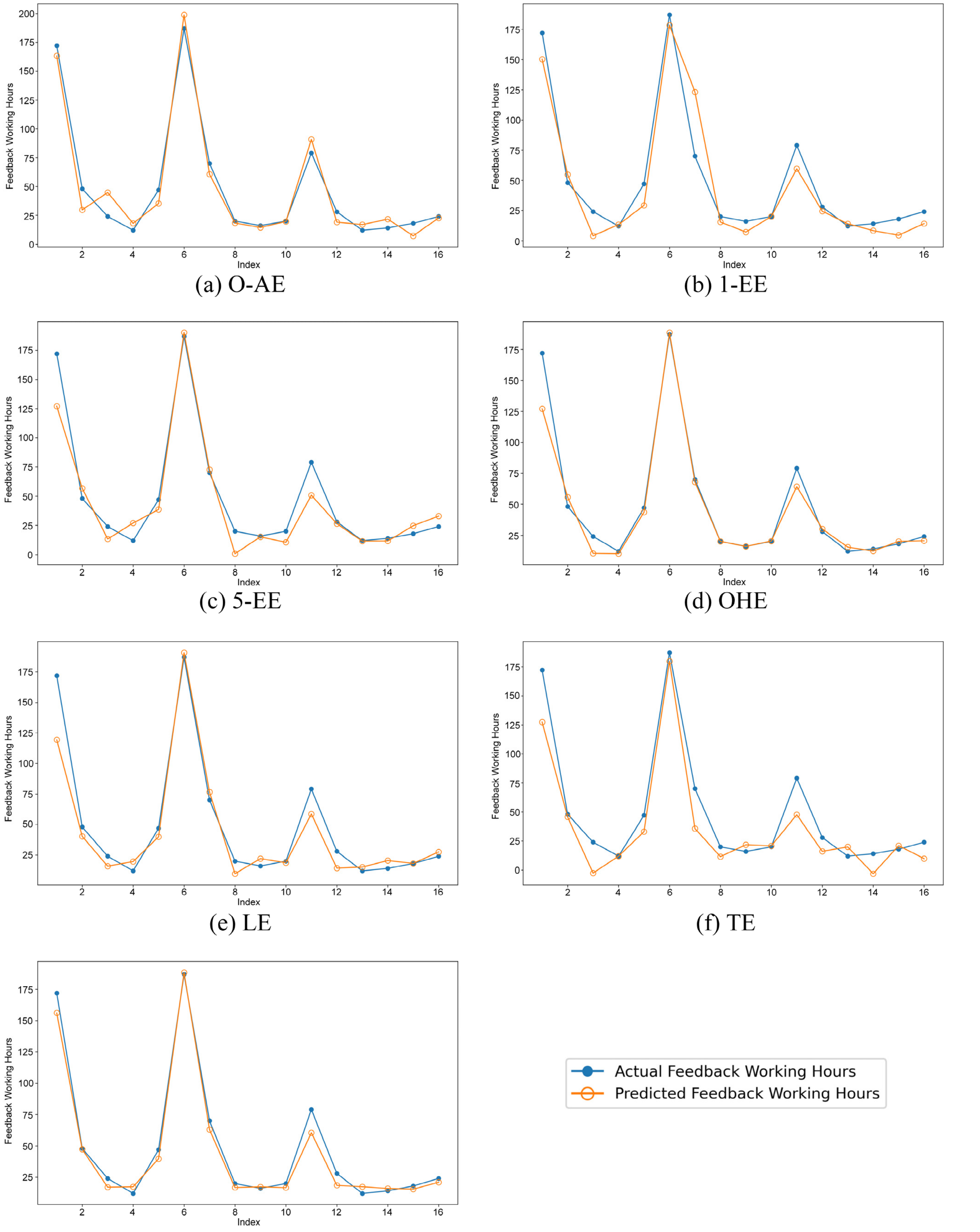
(d) OHE

(e) LE

(f) TE

(g) O-AE-b

Figure 9. Piping dataset–line charts.



