

Article

An Accuracy Prediction Method of the RV Reducer to Be Assembled Considering Dendritic Weighting Function

Shousong Jin , Yanxi Chen, Yiping Shao  and Yaliang Wang *

School of Mechanical Engineering, Zhejiang University of Technology, Hangzhou 310023, China

* Correspondence: wangyaliang@zjut.edu.cn

Abstract: There are many factors affecting the assembly quality of rotate vector reducer, and the assembly quality is unstable. Matching is an assembly method that can obtain high-precision products or avoid a large number of secondary rejects. Selecting suitable parts to assemble together can improve the transmission accuracy of the reducer. In the actual assembly of the reducer, the success rate of one-time selection of parts is low, and “trial and error assembly” will lead to a waste of labor, time cost, and errors accumulation. In view of this situation, a dendritic neural network prediction model based on mass production and practical engineering applications has been established. The size parameters of the parts that affected transmission error of the reducer were selected as influencing factors for input. The key performance index of reducer was transmission error as output index. After data standardization preprocessing, a quality prediction model was established to predict the transmission error. The experimental results show that the dendritic neural network model can realize the regression prediction of reducer mass and has good prediction accuracy and generalization capability. The proposed method can provide help for the selection of parts in the assembly process of the RV reducer.



Citation: Jin, S.; Chen, Y.; Shao, Y.; Wang, Y. An Accuracy Prediction Method of the RV Reducer to Be Assembled Considering Dendritic Weighting Function. *Energies* **2022**, *15*, 7069. <https://doi.org/10.3390/en15197069>

Academic Editors: Xi Gu, Tangbin Xia, Ershun Pan, Rongxi Wang and Yupeng Li

Received: 31 August 2022

Accepted: 21 September 2022

Published: 26 September 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: RV reducer; assembly quality; dendrites; neural network; transmission accuracy

1. Introduction

Rotate vector (RV) reducer has the advantages of small size, compact structure and large transmission ratio [1]. The quality of its assembly determines the performance, production cost, and production efficiency of the product. It is mainly used in robot joints with high precision and large load. At present, foreign countries already have a relatively complete theoretical system for RV reducers. Domestic RV reducers have been developed for many years without major breakthroughs in accuracy. The dynamic transmission error of RV reducers depends on the manufacturing error of each component, assembly errors, and elastic deformation. However, the material properties of parts are easy to determine and the manufacturing accuracy is difficult to ensure. In this case, companies generally measure parts and assemble parts to improve the dynamic transmission accuracy of the reducer.

In 2007, Kannan SM et al. used particle swarm algorithm to obtain the best combination of parts [2] and successfully made the assembly deviation less than the sum of the tolerances of the parts. Gentilini et al. established a finite element model; this method can predict and show the final shape of the assembly [3]. There is no need for physical assembly in future practice, reducing the time and cost of product quality inspection. S.Khodaygan et al. proposed to estimate the tolerance assembly and the reliability of the mechanical assembly to meet the quality requirements through the Bayesian modeling [4], which can formulate accurate assembly functions for complex mechanical assemblies. At present, artificial neural networks are widely used in the field of speech recognition, computer vision, and bioinformatics, etc. In recent years, some scholars have used neural networks to develop assembly models, which avoids the complicated operation and heavy workload of traditional methods to solve the accuracy. Steinberg Fabian [5] utilized a gradient boosting

classifier to identify assembly start delayers. Stark Rainerr et al. selected three approaches based upon regression, natural language processing, and clustering into intelligent services in the digital production process to reduce the time for resolving upcoming issues in assembly [6]. It has certain guiding significance for improving the assembly quality.

The current neuron model relies on the McCulloch–Pitts structure. This neuron uses the weight between the synapses, and then obtains the output through the activation function. It does not consider the local interaction played by the dendritic structure as a part of the neuron and is only used to transmit signals. Experiments had shown that dendrites play an important role in information processing in biological neurons, which can nonlinearly integrate postsynaptic signals and filter out irrelevant background information [7–9]. The dendrites receive signals (action potentials) from upstream neurons. After the information is integrated and processed, it is transmitted to the neuron soma. The powerful information processing capabilities of dendrites enable individual neurons to process thousands of different synaptic inputs unequally [10]. Inspired by biological phenomena, Yuki Todo et al. proposed a single dendritic neuron with synaptic, dendritic, membranous, and somatic layers [11]. A synapse performs a sigmoid non-linear operation on its input, the nodes in the synaptic layer contain the initial weights and thresholds of the dendritic neural network, each branch receives a signal and performs a multiplication operation at its synapse, the membrane sums the results of all branches, ultimately transmitting the signal to the cell body. When the threshold is exceeded, cells send signals to other neurons through the axon. At present, this model has been widely used, such as computer-aided medical diagnosis [12] and morphological hardware implementation [13]. Gang et al. believed that the dendritic neuron model proposed by Yuki Todo and Gao [14] is not enough to be expressive, can only express first-order input, and is not conducive to computer operation [15]. Considering that the definition of multivalued logic and the integration of potentials in biological neurons can be described in a multiplicative form, a dendrite (DD) containing only matrix multiplication and Hadamard product was proposed to simulate the function of data interaction processing. Gang [16] developed a Taylor series using the dendritic network proposed by him and constructed a relational spectrum model to analyze the synergy and coupling of hand muscles, which is helpful for the design of prosthetic hands. In this article, the DD module is used to develop a prediction model to capture the internal structure dependence and increase the interaction between information, which provides a new idea for the prediction of reducer assembly accuracy.

2. Analysis of Quality Influencing Factors

Transmission error (TE) is an important index to evaluate the gear meshing mass, which is directly related to working accuracy, reliability, vibration noise, and service life of gear transmission [17]. In the actual manufacturing and installation process of the reducer, transmission errors are inevitable and the main sources include their own processing errors and assembly errors. According to the different specifications of the RV reducer, the transmission error is strictly limited within (1~1.5').

