OPTIMIZATION OF MACHINE TRANSLATION MODEL BASED ON DBOA-BP NEURAL NETWORK

Limin HAN[1,*], Hong GAO[2], Rongjie ZHAI[3]

To enhance the translation quality of neural machine translation (NMT), a developed butterfly optimization algorithm (DBOA) and back propagation (BP) neural network were applied to optimize the dropout and Learning_data parameters in the machine translation model. Then, a neural machine translation model was built based on sequence-to-sequence (Seq2Seq) model with attention mechanism, and the training data of BP neural network was obtained after many times of training. Meanwhile, the key parameters of the neural network translation model were optimized by DBOA, which was mainly improved by two strategies: changing weights dynamically and adjusting switch coefficients of searching mode dynamically. With the Bleu value as the fitness value, DBOA was combined with the BP *neural network to optimize the dropout and Learning_data parameters in the NMT model to achieve the theoretical optimal Bleu value. The dropout values and Learning_data solved by the algorithm were substituted into the NMT model to get the true Bleu value, which was approximately the same as the predicted value. Thus, the parameters of dropout and Learning_data in the neural translation model were effectively optimized, so the translation quality was developed to a certain extent.*

Keywords: BP neural network; Seq2Seq model; Butterfly optimization algorithm; Algorithm improvement; Neural machine translation

1. Introduction

Further study of computer technology provides a good platform for neural machine translation (NMT) technology. For more efficient international communication, many scholars have conducted deep research on machine translation and made some scientific achievements. For example, some scholars have built machine translation models from different angles. Tezcan expanded the calculation range of machine translation indexes and established a fuzzy machine translation model, which achieved good results [1]. Kumar studied the key problems in eliminating ambiguity of word sense in order to improve the expressing ability of captured semantic texts [2]. To solve the machine translation problem, a new sequence-to-sequence (Seq2Seq) advanced training approach was proposed and proved to be efficient by experiments [3]. At present, a processing method of sub-word segmentation based on data compression algorithm was

[1] Associate Prof., School of Foreign Studies, Anhui Polytechnic University, China, corresponding e-mail: xp@ahpu.edu.cn
[2] Prof., School of Artificial Intelligence, Anhui Polytechnic University, China, e-mail: gaohong0706@ahpu.edu.cn
[3] Postgraduate, School of Artificial Intelligence, Anhui Polytechnic University, China, e-mail: 2200120174@stu.ahpu.edu.cn

proposed, which effectively improved the training level of NMT model [4]. Maimaiti studied the paradigm with few data source and transfer learning method with an aim to enhance the poor quality of NMT due to lack of data [5]. To sum up, the major problem of machine translation lies in that the input sequence is not equal to the output one. Therefore, in this study, the Seq2Seq model was adopted, because it was very suitable for solving this kind of problem. In addition, attention mechanism could effectively reduce the limitation of Seq2Seq model in long sentences' translation.

At the moment, the research on NMT model has been conducted widely, but the translation quality is still limited by model parameters, model organization and word processing. Therefore, some scholars have developed their proposed machine translation model by different methods in terms of parameters, principles and mechanisms. To tackle the inability of English-Chinese translation software to appreciate and understand sentence features, a semantic block processing method was proposed, whose open test showed that the success rate of semantic pattern corpus model matching became higher [6]. Moreover, some scholars have studied different text recognition methods to improve the translation quality of machine translation models [7]. Some scholars have also studied linguistic features of different orders and proposed a denoising and self-encoding method to improve translation quality [8]. At present, NMT model parameters are mainly from researchers' rich experience, so the model cannot achieve good enough translation effect. Although optimization algorithms have presented stable and excellent solving ability in parameter setting, there is little research on setting NMT model parameters by optimization algorithm, leaving much room for future research. If model parameters are optimized by an optimization algorithm and back propagation (BP) neural network, translation quality of NMT models will be greatly improved. The butterfly optimization algorithm (BOA), with strong search ability and high accuracy, was proposed in 2015 [9]. Given that BOA is easily affected by local optimal solution in problems solving process, this study develops its solving ability through optimizing algorithm searching method and adding the adaptive variable weight. Moreover, reliability and effectiveness of the developed butterfly optimization algorithm (DBOA) is verified through comparing it with other optimization algorithms. To sum up, with an aim to enhance translation quality of NMT model, this study optimizes key parameters by DBOA. Finally, it is proved that parameters optimized by DBOA-BP neural network can effectively improve the translation accuracy of NMT model.

This paper mainly includes three parts. In the second part, the NMT model was established based on the Seq2Seq with the attention mechanism. In the third part, two improvement strategies for BOA were proposed; in the fourth part, parameters of the NMT model were optimized by combining the DBOA and BP neural network models. Finally, the study results were sorted out, effectiveness of the algorithm improvement and optimization of NMT parameters were illustrated.