(a) O-AE

(b) 1-EE

(c) 5-EE

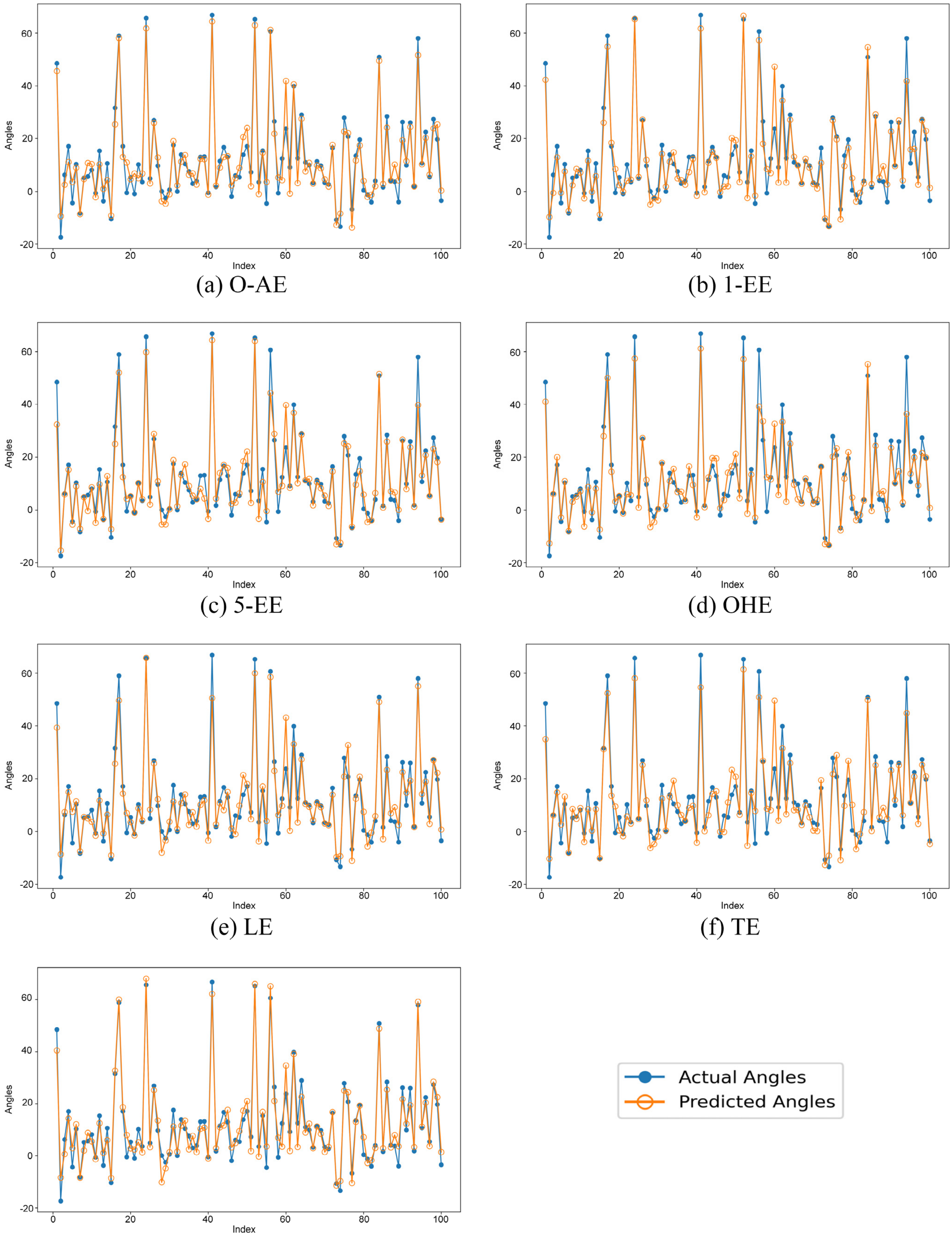
(d) OHE

(e) LE

(f) TE

(g) O-AE-b

Figure 10. Hull structure dataset–line charts.