The RV reducer has a sophisticated and complex structure, and its components mainly include input shaft, planetary gear, planet carrier, flange, crankshaft, pin teeth, etc. Figure 1 is a schematic diagram of the RV-40E reducer. Figure 1a shows the component assembly structure of the reducer. The transmission of the RV reducer is divided into two stages, which are the first-stage involute planetary gear transmission and the second-stage cycloidal pinwheel transmission [18,19]. The movement of the input shaft drives the planetary gear to achieve first-level deceleration. The planetary gear drives the crankshaft, then drives the swivel arm bearing, and transmits the power to the cycloidal pinwheels. Under the combined action of the swivel arm bearing and the pin teeth, the cycloid pinwheel produces a revolution motion that rotates around the central circular axis of the pin tooth and an autorotation motion that rotates around its own axis. The autorotation drives the flange and output planet carrier to achieve 1:1 speed ratio output to complete the second-stage of deceleration. Since the influence of the cycloidal pinwheel transmission on the

transmission error is directly reflected on the output shaft, the first-stage planetary gear reduction mechanism is far from the output, and its transmission ratio is only the reciprocal of the second-stage cycloidal pinwheel transmission ratio. Therefore, the RV reducer transmission accuracy mainly depends on the second transmission. Figure 1b shows the schematic diagram of the transmission principle mechanism of the reducer [20,21]. The key parts involved in cycloidal pinwheel transmission include: cycloidal wheel, pinwheel, crankshaft, and crank bearing. However, in actual engineering assembly, the crank bearing clearance is usually adjusted to a small value, and the influence on the transmission error can be ignored. The influence of error factors of remaining key components on transmission error were analyzed, and the influence degree of each factor on the output shaft error were obtained. The parameters of each influencing factor $\theta = [\theta_1, \theta_2, \dots, \theta_n]$ have errors $\Delta\theta = [\Delta\theta_1, \Delta\theta_2, \dots, \Delta\theta_n]$. The transmission error of RV reducer can be recorded as $\varphi(\theta) = \varphi(\theta_1, \theta_2, \dots, \theta_n)$, and each error sensitivity index can be defined as [22,23]:

$$S_N = \nabla \varphi(\theta) = \left[\frac{\partial \varphi}{\partial \theta_1}, \frac{\partial \varphi}{\partial \theta_2}, \dots, \frac{\partial \varphi}{\partial \theta_n} \right] = [S_1, S_2, \dots, S_n] \quad (1)$$

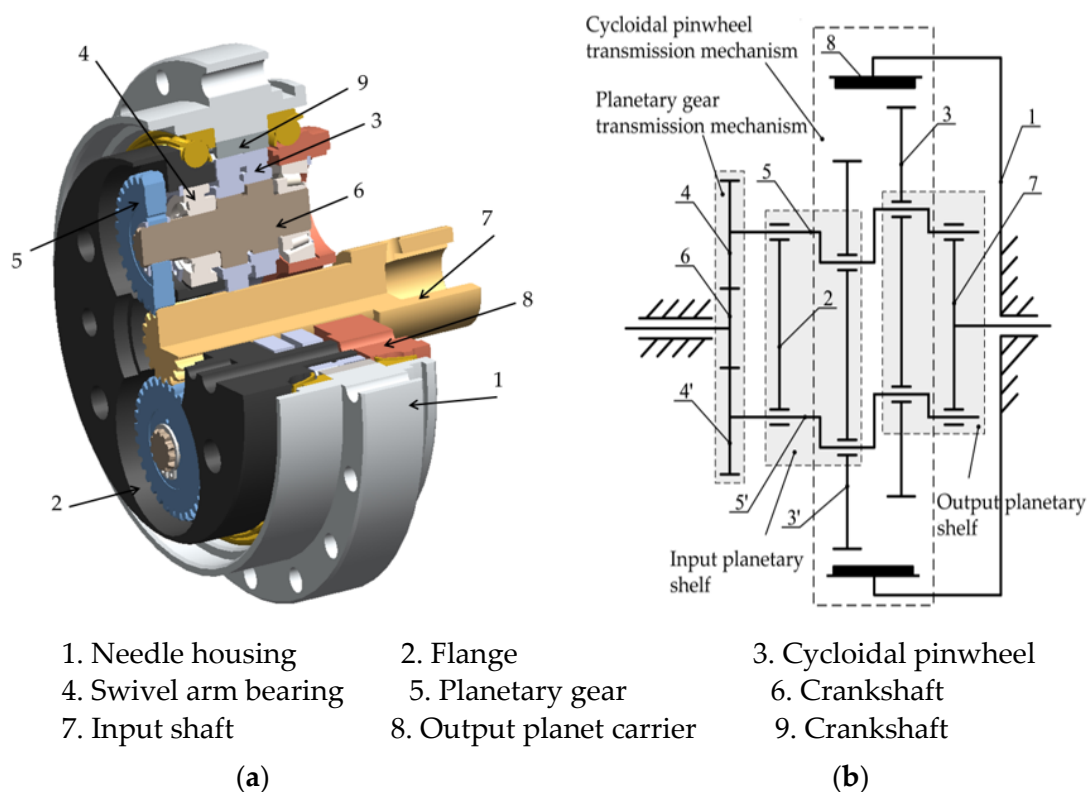


Figure 1. Schematic diagram of RV-40E reducer structure: (a) structure diagram; and (b) schematic diagram.

This article takes the RV-40E type reducer as an example. Taking the center circle radius error of the needle tooth as reference error. The sensitivity index of the needle tooth center circle radius error is 1, the sensitivity index of other error parameters is compared with the reference error parameter. Through the calculation of sensitivity. It can be concluded that the factors that have a larger sensitivity index to the transmission error are: needle tooth center circle radius error δr_p , needle tooth pin radius error δr_{rp} , needle tooth pin hole circumferential position error δt_{Σ} , equidistant modification error $\delta \Delta r_p$, shift distance modification error $\delta \Delta r_{rp}$.

3. Development a Network Model

In the back propagation feed-forward neural network, each neuron takes the output of the node in the upper layer as the input. The input is activated by linear weighting and nonlinear function to obtain the output of this node, then pass the output to the node in the next layer. When using McCulloch–Pitts neurons to learn the relationship between input space and output space, it ignores the function of dendritic structure as a part of neurons to process information. For this phenomenon, this article uses the logic extractor DD proposed by gang to realize data interaction processing function and constructs a dendrite neural network (DDNN) model for the reducer transmission error prediction. In this model, the dendrites first to perform interactive preprocessing on the signals transmitted from upstream neurons, and then transmit them to the cell body for linear weighting. Finally, the axons are activated by nonlinear functions, then the signals are transmitted to downstream neurons. Compared with McCulloch–Pitts neuron, the neuron model constructed by DD module has lower computational complexity, stronger network generalization, and fitting capability.