2. Machine Translation Model

2.1 Translation Model based on Seq2Seq

Seq2Seq is a model of generating a sequence based on another sequence, which involves two processes: understanding the previous sequence and generating a new sequence from the understood content [10]. Seq2Seq is very suitable for models with uncertain output length, such as the model in Fig. 1, in which the input length of Chinese sentences is 4, and the output length of English sentences is 3. Specifically, the network structure outputs an English sequence according to the given Chinese sequence sequentially. The results output first will affect those output later.

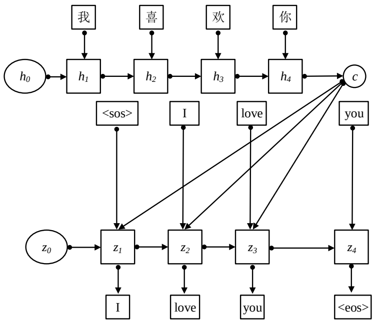


Fig. 1. Schematic diagram of Seq2Seq

Seq2Seq mainly includes two parts, Encoder and Decoder, namely two cyclic neural networks. The Encoder receives the input information at the initial moment, and extracts features of a word in the input Chinese sentence. After a few moments, all the words will be compressed into the hidden state of the Encoder and form a transition vector *c*. The final output features of the Encoder are decoded by the Decoder to generate the target language, in which the target words are generated sequentially, so the model adopts the real target text during training. The learning goal of the model training is to build the correct mapping between input and target sequence, as well as reducing neural network loss by regularly updating parameters in the training process.

The attention mechanism makes it possible for the Decoder to “focus” on different parts of encoder output for each step of its own output. Firstly, a group of attention weights are calculated and multiplied by the encoder output vector to realize weighting and generate attention vector. Then, this attention vector is spliced with the Decoder’s current input as Gate Recurrent Unit (GRU) input. The introduction of attention mechanism can refer to the hidden state of coding at all times to effectively improve translation quality.

As Chinese sentences cannot be segmented by spaces in pre-processing process, jieba, a Python-based Chinese word splitting open-source project, is applied. It can operate word segmentation, lexical annotation and keyword extraction for Chinese texts with high reliability and versatility. Therefore, it can segment Chinese utterances very well. By contrast, English words are separated by spaces. Therefore, in processing process, English corpus can be divided into sequences by inserting spaces between punctuation marks and the previous word.

Data of corpus is shown in Table 1. The corpus contains 56,000 pairs of everyday phrases, with a total of more than 1400,000 characters. The corpus also contains some sentences commonly used in daily life.

Table 1

Data of corpus

Data	Value
Training data	50346 pairs
Validation data	2000 pairs
Test data	3000 pairs
Length range of Chinese sentences	1-43 words
Length range of English sentences	1-34 words

2.2 Testing Standard

Kishore Papineni once proposed a new method to calculate the evaluation of machine translation in his paper [11]. The modified n-gram precision approach relies primarily on calculating the number of words presented in the reference translation in actual translated sentences. Assuming that the *i-th* word in actual translation exists in reference translation, the number of R_i appearing in actual translation is V_i in reference translated sentence. Then, the calculated number of the *i-th* word is C_i, which is calculated as follows:

$$C_i = \min(R_i, V_i) \tag{1}$$

Then, the coincidence degree P_o of a single actual translated sentence relative to a single reference translated sentence is

$$P_o = \frac{\sum_{i=1}^{m} C_i}{\max(c, r)} \tag{2}$$

Where,

c = the total number of words in a single actual translated sentence;

r = the total number of words in a single reference translated sentence;

m = the total number of words in actual translated sentence that coincide with reference translated sentence.

The correction accuracy fraction P_n for one translation target relative to the n reference translated sentences is solved as follows:

$$Pn = \frac{\sum_{i=1}^{n}\sum_{i=1}^{m} C_i}{\sum_{i=1}^{n}\max(c, r_i)} \tag{3}$$

Where r_i is a word sum in the *i-th* reference translation sentence for a certain target sentence.

Bleu is calculated as follows:

$$Bleu = B_{\mathrm{P}} e^{\left(\sum_{ii=1}^{n} w_{ii} \log p_{ii}\right)} \tag{4}$$

Where,

B_{P} = the penalty required for calculation;

w_{ii} = a positive weight and sum of weights is 1.

$$B_{\mathrm{P}} = \begin{cases} 1 & \sum_{i=1}^{n} c_i > \sum_{i=1}^{n} r_i \\ e^{(1-r/c)} & \sum_{i=1}^{n} c_i \le \sum_{i=1}^{n} r_i \end{cases} \tag{5}$$

3. Improvement of Butterfly Optimization Algorithm

3.1 Butterfly Optimization Algorithm

Butterfly optimization algorithm, proposed in 2015, was adopted in this paper for its advantages in optimizing parameters in machine translation model to achieve a better *Bleu* value. Butterfly population mainly depends on scent produced by butterflies for foraging. The scent of a butterfly will be stronger when it is in a better place, so that surrounding butterflies will approach it. This is the global search of butterfly population. When a butterfly does not find any individual with a stronger scent than itself, it will move away immediately. This process is a local search.