(a) O-AE

(b) 1-EE

(c) 5-EE

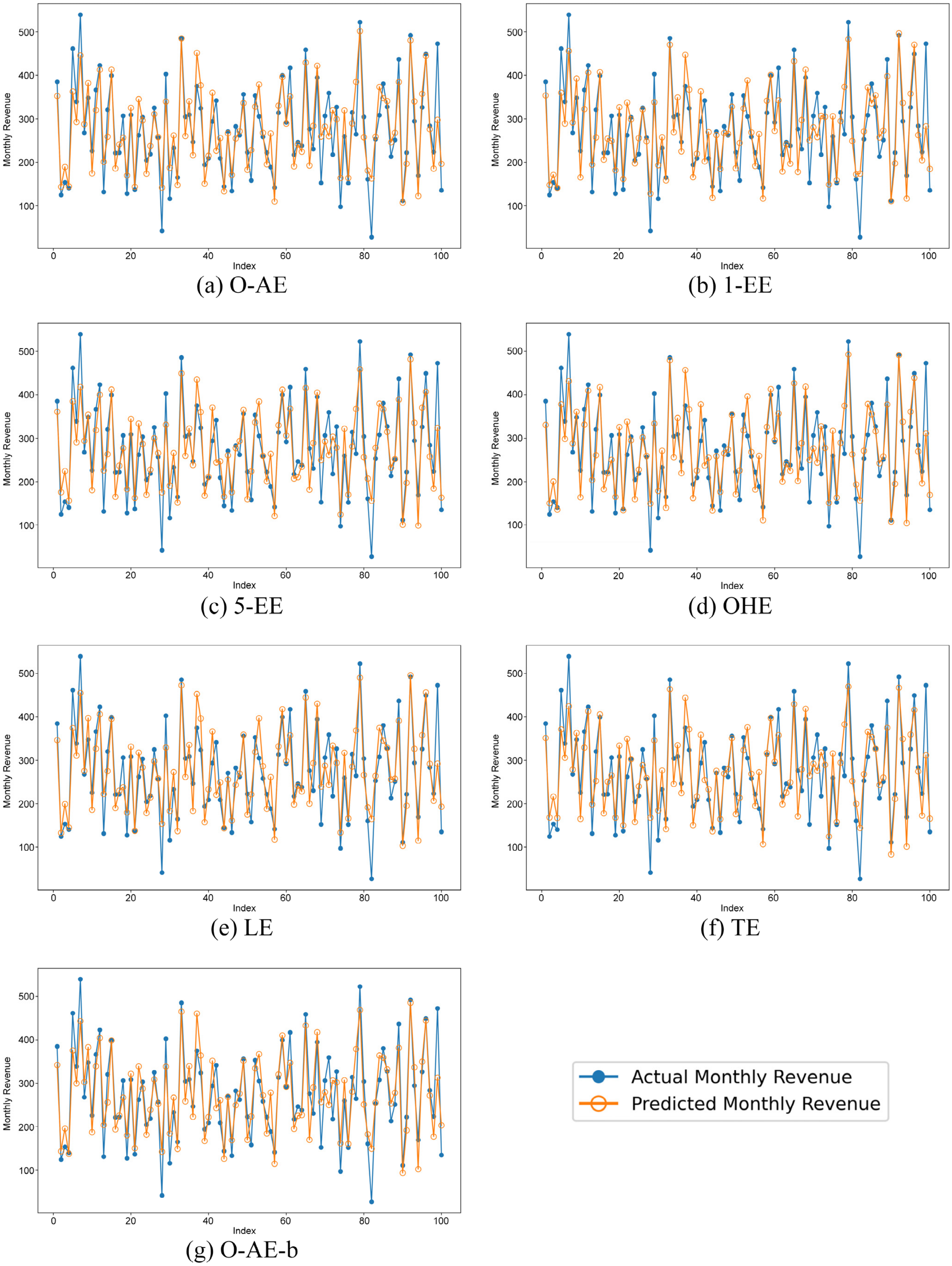
(d) OHE

(e) LE

(f) TE

(g) O-AE-b

Figure 11. Gait dataset–line charts.