3.1. Back Propagation Feed-Forward Neural Network

In the artificial neuron, $X = \{x_1, x_2, \dots, x_n\}$ is feature input, n is the number of input features, x_i is the i th influencing factor. The input value of the neuron is transmitted by the connection of the weighting coefficient. The positive and negative values of the weighting coefficient simulate the excitation and inhibition of the synapse. The size indicates the strength of the connection, $\sum_{i=1}^n w_i x_i$ is the integration of all input signals by the cell body. The threshold b controls the activation of neurons. When the sum of the inputs exceeds the threshold, the neuron is activated and an output signal is generated to transmit information. f represents nonlinear activation function, the output $\hat{y}$ can be expressed as:

$$\hat{y} = f(W^{2,1}X + b) \quad (2)$$

where $W^{2,1}$ is the weight matrix from the first layer to the second layer, b is the bias.

Back propagation feed-forward (bp) neural network is a multi-layer feed-forward network trained according to the error back propagation algorithm, which is widely used at present. The network usually composed of three layers: input layer, hidden layer, and output layer. Its learning rule is gradient descent, which gradually adjusts weights and thresholds between layers through back propagation error signals to minimize the loss function of the network.

Firstly, define E as the loss function of the latter layer in back propagation and a as the learning rate. The weights from the input layer to the hidden layer is represented by v_j , and from the hidden layer to the output layer is represented by w_j . ins_p is the input of the p th neuron in the output layer, and ins_q is the input of the q th neuron in the hidden layer. The output vector of the output layer is o , the expected output vector is e , l represents the l th layer of the network model. The output error is:

$$E = \frac{1}{2} \sum_{p=1}^l (e_p - o_p)^2 = \frac{1}{2} \sum_{p=1}^l \left\{ e_p - f \left[\sum_{q=0}^m w_{qp} f \left(\sum_{i=0}^n v_{iq} x_n \right) \right] \right\}^2 \quad (3)$$

It can be seen from the above formula that the size of the output error of the network is related to the weights v_j and w_j of each layer. The size $W = \{v_j, w_j\}$ of the output error E can be changed by adjusting the weights. The weight matrix update can be simplified as:

$$W^{2,1(\text{new})} = W^{2,1(\text{old})} - a \frac{\partial E}{\partial W^{2,1(\text{old})}} \quad (4)$$

3.2. Dynamic Adam Optimization Algorithm

For machine learning problems in high-dimensional parameter spaces or large datasets, Jimmy Ba et al. proposed a dynamic Adam optimization algorithm based on gradient optimization of the objective function with variable learning rate during iterations [24]. It continues the strengths of Adagrad and RMSprop. Independent adaptive learning rates are set for different parameters through the first and second moments of the gradient, which makes the algorithm converge faster. It improves problems, such as objective function fluctuations, and maintains prediction performance in non-dense gradient problems and unstable data. The first-order and second-order moment deviations are calculated as follows:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) dW^{l,l-1} \quad (5)$$

$$v_t = \beta_2 m_{t-1} + (1 - \beta_2) (dW^{l,l-1})^2 \quad (6)$$

$$\begin{cases} \hat{m} = \frac{m_t}{1 - \beta_1^t} \\ \hat{v} = \frac{v_t}{1 - \beta_2^t} \end{cases} \quad (7)$$

where t is the number of iterations, m_t is the first-order moment vector, n_t is the second-order moment vector. $\hat{m}$ is the first-order moment deviation, $\hat{v}$ is the second-order moment deviation, β_1 is the first-order moment attenuation coefficient, β_2 is the second-order moment attenuation coefficient. This article takes 0.9 and 0.999, respectively. $W^{l,l-1}$ is the weight matrix from the $(l - 1)$ th layer to the l th layer in the DD module. The weight matrix using the Adam optimizer can be updated as:

$$W^{2,1(\text{new})} = W^{2,1(\text{old})} - a^{2,1} \frac{\hat{m}}{\sqrt{\hat{v}} + \varepsilon} \quad (8)$$

where $\varepsilon = 10^{-8}$ prevents the divisor from becoming 0.

3.3. Dendritic Unit

In biological nervous systems, dendrites had been shown to have logical operations [25]. The dendritic used in this article is the 0 product between the current input and the previous input. The Hadamard product can be used to establish the logical relationship between inputs, so DD is the expression of the logical relationship between features. DD can capture the logical relationship between features. The DD module can not only be used to extract the local relationship between inputs, but they also use the internal correlation information to strengthen the connection between the two features and improve the network accuracy while capturing the internal structural dependencies. The DD module is shown in Figure 2.

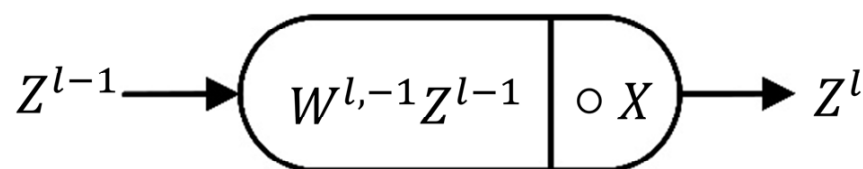


Figure 2. Dendrite module.

The expression of the dendrite module is as follows:

$$Z^l = W^{l,l-1} Z^{l-1} \circ X \quad (9)$$

where Z^{l-1} is the input of the module, Z^l is the output of the module. $W^{l,l-1}$ is the weight matrix from the $(l - 1)$ th layer to the l th layer. $X = \{x_1, x_2, \dots, x_n\}$ represents the original input. “ $\circ$ ” represents the Hadamard product, which represents the multiplication of corresponding elements. It is used to establish the logical relationship between inputs.

So, it produces another matrix with the same dimensions as the original product matrix. The features are deeply fused by multiplying corresponding elements, which can better reflect the meaning of feature intersection.

The DD module is essentially an information processing method, similar to a variant of the self-attention mechanism. It pays attention to the input feature variable itself, increases the interaction between information, and strengthens the connection between features. By training its own information to update the parameters, it gives the network stronger information processing capability and better generalization capability.