The BOA was calculated in 3 stages in the following way: (1) stage of setting the initial parameters of butterfly population and initializing the population; (2) stage of updating the individual position of butterfly population and searching the population collectively; and (3) stage of outputting the optimal position and optimal solution when the target conditions are met. The initial parameters of BOA are mainly the number of butterfly population and individual butterfly position. The scent of each butterfly is calculated via Equation (6)

$$f = cI^{a} \tag{6}$$

Where,

f = the scent of butterflies;

c = the sensory coefficient of butterflies;
I = the stimulus intensity, related to fitness value;
a = the power exponent.

After the initial butterfly population is initialized, the scent of each initial butterfly can be obtained according to Equation (6), and local and global search can be carried out based on the scent information. g^* is used to indicate the position of the initial butterfly population with the strongest scent, i.e. the optimal position. During the global search process, each butterfly performs position transformation by taking the optimal position as a reference, as is expressed in Equation (7).

$$x_i^{t+1} = x_i^t + (r_1^2 \times g^* - x_i^t) \times f_i \tag{7}$$

Where,
x_i^t = position of the i-th butterfly in the i-th iteration;
r_1 = a random parameter for butterfly position update.

In the local search process, each butterfly randomly updates its position by referring to that of a nearby butterfly, as is expressed in Equation (8).

$$x_i^{t+1} = x_i^t + (r_2^2 \times x_j^t - x_k^t) \times f_i \tag{8}$$

Where, x_j^t and x_k^t respectively represent the k-th butterfly and the j-th butterfly randomly selected in the solution set. r_2 is the random parameter for butterfly position updating.

In position updating process, each butterfly can select only one search method of position updating in each generation. In BOA, a judgment coefficient p is set and a random number is generated to compare with p before position updating of each butterfly. Different updating methods are adopted according to the comparison results.

3.2 BOA Improvement through Changing Weight Dynamically

The BOA will have a poor solution accuracy when there are multiple local optimal solutions, so the nonlinear adaptive inertial weight is adopted to improve it to solve the local optimal values problem. Furthermore, the adopted inertial weight is designed based on the power function.

$$w = ((\max\_iter + 1 - i) / (0.25 \max\_iter))^{-0.2} \tag{9}$$

As is shown in the Fig. 2, the weight dynamically changes along with iterations number increasing and keeps low in the early stage, which prevents individual movement falling into local optimum in the early stage due to excessive influence of optimal solution. However, the weight is greatly developed in the middle and late stage to improve the convergence speed of the population.

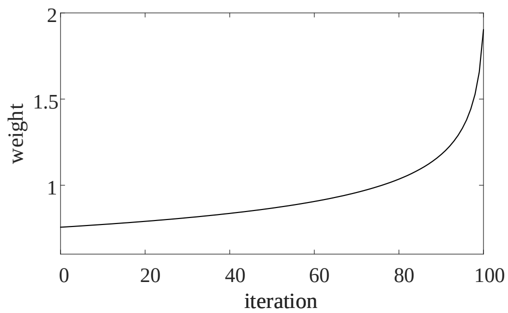


Fig. 2. Curve of dynamically changing weight with iterations

A new global search equation is obtained by transforming the original position updating equation of BOA.

$$x_i^{t+1} = x_i^t + w(r_1^2 \times g^* - x_i^t) \times f_i \tag{10}$$

$$x_i^{t+1} = x_i^t + w(r_2^2 \times x_j^t - x_k^t) \times f_i \tag{11}$$

3.3 BOA Improvement through Adjusting Switch Coefficients of Searching Mode Dynamically

In BOA calculation process, the position updating mode is often chosen by a fixed judgment coefficient *p*, which will limit its solving ability. Hence, the variable judgment parameters are used instead. In the first three-fifths of the calculation, the initial value of the judgment coefficient is around 0.5, which will make half of the butterfly individuals adopt global search and another half adopt local search to promote population diversity. As iterations number increases, more and more individuals will adopt global search, and the judgment coefficient will suddenly turn to 0.9 when it reaches three fifths of total iteration times to ensure the convergence accuracy of the population.

$$\begin{cases} p = 0.5 + (1/(40\max\_iter)) & 0 < i < 0.6\max\_iter \\ p = 0.9 & 0.6\max\_iter \le i \end{cases} \tag{12}$$

The flow chart of DBOA is shown in Fig. 3.