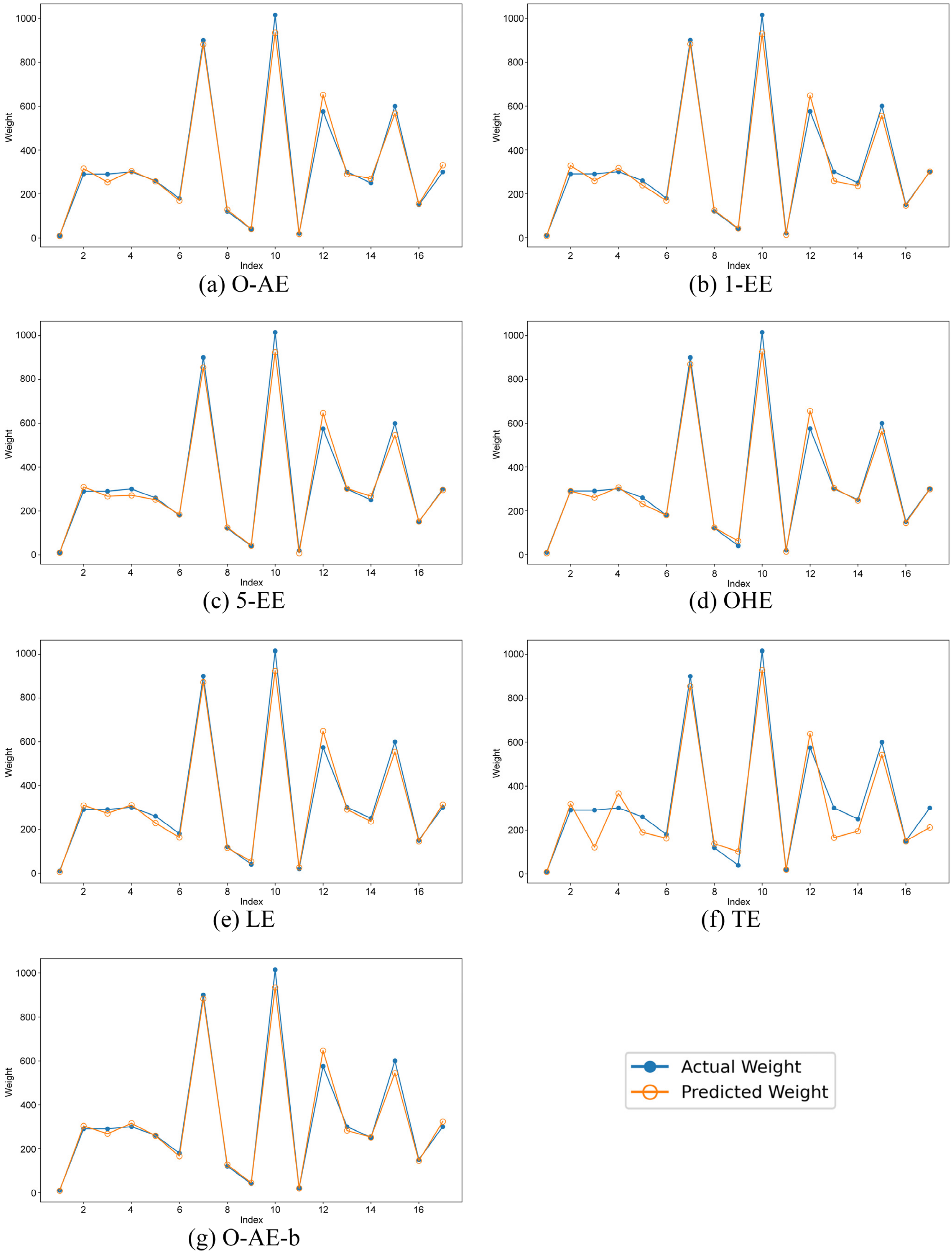
(a) O-AE (b) 1-EE

(c) 5-EE (d) OHE

(e) LE (f) TE

(g) O-AE-b

Figure 12. Restaurant dataset–line charts.



(a) O-AE

(b) 1-EE

(c) 5-EE

(d) OHE

(e) LE

(f) TE

(g) O-AE-b

Figure 13. Fish dataset–line charts.

6.3. Analysis of Results

In this paper, MAE, RMSE and R^2 are chosen as the evaluation metrics of target prediction results, and the smaller MAE and RMSE, and the closer R^2 is to 1 means the target prediction quality is better.

(1) Comparing the metrics results of target prediction.

In the Piping dataset, the MAE results obtained by O-AE are the best, the RMSE and R^2 results obtained by O-AE-b are the best, and the MAE results of O-AE-b are the best among the seven experiments except for O-AE. The RMSE and R^2 results of O-AE are also the best among the seven experiments except for O-AE-b. The results of all three metrics for TE are worse in the other experiments.