3.4. Developing a Dendritic Neural Network Model

The artificial neuron structure ignores the capability of dendrites to process information interaction and only uses dendrites to transmit information. However, in actual biological information, each neuron can have one or more dendrites. In this article, multiple DD modules are used to simulate dendritic functions to form a single dendritic neuron [26]. The dendrite extracts logical information from the data transmitted by the upstream neurons. Then, the information is interactively preprocessed and passed to the cell body for linear weighting. The signal is nonlinearly activated and, finally, output by the axon. Figure 3 shows the structure of neurons. As shown in Figure 3a, the biological neurons have dendrites, cell bodies, axons, and other organizations. Figure 3b shows the filter, accumulator, and balancer in dendritic neurons to simulate the organizational function of biological neurons, and the expression is as follows:

$$\begin{cases} Z^1 = W^{1,0} X \circ X \\ \hat{y} = f(W^{2,1} Z^1 + b) \end{cases} \quad (10)$$

where $X = \{x_1, x_2, \dots, x_n\}$ is the feature input, n is the number of influencing factors, $W^{1,0}$ and $W^{2,1}$ represent the weight matrix from layer 0 to layer 1 and the weight matrix from layer 1 to layer 2, respectively. b is bias, f is nonlinear activation function, where ReLU is selected as the activation function.

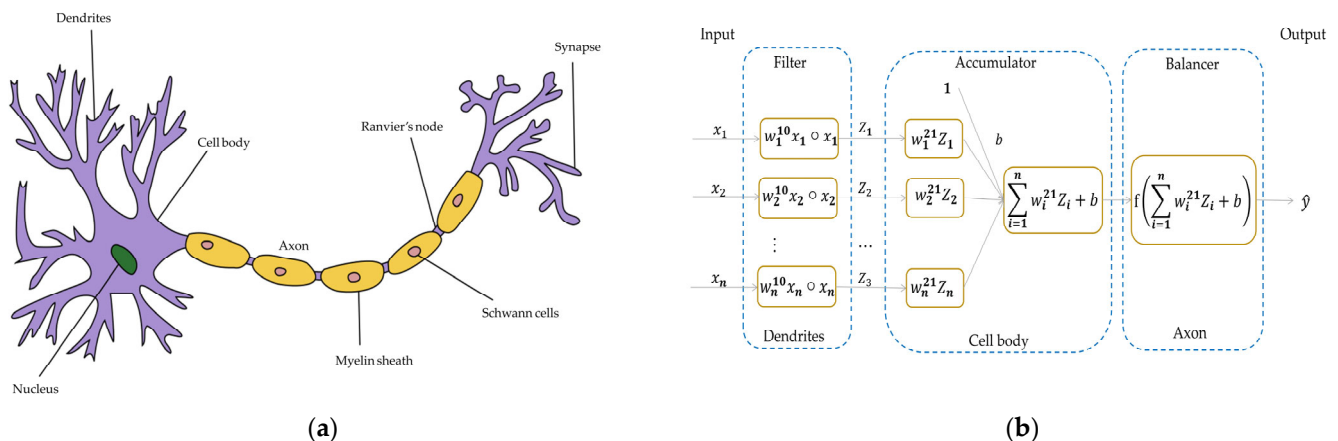


Figure 3. The structure of neurons: (a) biological neuron structure; and (b) functional structure diagram of dendritic neuron model.

According to the source of the transmission error, five influencing factors are divided into two dimensions. The information of different dimensions was represented by different neurons. Figure 4 shows the fusion of information from different neurons, and the transmission error prediction model was constructed.

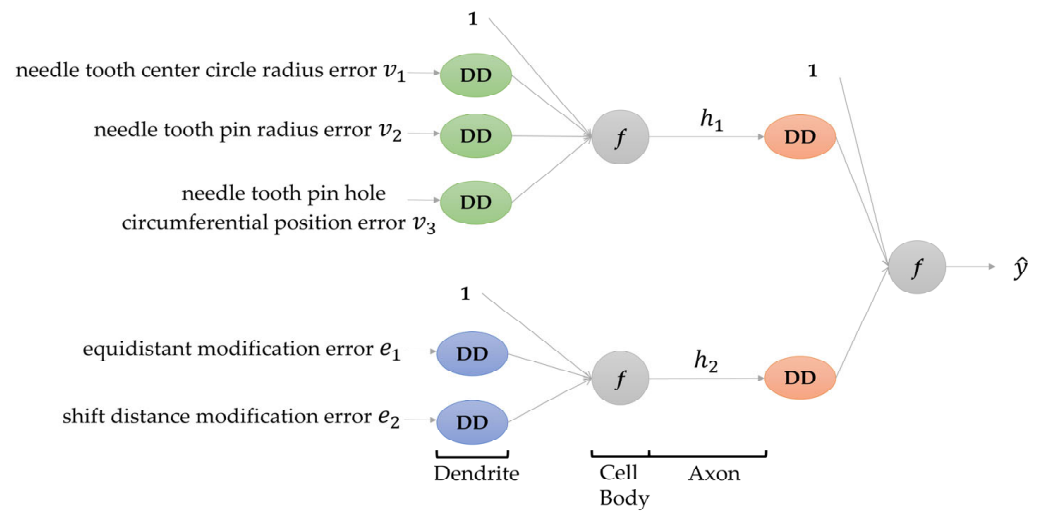


Figure 4. Dendritic neural network (DDNN) model.