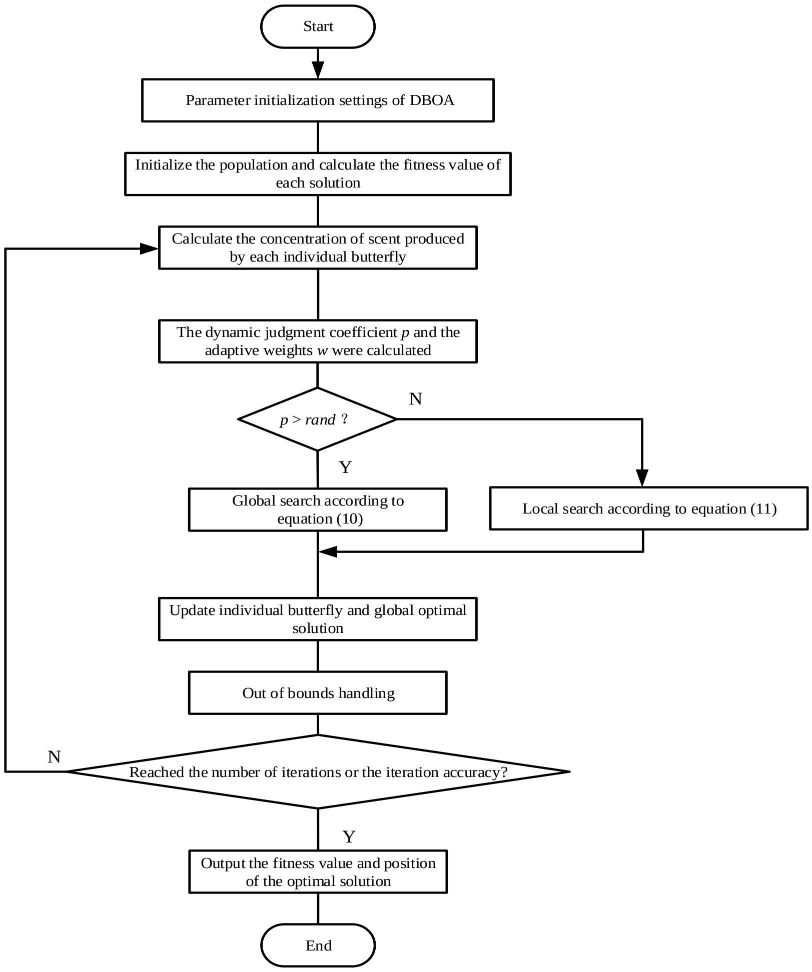


Fig. 3. Flow chart of DBOA

3.4 Verification of Optimization Effect of DBOA

Six test functions were used to compare solving abilities of the gray wolf optimization (GWO) algorithm, BOA and DBOA. Detailed information of the six benchmark functions is shown in Table 2.

Table 2

Benchmark functions used in this study.

Function	V_no	Range	f_{min}
$F_1(x)=\sum_{i=1}^{n} x_i^{\ 2}$	30	[-100,100]	0
$F_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	30	[-10,10]	0
	30	[-30,30]	0
$F_4(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	30	[−5.12,5.12]	0

$F_3(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$			
$F_5(x)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x^2})-e(\frac{1}{n}\sum_{i=1}^{n}cos\left(2\pi x_i\right))+2+\exp(1)$	30	[−32,32]	0
$F_6(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	30	[−600,600]	0

The six functions used in the test are classified into unimodal and multimodal functions to examine solving ability of the optimization algorithm from different angles. The solving abilities, accuracy and speed of three optimization algorithms can be compared through solving unimodal functions. Meanwhile, abilities of jumping out of local optimum solution can be measured through solving multimodal functions.

The iterative curves of the three algorithms results are shown in Fig. 4. The results show that BOA significantly outperforms GWO algorithm in solving different benchmark functions. The BOA improvement is also effective, but the DBOA shows faster convergence speed and accuracy in different function solving processes. To verify the reliability of algorithm improving strategy, the above functions were solved several times in this test. The reliability of the developed algorithm was evaluated by calculating the average and standard deviation of 30 solutions. Those calculation results are presented in Table 3. Table 3 shows that the solution ability of the DBOA is obviously superior. The comparison of the average solution shows that DBOA keeps the solution accuracy much higher than the other two algorithms. The comparison of the standard deviation of the solution results shows that DBOA generally guarantees a more stable solution ability.

4. Parameters Setting and Simulation Experiment of DBOA-based Model

The parameters of machine translation algorithm determine translation quality. In this paper, two parameters, *dropout* and *Learning_data*, with great influence on translation quality, are set. The DBOA is combined with BP neural network by Matlab to calculate the optimal values of *dropout* and *Learning_data*.

The BP neural network is a functional model to show the relationship between input and output on the basis of training parameters. The predicted value can be obtained by importing the test data into the model. A difference is obtained by comparing it with the corresponding actual values in the test data. The established function model adjusts the weight parameter according to the difference to reduce the error. A suitable set of parameters is found through multiple loop iterations, and a complex function is fitted. The BP neural network map is displayed in Fig. 5.