In the Hull structure dataset, the MAE, RMSE, and R^2 results obtained by O-AE-b are the best, and the RMSE and R^2 results obtained by O-AE are also the best among the seven experiments except for O-AE-b. The results of all three metrics for TE are worse in the other experiments.

In the Gait dataset, the MAE results obtained by 5-EE are the best, the RMSE and R^2 results obtained by O-AE-b are the best, and the MAE results of O-AE-b are the best among the seven experiments except for 5-EE. The RMSE and R^2 results of O-AE are also the best among the seven experiments except for O-AE-b. The results of all three metrics for TE are worse in the other experiments.

In the Fish dataset, the MAE results obtained by OHE are the best, the RMSE and R^2 results obtained by O-AE-b are the best, and the MAE results of O-AE-b are the best among the seven experiments except for OHE. The RMSE and R^2 results of O-AE are also the best among the seven experiments except for O-AE-b. The results of all three metrics for TE are worse in the other experiments.

In particular, the prediction results of the seven experiments on the Restaurant dataset are not very satisfactory, with the highest R^2 of only 0.7624, which is analyzed in this paper as a possible reason for the fact that the chosen BP neural network model is not applicable to the Restaurant dataset. However, in the Restaurant dataset, the MAE results obtained by O-AE are the best, the RMSE and R^2 results obtained by O-AE-b are the best, and the MAE results of O-AE-b are the best among the seven experiments except for O-AE. The RMSE and R^2 results of O-AE are also the best among the seven experiments except for O-AE-b. The results of all three metrics for 1-EE are worse in the other experiments.

Overall, the Bayesian optimization of the hyperparameters of the autoencoder for O-AE as well as the basic O-AE have consistently excellent target prediction results when experiments are carried out on different datasets. Analyzing the metric results, the basic O-AE can then meet the encoding needs of unordered multi-categorical variables, while the MAE results of O-AE are better than the MAE results of O-AE-b in some datasets.

(2) Comparing the experimental performance of different methods.

In other experiments, 1-EE and 5-EE are embedding methods with one and five embedding dimensions, respectively, and the metric results of 1-EE and 5-EE are in the middle of the range on the five datasets, with little difference in experimental performance. The metric results of OHE on the five datasets are also more stable, and the experimental performance is a little bit poorer than that of O-AE and O-AE-b, but OHE has the problems of unordered multi-categorical variables with larger dimensions and sparse data after encoding. The experimental performance of LE and TE on the five datasets is general mainly because the use of LE introduces additional order misclassification and distance bias in addition to the originally unordered features, and TE affects the fitting ability of the neural network model.

Overall, the experimental results of O-AE-b and O-AE are superior compared to other classical encoding methods. The unordered multi-categorical variables processed by O-AE-b and O-AE were able to satisfy the data input requirements of the subsequent neural network model and facilitated the ability of neural network data learning to improve the neural network target prediction results.

7. Conclusions

The objective of this paper is to find an encoding method for unordered multi-categorical variables that results in the lower dimensionality of the encoded data and better target prediction results when fed into a neural network.

After analyzing the characteristics of unordered multi-categorical variables and comparing them with other encoding methods, this paper proposes a method for the encoding and dimensionality reduction of unordered multi-categorical variables using an orthogonal matrix-autoencoder. Seven experiments are designed for validation using the basic O-AE, the Bayesian optimization of the hyperparameters of the autoencoder for O-AE, and several other classical encoding methods. The experimental results show that O-AE and O-AE-b have more stable and excellent evaluate metrics, indicating that the method proposed in this paper is highly feasible and applicable and can be an optional method for data processing of unordered multi-categorical variables.

Through the experimental analysis, the basic O-AE can meet the encoding requirements of unordered multi-categorical variables, but O-AE manually adjusts the number of layers and nodes in the hidden layer of the autoencoder and needs to adjust the parameters several times to achieve the optimal results of the metrics. O-AE-b uses Bayesian optimization to find the optimal number of layers and nodes in the hidden layer, but the Bayesian optimization of hyperparameters consumes more time than that of the basic O-AE. Therefore, the encoding method can be selected according to the characteristics of the problem and the demand of experimental resources in the specific use.