The architecture of DDNN can be expressed as follows:

$$\begin{cases} Z_v^1 = W_v^{1,0} X \circ X \\ h_1 = f(W_v^{2,1} Z_v^1 + b) \end{cases} \quad (11)$$

$$\begin{cases} Z_e^1 = W_e^{1,0} X \circ X \\ h_2 = f(W_e^{2,1} Z_e^1 + b) \end{cases} \quad (12)$$

$$\begin{cases} A_h = W_h^{1,0} H \circ H \\ \hat{y} = f(W_h^{2,1} A_h + b) \end{cases} \quad (13)$$

where $V = \{v_1, v_2, v_3\}$, v_1 represents the radius error of needle tooth center circle δr_p , v_2 represents the needle tooth pin radius error δr_{rp} , v_3 represents the needle tooth pin hole circumferential position error δt_{Σ} , all from the component pin wheel. $E = \{e_1, e_2\}$, e_1 represents the equidistant modification error $\delta \Delta r_p$, e_2 represents the shift distance modification error $\delta \Delta r_{rp}$, all from components cycloid wheel. $H = \{h_1, h_2\}$. Feature vectors are combined into tensor, which is fed into DDNN model for training. The information is interactively preprocessed in the DD module, and then transmitted to the cell body for linear weighting. Finally, the axon performs nonlinear function activation and transmits information h_1 and h_2 to the next layer of DD module. Each DD module selects input units from upper layer without repeating to connect and repeats the above learning rules. Finally, axon output the predicted value. The overall process of DDNN model prediction is shown in Figure 5. Firstly, the sample data set is divided into training set and test set. Secondly, the factors affecting the quality of reducer in the training set and test set are denoted as eigenvalues, and the transmission errors are denoted as labels. The training set is imported into DDNN model for training. Equations (11)–(13) aim to obtain the minimum value of the loss function. They use the dynamic Adam optimization algorithm and the error signal back propagation algorithm to update threshold and weight parameters. When the convergence of the loss function of model training reaches the expectation, the training is stopped and the model is saved. Finally, the test set is brought into the trained model to obtain prediction results.

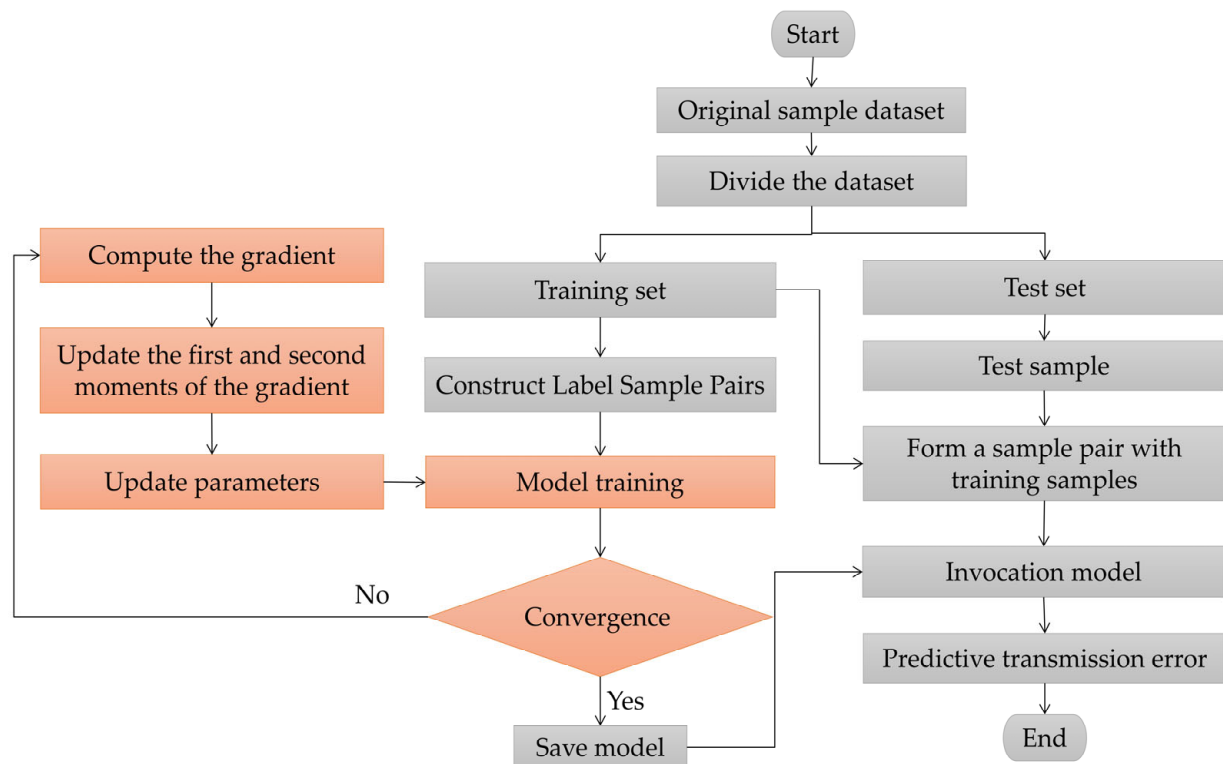


Figure 5. Prediction flow chart of the DDNN model.

4. The Model Solution

4.1. Preprocessing of Test Data

This article takes the RV-40E-121 reducer as an example to analyze. Five influencing factors related to transmission error during assembly were selected: needle tooth center circle radius error δr_p , needle tooth pin radius error δr_{rp} , needle tooth pin hole circumferential position error δt_{Σ} , equidistant modification error $\delta \Delta r_p$, shift distance modification error $\delta \Delta r_{rp}$. The design upper and lower limits of the five influencing factors are shown in Table 1. Dimensions of parts are in millimeters.

Table 1. Design upper and lower limits of parameter error of main parts (mm).

Parameter	δr_p	δt_{Σ}	δr_p	$\delta \Delta r_p$	$\delta \Delta r_{rp}$
Upper limits	0.03	0.001	0.015	0.078	0.048
Lower limits	0.01	−0.003	0.005	0.026	0.016

In order to avoid the large difference in the value range of each feature affecting the efficiency of the gradient descent method. The data set needs to be preprocessed before model training. First, removing the data that deviate too much, then using the Max–Min normalization method to linearly transform the sample eigenvalues. Thus, making the result mapped in the interval [0, 1], and the scaling function is as follows:

$$x_{\text{new}} = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (14)$$

where x is the original data, $\max(x)$ is the maximum, and $\min(x)$ is the minimum values in the data. x_{new} is the standardized data.

Through the error test platform, the size parameters of the above-mentioned main components were collected as characteristic inputs of the influencing factors, and the transmission error of the reducer was used as the output index. A data set was constructed

with a total of 300 samples information. Some sample point data are shown in Table 2, where the unit of transmission error is angular minutes.

Table 2. Some sample data example (mm).