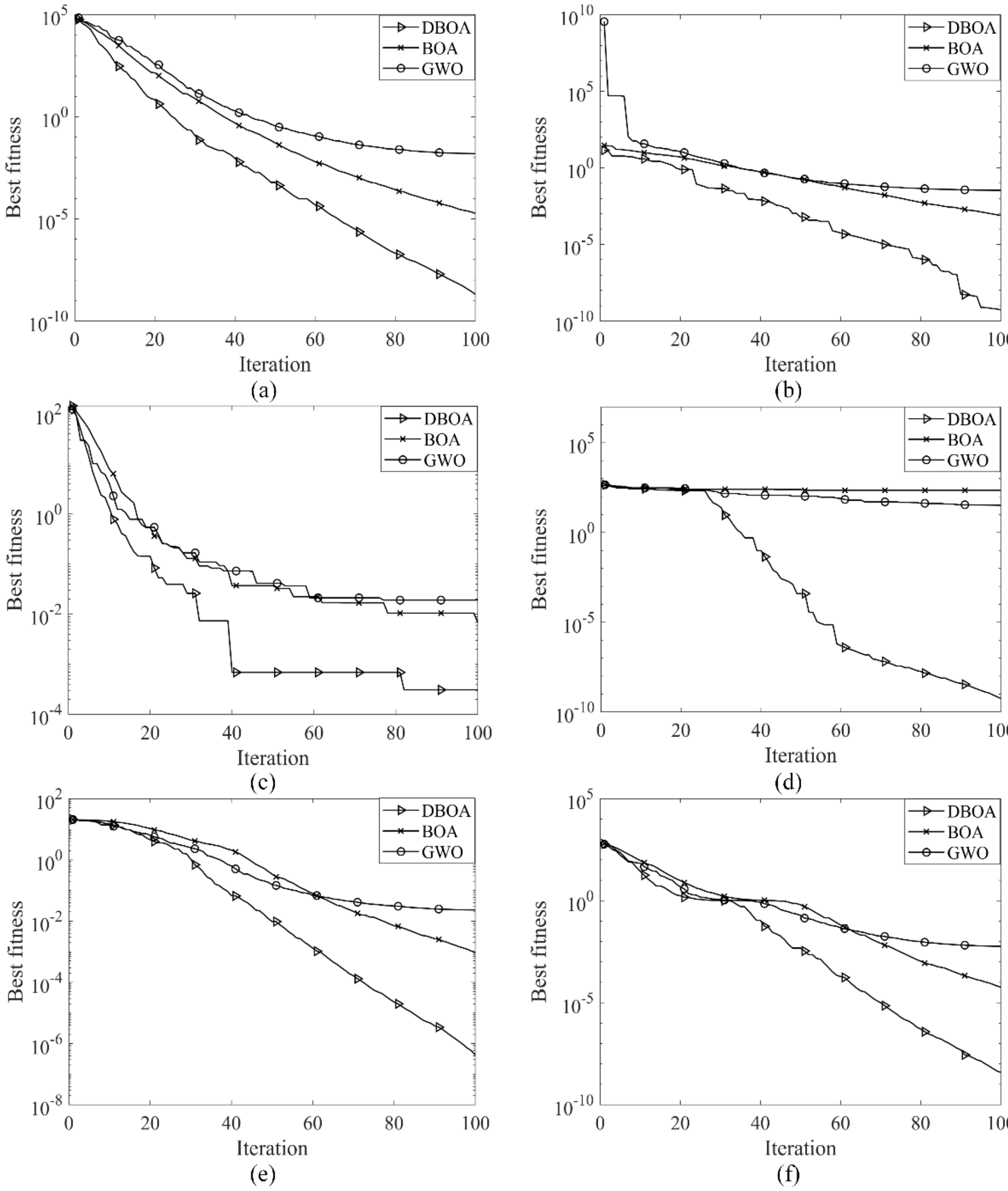


Fig. 4. Iterative curves of solution result of test function: (a) convergence curve of benchmark function F1; (b) convergence curve of benchmark function F2; (c) convergence curve of benchmark function F3; (d) convergence curve of benchmark function F4; (e) convergence curve of benchmark function F5; (f) convergence curve of benchmark function F6.

Table 3

Average and standard deviation of multiple solutions of test function

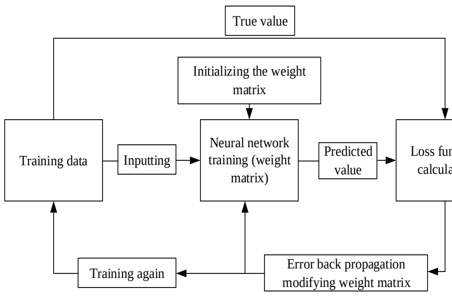
F	GWO		BOA		DBOA	
	ave	*std*	*ave*	*std*	*ave*	*std*
F1	0.024549	0.01265	2.5342e-05	2.1011e-06	3.328e-09	2.6914e-10
F2	0.035982	0.0094249	1.049e−03	1.600e−04	3.1634e-09	2.3749e-09
F3	0.030325	0.0084217	0.0098781	0.0026595	0.0021154	0.0011573
F4	43.5377	9.4449	54.0239	75.455	1.9729e-09	8.4558e-10
F5	0.03871	0.011424	0.0014128	8.5922e-05	7.6613e-07	5.8055e-08
F6	0.14178	0.064764	8.5223e-05	8.4902e-06	4.1798e-09	5.8768e-10

A neural network is actually a function from an input vector to an output vector, as is shown in Equation (13).

$$y = f_{network}(x) \tag{13}$$

Where,
y = the output vector;
x = the input vector;
$f_{network}$ = the neural network function.