Although the research in this paper has achieved preliminary results, it still contains vast room for deepening. Future work will focus on optimizing the model performance and try to introduce activation functions such as PReLU or other regularization strategies such as L1-L2 with a view to further improve the model performance. Meanwhile, we will also work on improving the efficiency of Bayesian optimization in the O-AE-b model and exploring more new ways to efficiently optimize the hyperparameters of the autoencoder in order to promote the continuous progress of research in this area.

Author Contributions: Conceptualization, Y.W. and J.L.; methodology, Y.W.; software, Y.W.; validation, Y.W. and J.L.; formal analysis, B.Y., D.S. and L.Z.; investigation, Y.W. and B.Y.; resources, J.L.; data curation, B.Y., D.S. and L.Z.; writing—original draft preparation, Y.W.; writing—review and editing, Y.W., J.L., B.Y., D.S. and L.Z.; visualization, Y.W.; supervision, J.L.; project administration, J.L. and B.Y.; funding acquisition, J.L. and B.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This study is funded by the Ministerial Civil Ship Research Project of China (Grant numbers [2024]56).

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding authors.

Acknowledgments: The authors are responsible for the contents of this publication. In addition, the authors would like to thank their lab classmates for their contributions to the writing quality.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Liu, J.; Li, C.; Ouyang, P.; Liu, J.; Wu, C. Interpreting the prediction results of the tree-based gradient boosting models for financial distress prediction with an explainable machine learning approach. *J. Forecast.* **2023**, *42*, 1112–1137. [CrossRef]
2. Seong, N.; Nam, K. Forecasting price movements of global financial indexes using complex quantitative financial networks. *Knowl.-Based Syst.* **2022**, *235*, 107608. [CrossRef]
3. El-Rashidy, N.; El-Sappagh, S.; Abuhmed, T.; Abdelrazek, S.; El-Bakry, H.M. Intensive Care Unit Mortality Prediction: An Improved Patient-Specific Stacking Ensemble Model. *IEEE Access* **2020**, *8*, 133541–133564. [CrossRef]
4. Han, M.; Fan, L. A short-term energy consumption forecasting method for attention mechanisms based on spatio-temporal deep learning. *Comput. Electr. Eng.* **2024**, *114*, 109063. [CrossRef]
5. Yukseltan, E.; Yucekaya, A.; Bilge, A.H. Hourly electricity demand forecasting using Fourier analysis with feedback. *Energy Strategy Rev.* **2020**, *31*, 100524. [CrossRef]