Serial Number	δr_p	δt_Σ	δr_p	$\delta \Delta r_p$	$\delta \Delta r_p$	Transmission Error/"
1	0.025	0.001	0.015	0.038	0.036	1.880
2	0.017	−0.001	0.023	0.040	0.041	2.151
3	0.015	0.001	0.020	0.032	0.036	1.193
4	0.020	0	0.020	0.038	0.068	1.419
5	0.010	−0.001	0.010	0.016	0.026	0.786
6	0.025	−0.002	0.020	0.026	0.052	1.762
7	0.020	−0.003	0.020	0.040	0.063	2.001
8	0.023	0.001	0.017	0.040	0.041	2.508
9	0.027	−0.002	0.027	0.032	0.052	1.952
10	0.020	0.001	0.013	0.032	0.031	1.166

Firstly, 300 groups of samples were randomly shuffled, and then the data set was divided. A total of 90% of the samples were set as the training set, and 10% of the samples were set as the test set.

4.2. Parameter Selection

The framework used in the experiment in this chapter is TensorFlow2.0, and the test was implemented through Python3.6. The specific hardware environment is: the CPU is Intel i7, the GPU model is NVIDIA 2080Ti, the CUDA version is 11.2.1, and the CUDNN version is 8.1.1.33.

The loss function can measure the difference between the output value of the model and the true value. It is a measure to evaluate the fitting capability of the model. The advantage of using Log-Cosh function as loss function is that when the feature errors between samples is small, the Log-Cosh function converges faster. when the feature errors between samples is large, the Log-Cosh function is not susceptible by outliers. Compared with other functions, the characteristic curve is smoother and can be derivable twice. To a certain extent, the robustness of the model can be improved [27]. The expression of the Log-Cosh function is as follows:

$$\text{Loss}(y_i, \hat{y}_i) = \sum_{i=1}^m \log(\cosh(\hat{y}_i - y_i)) \quad (15)$$

where y_i is the label vector of the i th training sample, $\hat{y}_i$ represents the predicted value of the i th output sample, m represents the total number of training samples. The relevant training parameter conditions: the loss function was the Log-Cosh function, the batch-size was 128, the training epoch was 200, the initial learning rate was 0.001, the optimizer was Adam. After iterative training according to the settings of the above training parameters, the loss function convergence curve of DDNN model, as shown in Figure 6, was obtained.

When the convergence of the loss function of model training reached the expectation, the training was stopped and the model was saved; the weight parameters were extracted from the saved model. Table 3 shows the weight information of each layer of the model parameters of the last training (keep values to three decimal places); here, symbol “—” means no weight value. Finally, the test set was brought into the trained model to get the prediction result and compared with the label.

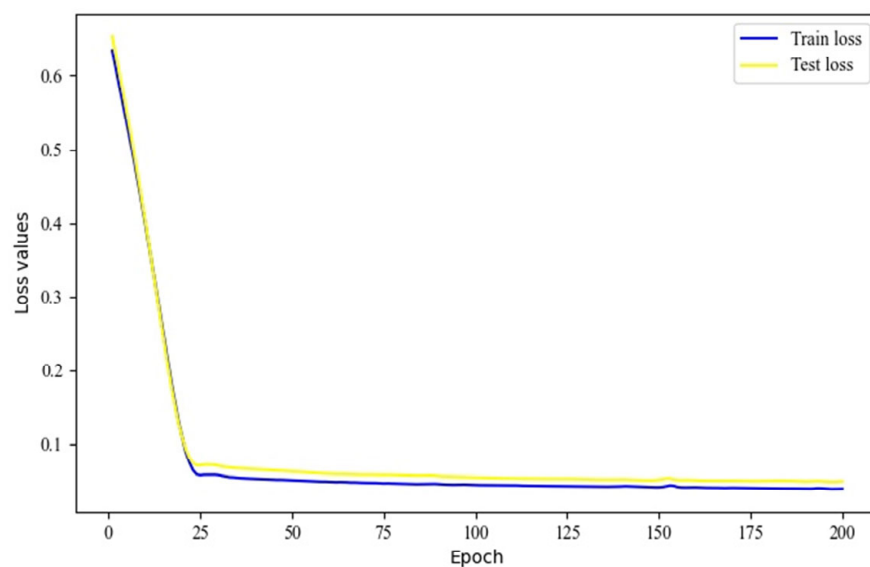


Figure 6. Convergence of loss function of reducer transmission accuracy error.

Table 3. Optimal weights.

The Weight of Each Layer		Weight Value							
The first layer	0.183	−0.070	−0.745	0.478	−0.765	The second layer	0.826	0.465	0.205
	−0.601	0.091	−0.313	−0.006	−0.498		0.656	−0.973	—
	0.146	−0.325	−0.538	−0.181	−0.119		—	—	—
	0.484	1.127	−0.681	−0.747	0.447		—	—	—
	−0.412	0.665	0.513	0.280	0.117		—	—	—
The third layer	0.987	0.756	—	—	—	The fourth layer	1.029	−0.725	—
	−1.273	0.987	—	—	—		—	—	—

The prediction curve of the DDNN model is shown in Figure 7, where the blue line represents real sample values, the red line represents the predicted values.

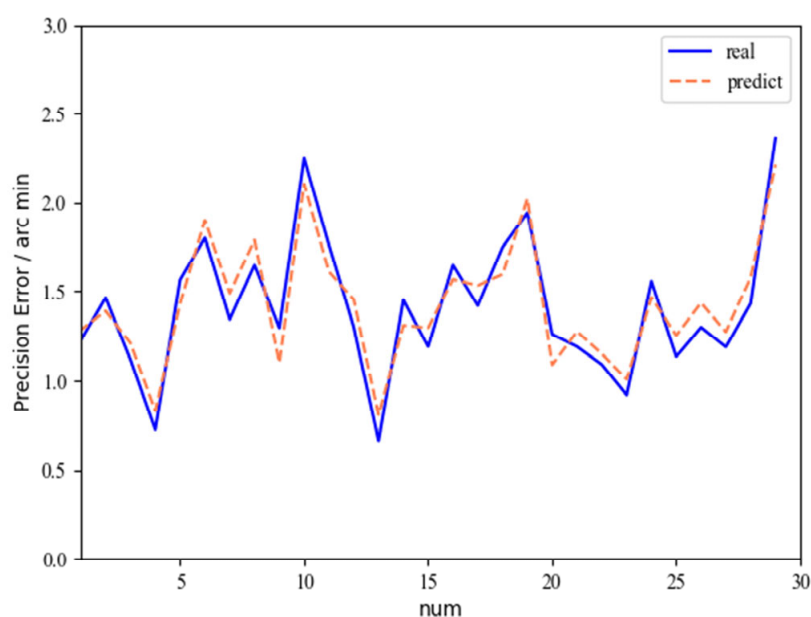


Figure 7. Prediction result of transmission error sample of RV-40E-121 reducer.