As shown in Fig. 6, during the neural network training process, values in the input vector are firstly assigned to the corresponding computational units in the input layer. The value of each computational cell in each layer can be calculated according to the formula mentioned in reference [12]. Finally, the output vector y is obtained by integrating value y_i of each computational unit into the output layer.



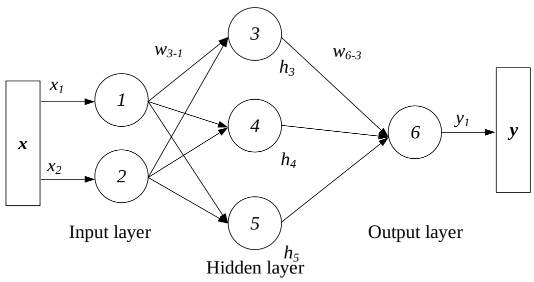


Fig. 5. Schematic diagram of BP neural network Fig. 6. Topology diagram of neural network

In Fig. 6, x_i is the input cell, $w_{i\text{-}j}$ is the weight between node i and node j, h_i is the output value of each cell in the hidden layer, and y_i is the output cell. For BP neural network, the error term of each node should also be calculated. The training samples were set to (x, d), in which x was an eigenvalue and d was a target value of the training samples.

First, this experiment needs to train the Seq2Seq-based machine translation model, and the required experimental environment is shown in Table 4. The deep learning framework used in this experiment is Pytorch l.7.1, and the jieba version is 0.42.1. This experiment aims to continue to build the model in the framework of Pytorch 1.7.1 and use GPU to accelerate the model training in Windows 10 environment, so as to conduct the translation experiment. In the Seq2Seq model, the dimensions number of the hidden units of the Encoder and Decoder is 256; the training optimizer used in the model is Stochastic Gradient Descent (SGD), and beam_size of the bundle search is 4. Each time-step batch processes 1000 sets of data, namely batch_size =1000, and the experiments involve calculation of 200 iterations, namely epoch=200. To improve the model training speed, the teacher_forcing method is introduced, and the odds of adopting it during the training process is 0.5, namely, teacher_forcing_ratio=0.5.

Table 4

Experimental environment configuration.

Properties	Value
CPU	12th Gen Intel (R) Core (TM) i7-12700K
GPU	GeForce RTX 3080
Graphics Memory	10G
Programming Language	Python 3.8.13
Deep learning framework	Pytorch 1.7.1
Operating System	Windows 10

Set 15 sets of *dropout* and *Learning_data* before training, and each group of model corresponding to *dropout* and *Learning_data* gets a corresponding *Bleu* value. The *dropout* and *Learning_data* and the corresponding *Bleu* values are shown in Table 5.

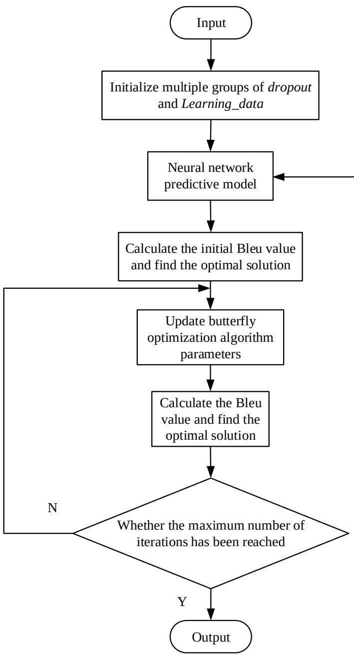
Table 5

Training parameters of neural network

dropout	*Learning_data*	*Bleu*
0.1	0.001	0.32462
0.1	0.005	0.57893
0.1	0.007	0.52871
0.1	0.0004	0.28785
0.1	0.0006	0.27824
0.11	0.01	0.50566
0.11	0.03	0.31879
0.11	0.003	0.66857
0.11	0.005	0.49000
0.11	0.0004	0.00001
0.15	0.001	0.41795
0.15	0.002	0.44186
0.15	0.003	0.49857
0.15	0.005	0.55440
0.15	0.0002	0.00002

The 15 groups of *dropout* and *Learning_data* are the neural network model inputs, and the corresponding 15 *Bleu* values are used as the model output. The neural network trained by the above parameters is used as the fitness function, and the *dropout* and *Learning_data* corresponding to the highest *Bleu* value can be obtained by solving DBOA. Specifically, BOA is firstly used to generate groups of initial *dropouts* and *Learning_data*, which are then introduced into the trained neural network to get multiple predicted *Bleu* values. Other solutions are adjusted according to the *dropout* and *Learning_data* corresponding to the optimal *Bleu* value. Finally, the optimal *Bleu* value and its corresponding *dropout* and *Learning_data* values are output through continuous circulation until the condition of jumping out of circulation is met. Fig. 7 shows the flow chart of parameter setting of DBOA-BP neural network.