6. Zhao, H.; Zhu, D.; Yang, Y.; Li, Q.; Zhang, E. Study on photovoltaic power forecasting model based on peak sunshine hours and sunshine duration. *Energy Sci. Eng.* **2023**, *11*, 4570–4580. [CrossRef]
7. Chu, W.-T.; Liang, Y.-H.; Ho, K.-C. Visual Weather Property Prediction by Multi-Task Learning and Two-Dimensional RNNs. *Atmosphere* **2021**, *12*, 584. [CrossRef]
8. Sundareswaran, A.; Lavanya, K. Real-Time Vehicle Traffic Prediction in Apache Spark Using Ensemble Learning for Deep Neural Networks. *Int. J. Intell. Inf. Technol.* **2020**, *16*, 19–36. [CrossRef]
9. Kim, Y.S. Comparison of the decision tree, artificial neural network, and linear regression methods based on the number and types of independent variables and sample size. *Expert Syst. Appl.* **2008**, *34*, 1227–1234. [CrossRef]
10. Reilly, D.; Taylor, M.; Fergus, P.; Chalmers, C.; Thompson, S. The Categorical Data Conundrum: Heuristics for Classification Problems—A Case Study on Domestic Fire Injuries. *IEEE Access* **2022**, *10*, 70113–70125. [CrossRef]
11. Hancock, J.T.; Khoshgoftaar, T.M. Survey on categorical data for neural networks. *J. Big Data* **2020**, *7*, 28. [CrossRef]
12. Chen, H.-S.; Lan, T.-S.; Lai, Y.-M. Prediction Model of Working Hoursa of Cooling Turbine of Jet Engine with Back-propagation Neural Network. *Sens. Mater.* **2021**, *33*, 843. [CrossRef]
13. Yu, T.; Cai, H. The Prediction of the Man-Hour in Aircraft Assembly Based on Support Vector Machine Particle Swarm Optimization. *J. Aerosp. Technol. Manag.* **2015**, *7*, 19–30. [CrossRef]
14. Ge, Y.; Nan, Y.; Bai, L. A Hybrid Prediction Model for Solar Radiation Based on Long Short-Term Memory, Empirical Mode Decomposition, and Solar Profiles for Energy Harvesting Wireless Sensor Networks. *Energies* **2019**, *12*, 4762. [CrossRef]
15. Bu, H.; Ge, Z.; Zhu, X.; Yang, T.; Zhou, H. Prediction of Ship Painting Man-Hours Based on Selective Ensemble Learning. *Coatings* **2024**, *14*, 318. [CrossRef]
16. Hur, M.; Lee, S.; Kim, B.; Cho, S.; Lee, D.; Lee, D. A study on the man-hour prediction system for shipbuilding. *J. Intell. Manuf.* **2015**, *26*, 1267–1279. [CrossRef]
17. Golnaraghi, S.; Zangenehmadar, Z.; Moselhi, O.; Alkass, S. Application of Artificial Neural Network(s) in Predicting Formwork Labour Productivity. *Adv. Civ. Eng.* **2019**, *2019*, e5972620. [CrossRef]
18. Wang, L.; Xie, D.; Zhou, L.; Zhang, Z. Application of the hybrid neural network model for energy consumption prediction of office buildings. *J. Build. Eng.* **2023**, *72*, 106503. [CrossRef]
19. Carrizosa, E.; Nogales-Gómez, A.; Romero Morales, D. Clustering categories in support vector machines. *Omega* **2017**, *66*, 28–37. [CrossRef]
20. Gnat, S. Impact of Categorical Variables Encoding on Property Mass Valuation. *Procedia Comput. Sci.* **2021**, *192*, 3542–3550. [CrossRef]
21. Hien, D.T.T.; Thuy, C.T.T.; Anh, T.K.; Son, D.T.; Giap, C.N. Optimize the Combination of Categorical Variable Encoding and Deep Learning Technique for the Problem of Prediction of Vietnamese Student Academic Performance. *Int. J. Adv. Comput. Sci. Appl.* **2020**, *11*, 274–280. [CrossRef]
22. Li, J.; Xu, J.; Zhou, Q. Monitoring serially dependent categorical processes with ordinal information. *IISE Trans.* **2018**, *50*, 596–605. [CrossRef]
23. De Meulemeester, H.; De Moor, B. Unsupervised Embeddings for Categorical Variables. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–8.
24. Dahouda, M.K.; Joe, I. A Deep-Learned Embedding Technique for Categorical Features Encoding. *IEEE Access* **2021**, *9*, 114381–114391. [CrossRef]
25. Lee, N.; Kim, J.-M. Conversion of categorical variables into numerical variables via Bayesian network classifiers for binary classifications. *Comput. Stat. Data Anal.* **2010**, *54*, 1247–1265. [CrossRef]
26. Jung, T.; Kim, J. A new support vector machine for categorical features. *Expert Syst. Appl.* **2023**, *229*, 120449. [CrossRef]
27. Yang, M.; Xu, S. Orthogonal Nonnegative Matrix Factorization using a novel deep Autoencoder Network. *Knowl.-Based Syst.* **2021**, *227*, 107236. [CrossRef]
28. Lecun, Y.; Soulie Fogelman, F. Modeles connexionnistes de l'apprentissage. *Intellectica Spec. Issue Apprentiss. Mach.* **1987**, *2*, 114–143. [CrossRef]
29. Bourlard, H.; Kamp, Y. Auto-association by multilayer perceptrons and singular value decomposition. *Biol. Cybern.* **1988**, *59*, 291–294. [CrossRef]
30. Helwig, N.; Hsiao-Wecksler, E. *Multivariate Gait Data*; UCI Machine Learning Repository: Irvine, CA, USA, 2022. [CrossRef]
31. MrSimple07. *Restaurants Revenue Prediction 2024*; Kaggle: San Francisco, CA, USA, 2024. [CrossRef]
32. Kaggle. Fish Market. Available online: https://www.kaggle.com/datasets/vipullrathod/fish-market (accessed on 12 July 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.