4.3. Predictive Performance Analysis

In order to further quantify the prediction accuracy of DDNN model, the mean absolute error and mean square error were introduced as performance evaluation indicators. The effectiveness of the DDNN prediction method was verified by analyzing the performance of model predictions. The specific definitions are as follows:

MSE (Mean squared error)

$$\text{MSE} = \frac{1}{m} \sum_{i=1}^m (\hat{y}_i - y_i)^2 \quad (16)$$

MAE (Mean absolute error)

$$\text{MAE} = \frac{1}{m} \sum_{i=1}^m |\hat{y}_i - y_i| \quad (17)$$

where y_i is the label vector of the i th training sample, $\hat{y}_i$ represents the predicted value of the i th output sample, and m represents the total number of training samples.

MSE reflects the mean squared prediction error of the model. MAE describes the average value of the absolute error between the predicted value and the observed value [28]. The smaller the value of MSE and MAE, the better. The method proposed in this paper was compared with the other three commonly used prediction models, which are BP neural network, support vector regression with Gaussian kernel (SVR-R) and general regression neural network (GRNN) [29]. The initial parameter settings for all models are shown in Table 4.

Table 4. The initial parameter values of models.

Model	Parameter	Value
BP neural network	Learning rate	0.001
	Optimizer	Stochastic Gradient Descent
GRNN	Smooth factor	0.03
SVR-R	Penalty term	40
	Gamma	4
Dendritic Neural Network	Learning rate	0.001
	Optimizer	Adam

Table 5 shows the comparison results of the prediction performance and computational efficiency of the above four models. From the error results in the table, it can be concluded that for transmission error prediction, the prediction accuracy of the DDNN model is better than the other three models. For computational efficiency, the running time of GRNN and SVR-R outperforms the remaining two models. Considering the prediction performance and calculation efficiency comprehensively, the DDNN model has the highest prediction accuracy and can also meet the time requirements of actual assembly.

Table 5. Comparison of the DDNN with other models.

Model	Mean Squared Error	Mean Absolute Error	Computational Efficiency/s
BP neural network	0.1411	0.2545	13.708
GRNN	0.051	0.229	0.057
SVR-R	0.048	0.216	0.036
Dendritic Neural Network	0.032	0.121	9.751

5. Conclusions and Future Work

In the actual production and assembly process of reducer, the success rate of one-time selection of parts is low, and repeated disassembly and assembly will lead to the generation and accumulation of errors. Through the analysis of the factors affecting the quality of RV reducer, five factors with larger sensitivity indices were selected as sample eigenvalues in the data set, and transmission error was noted as sample label. The prediction results show

that DDNN model can capture the logical relationship between features and strengthen the internal information correlation; it can effectively avoid the failure of the loss function to converge due to large fluctuations of parameters when updating; it has better generalization ability and can effectively predict reducer dynamic transmission error. The research in this article has practical guiding significance for the selection of parts and components in the RV reducer assembly, which can improve the assembly qualification rate, avoid repeated disassembly and assembly, and reduce the waste of labor and time costs.

However, there is still room for improvement in this study, such as the deficiencies in selection of quality influencing factors, and the applicability of the method for different examples. In the follow-up, the influence of quality factors on transmission error of reducer will be comprehensively analyzed to obtain a more ideal DDNN prediction model.

Author Contributions: Conceptualization; Methodology; Project administration, S.J.; Writing; Software and Validation, Y.C.; Review, Y.S.; Supervision; Editing, Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by National High-tech R&D Program of China (Grant No.2015AA043002), Natural Science Foundation of Zhejiang Province (Grant No. LQ22E050017), Post-doctoral Science Foundation of China (Grant No. 2021M702894), and Zhejiang Provincial Postdoctoral Science Foundation (Grant No. ZJ2021119).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to corporate data privacy.

Acknowledgments: This work is supported by Zhejiang Shuanghuan Transmission Machinery Co., Ltd. providing the RV reducer sample and some data.

Conflicts of Interest: The authors declared no potential conflict of interest with respect to the research, authorship, and/or publication of this article.