The optimization results based on DBOA-BP neural network are *dropout* = 0.12989, *Learning_data* = 0.004149, respectively, and the corresponding *Bleu* prediction value is 0.93329. Fig. 8 shows the iterative curve of *Bleu* in DBOA solving process. In order to check translating effect of the NMT model optimized by DBOA proposed in this paper, experiments are conducted. The translation sample 1 and results are shown in Table 6 and Table 7.



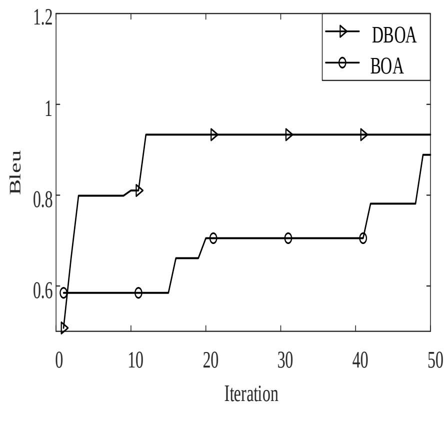


Fig. 7. Flow chart of parameter setting Fig. 8. Iterative curve of optimization

As it is shown in Table 7, *Bleu* values of this model are compared with those of traditional translating models such as RNNSearch [13], SentAlign [14], Transformer [15], GNMT + RL [16] and ConvS2S [17]. The *Bleu* value of translation sample 1 is 0.92157, which is higher than those of SentAlign, Transformer and RNNSearch by 0.8378, 0.86929 and 0.90779, respectively. The *Bleu* value 0.92157, is very close to the predicted result, indicating that the optimization effect of translation model parameters is significant.

Table 6

Comparative analysis of translation results of Sample 1

Translation sample 1	Reference translation	Practical translation
我们怎样才可以救汤姆?	How can we save Tom?	How can we see Tom?
我已经吃过午餐了。	I have already eaten lunch.	I have already eaten lunch.
我永远不会再看到他。	I will never see him again.	I will never see him again.
我有朝一日会成为一名医生。	I will be a doctor someday.	I will be a doctor.
我站在你这边。	I am on your side.	I am on your.
我想和你一起去。	I want to go with you.	I want to go with you.
你有地图吗？	Do you have a map?	Do you have a driver?
我正在做我的作业。	I am doing my homework.	I am doing my homework.
她是一个著名的歌手。	She is a noted singer.	She is a singer.
我觉得这本书很简单。	I think this book is easy.	I think this is is easy.

The performance was compared with the published work on neural machine translation, as can be seen from Table 7.

Table 7

Comparison of B*leu* values of Sample 1

Translation models	*Bleu* values
RNNSearch	0.83758
GNMT + RL	0.87425
Transformer	0.86929
ConvS2S	0.85329
SentAlign	0.90779
Model in this paper	0.92157

Bleu value of model in this paper increases 0.08399, 0.04732, 0.05228, 0.06828 and 0.01378 compared with RNNSearch, GNMT+RL, Transformer, ConvS2S and SentAlign, respectively. In order to further compare translation effects of NMT model optimized by DBOA, experiments are conducted again based on translation sample 2. The translation sample 2 and results are shown in Table 8 and Table 9.

Table 8

Comparative analysis of translation results of Sample 2

Translation sample 2	Reference translation	Practical translation
你知道那是你的责任。	You know that is your duty.	You know that is you duty.
你需要更有进取心。	You need to be aggressive.	You need to be aggressive.
他把他的自行车刷成红色。	He painted his bicycle red.	He painted his bicycle red.
他寄给我一张生日贺卡。	He sent me a birthday card.	He sent me a birthday card.
我忘记他叫什么名字了。	I forgot what his name was.	I forgot what his name.
我没时间阅读。	I have no time for reading.	I have no time for reading.

我常常访问我的亲戚。	I often visit my relatives.	I often visit my.
我什么都不怕。	I am not afraid of anything.	I am not afraid of anything.
这间房里很热。	It is very hot in this room.	It is very hot in this room.
我的母亲在做蛋糕。	My mother is making a cake.	My mother is making a cake.

Table 9

Comparison of B*leu* values of Sample 2

Translation models	*Bleu* values
RNNSearch	0.87261
GNMT + RL	0.89145
Transformer	0.87886
ConvS2S	0.86733
SentAlign	0.89324
Model in this paper	0.91442

Table 9 shows that *Bleu* value of the proposed model is higher than that of RNNSearch, GNMT+RL, Transformer, ConvS2S and SentAlign models by 0.04181, 0.02297, 0.03556, 0.04709 and 0.02118, respectively. To summarize, compared with other methods, the method in this paper incorporated an attention mechanism into the seq2seq model and optimized model parameters by an improved butterfly optimization algorithm. The experimental results verified its effectiveness in improving training effect and this method could be applied to different translation models with parameters. Therefore, the above experimental results show that translation effect of NMT model optimized by DBOA is better than that of traditional models, which verifies that the method of optimizing NMT through DBOA is effective.