References

1. Xue, J.; Qiu, Z.R.; Fang, L.; Lu, Y.H.; Hu, W.C. Angular Measurement of High Precision Reducer for Industrial Robot. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–10. [\[CrossRef\]](#)
2. Kumar, M.S.; Kannan, S. Optimum manufacturing tolerance to selective assembly technique for different assembly specifications by using genetic algorithm. *Int. J. Adv. Manuf. Technol.* **2007**, *32*, 5–6. [\[CrossRef\]](#)
3. Gentilini, I.; Shimada, K. Predicting and evaluating the post-assembly shape of thin-walled components via 3D laser digitization and FEA simulation of the assembly process. *Comput.-Aided Des.* **2010**, *43*, 316–328. [\[CrossRef\]](#)
4. Khodaygan, S.; Ghaderi, A. Tolerance–reliability analysis of mechanical assemblies for quality control based on Bayesian modeling. *Assem. Autom.* **2019**, *39*, 769–782. [\[CrossRef\]](#)
5. Burggraef, P.; Wagner, J.; Heinbach, B.; Steinberg, F. Machine Learning-Based Prediction of Missing Components for Assembly—A Case Study at an Engineer-to-Order Manufacturer. *IEEE Access* **2021**, *9*, 105926–105938. [\[CrossRef\]](#)
6. Bruennhaeusser, J.; Gogineni, S.; Nickel, J.; Witte, H.; Stark, R. Assembly issue resolution system using machine learning in Aero engine manufacturing. In Proceedings of the International Conference on Advances in Production Management Systems, Novi Sad, Serbia, 30 August–30 September 2020; Volume 591, pp. 149–157.
7. Takahashi, N.; Oertner, T.G.; Hegemann, P.; Larkum, M.E. Active cortical dendrites modulate perception. *Science* **2016**, *354*, 1587–1590. [\[CrossRef\]](#)
8. Trenholm, S.; McLaughlin, A.J.; Schwab, D.J.; Turner, M.H.; Smith, R.G.; Rieke, F.; Awatramani, G.B. Nonlinear dendritic integration of electrical and chemical synaptic inputs drives fine-scale correlations. *Nat. Neurosci.* **2014**, *17*, 1759–1766. [\[CrossRef\]](#)
9. Li, X.Y.; Tang, J.S.; Zhang, Q.T.; Gao, B.; Yang, J.J.; Song, S.; Wu, W.; Zhang, W.Q.; Yao, P.; Deng, N.; et al. Power-efficient neural network with artificial dendrites. *Nat. Nanotechnol.* **2020**, *15*, 776–782. [\[CrossRef\]](#)
10. Antic, S.D.; Zhou, W.L.; Moore, A.R.; Short, S.M.; Ikonou, K.D. The decade of the dendritic NMDA spike. *J. Neurosci. Res.* **2010**, *88*, 2991–3001. [\[CrossRef\]](#)
11. Todo, Y.; Tamura, H.; Yamashita, K.; Tang, Z. Unsupervised learnable neuron model with nonlinear interaction on dendrites. *Neural Netw.* **2014**, *60*, 96–103. [\[CrossRef\]](#)
12. Tang, C.; Ji, J.; Tang, Y.; Gao, S.; Tang, Z.; Todo, Y. A novel machine learning technique for computer-aided diagnosis. *Eng. Appl. Artif. Intell.* **2020**, *92*, 103627. [\[CrossRef\]](#)

13. Song, S.; Chen, X.; Tang, C.; Song, S.; Tang, Z.; Todo, Y. Training an Approximate Logic Dendritic Neuron Model Using Social Learning Particle Swarm Optimization Algorithm. *IEEE Access* **2019**, *7*, 141947–141959. [[CrossRef](#)]
14. Gao, S.C.; Zhou, M.C.; Wang, Y.R.; Cheng, J.J.; Yachi, H.; Wang, J.H. Dendritic Neuron Model with Effective Learning Algorithms for Classification, Approximation, and Prediction. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 601–614. [[CrossRef](#)] [[PubMed](#)]
15. Liu, G.; Wang, J. Dendrite Net: A White-Box Module for Classification, Regression, and System Identification. *IEEE Trans. Cybern.* **2021**, *in press*. [[CrossRef](#)]
16. Liu, G.; Wang, J. A relation spectrum inheriting Taylor series: Muscle synergy and coupling for hand. *Front. Inform. Technol. Elect. Eng.* **2022**, *23*, 145–157. [[CrossRef](#)]
17. Xu, A.J.; Deng, X.Z.; Zhang, J.; Xu, K.; Li, J.B. A New Numerical Algorithm for Transmission Error Measurement at Gears Meshing. *Adv. Mater. Res.* **2012**, *472*, 1563–1567. [[CrossRef](#)]
18. García-Ayuso, L.E.; Velasco, J.; Dobarganes, M.C.; de Castro, M.D.L. Accelerated extraction of the fat content in cheese using a focused microwave-assisted soxhlet device. *J. Agric. Food Chem.* **1999**, *47*, 2308–2315. [[CrossRef](#)]
19. Zhang, Z.H.; Wang, J.X.; Zhou, G.W.; Pei, X. Analysis of mixed lubrication of RV reducer turning arm roller bearing. *Ind. Lubr. Tribol.* **2018**, *70*, 161–171. [[CrossRef](#)]
20. Bahubalendruni, M.V.A.R.; Gulivindala, A.; Kumar, M.; Biswal, B.B.; Annepu, L.N. A hybrid conjugated method for assembly sequence generation and explode view generation. *Assem. Autom.* **2019**, *39*, 211–225. [[CrossRef](#)]
21. Chen, C.; Yang, Y.H. Structural Characteristics of Rotate Vector Reducer Free Vibration. *Shock Vib.* **2017**, *2017*, 4214370. [[CrossRef](#)]
22. Jin, S.S.; Tong, X.T.; Wang, Y.L. Influencing Factors on Rotate Vector Reducer Dynamic Transmission Error. *Int. J. Autom. Technol.* **2019**, *13*, 545–556. [[CrossRef](#)]
23. Hu, Y.H.; Li, G.; Zhu, W.D.; Cui, J.K. An Elastic Transmission Error Compensation Method for Rotary Vector Speed Reducers Based on Error Sensitivity Analysis. *Appl. Sci.* **2020**, *10*, 481. [[CrossRef](#)]
24. Kingma, D.P.; Ba, J.L. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA, USA, 7–9 May 2015.
25. Gidon, A.; Zolnik, T.A.; Fidzinski, P.; Bolduan, F.; Papoutsis, A.; Poirazi, P.; Holtkamp, M.; Vida, I.; Larkum, M.E. Dendritic action potentials and computation in human layer 2/3 cortical neurons. *Science* **2020**, *367*, 83–87. [[CrossRef](#)] [[PubMed](#)]
26. Liu, G. It may be time to improve the neuron of artificial neural network. *TechRxiv*, 2020, preprint.
27. Mishra, B.P.; Panigrahi, T.; Wilson, A.M.; Sabat, S.L. A robust diffusion algorithm using logarithmic hyperbolic cosine cost function for channel estimation in wireless sensor network under impulsive noise environment. *Digit. Signal Process. Rev. J.* **2022**, *123*, 103384. [[CrossRef](#)]
28. Zulfiqar, A.K.; Tanveer, H.; Ijaz, U.H.; Fath, U.M.U.; Sung, W.B. Towards efficient and effective renewable energy prediction via deep learning. *Energy Rep.* **2022**, *8*, 10230–10243.
29. Song, Z.Y.; Tang, C.; Ji, J.K.; Todo, Y.; Tang, Z. A Simple Dendritic Neural Network Model-Based Approach for Daily PM_{2.5} Concentration Prediction. *Electronics* **2021**, *10*, 373. [[CrossRef](#)]