5. Conclusion

In view of problems in machine translation, a developed butterfly optimization algorithm was proposed to optimize parameters of neural network machine translation. Two improvement strategies, changing weights dynamically and adjusting switch coefficients of searching mode dynamically, effectively developed solution speed and precision of BOA. The solution ability of DBOA is obviously higher than other algorithms. The optimized result of DBOA-BP neural network is as follows: *dropout* is 0.12989; *Learning_data* is 0.004149; *Bleu* value is 0.93329. By introducing the optimized results into the NMT model, *Bleu* values of the translation results of two samples were 0.92157 and 0.91442, respectively. The translation results of NMT model optimized by DBOA were more satisfactory. These actual *Bleu* values are in line with the predicted results. Therefore, the method of rectifying key parameters of the machine translation model through DBOA-BP neural network is effective. By adopting this method, the translation ability of machine translation model is significantly improved.

ACKNOWLEDGEMENTS. This research was supported by Cooperative project between Anhui Polytechnic University and Hongqi Technology (Hangzhou) Co., LTD (No. HX-2022-12-135).

REFERENCES

[1] *Arda Tezcan, Bram Bulté,* "Evaluating the Impact of Integrating Similar Translations into Neural Machine Translation", in Information, **vol.13,** no. 1, Jan. 2022, pp. 1-33

[2] *Chakrawarti Rajesh Kumar, Bansal Jayshri, Bansal Pratosh,* "Machine translation model for effective translation of Hindi poetries into English", in Journal of Experimental and Theoretical Artificial Intelligence, **vol.34,** no. 1, Jan. 2022, pp. 95-109

[3] *Zhuoyuan Mao, Chenhui Chu, Sadao Kurohashi,* "Linguistically-driven Multi-task Pre-training for Low-resource Neural Machine Translation", in Transactions on Asian and Low-Resource Language Information Processing, **vol.21,** no.4, Jul. 2022, pp. 1-29

[4] *Nonaka Keita, Yamanouchi Kazutaka, Tomohiro I., "*A Compression-Based Multiple Subword Segmentation for Neural Machine Translation", in Electronics, **vol.11,** no.7, Apr. 2022, pp.1014-1025

[5] *Mieradilijiang Maimaiti, Yang Liu, Huanbo Luan,* "Enriching the Transfer Learning with Pre-Trained Lexicon Embedding for Low-Resource Neural Machine Translation", in Journal of Tsinghua University (Science and Technology), **vol.27,** no.1, Aug. 2021, pp.150-163

[6] *Yuxiu Yu*, "English Characteristic Semantic Block Processing Based on English-Chinese Machine Translation", in Advances in Multimedia, **vol. 2022**, May 2022, pp.1-9

[7] *Muneer Iqra, Nawab Rao Muhammad Adeel,* "Cross-lingual Text Reuse Detection Using Translation Plus Monolingual Analysis for English-Urdu Language Pair", in Transactions on Asian and Low-Resource Language Information Processing, **vol. 21,** no. 2, Mar. 2021, pp. 1-18

[8] *Hong-Viet Tran, Van-Vinh Nguyen, Hoang-Quan Nguyen,* "Improving Machine Translation Quality with Denoising Autoencoder and Pre-Ordering", in Journal of Computing and Information Technology, **vol. 29,** no. 1, Mar. 2021, pp. 39-56

[9] *Sharma, Tarun K.,* "Enhanced butterfly optimization algorithm for reliability optimization problems" in JOURNAL OF AMBIENT INTELLIGENCE AND HUMANIZED COMPUTING, **vol. 12,** no. 7, Agu. 2020, pp. 7595-7619

[10] *Sutskever Ilya, Vinyals, Oriol, Le Quoc V.,* "Sequence to sequence learning with neural networks" in Advances in Neural Information Processing Systems, **vol. 4**, Jan. 2014, pp. 3104-3112

[11] *Papineni K., Roukos S., Ward T., etc,* "BLEU: a Method for Automatic Evaluation of Machine Translation" in Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), **vol. 2002**, Jul. 2002, pp. 311-318

[12] *Zhang Li, Wang Fulin, Sun Ting, etc,* "A constrained optimization method based on BP neural network" in Neural Computing and Applications, **vol. 29,** Jul. 2016, pp. 413-421

[13] *Bahdanau Dzmitry, Cho KyungHyun, Bengio Yoshua.* "Neural machine translation by jointly learning to align and translate" in ICLR, San Diego, 2015:1-1

[14] *Vaswani Ashish, Shazeer Noam, Parmar Niki, etc,* "Attention is all you need" in NIPS. Los Angeles, 2017: 5998-6008

[15] *Tu Zhaopeng, Liu Yang, Shang Lifeng, etc,* "Neural machine translation with reconstruction" in Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. SanFrancisco, 2017: 3097-3103.

[16] *Wu Yonghui, Schuster Mike, Chen Zhifeng, etc,* "Google's neural machine translation system: Bridging the gap between human and machine translation" in arXiv preprint arXiv:1609.08144, 2016.

[17] *Gehring Jonas, Auli Michael, Grangier David, etc,* "Convolutional sequence to sequence learning" in Proceedings of the 34th International Conference on Machine Learning, Sydney, 2017: 1243-1252.