APEM journal
Advances in Production Engineering & Management
Volume 16 | Number 1 | March 2021 | pp 23–36
https://doi.org/10.14743/apem2021.1.382
ISSN 1854-6250
Journal home: apem-journal.org
Original scientific paper

A dynamic job-shop scheduling model based on deep learning

Tian, W.[a,*], **Zhang, H.P.**[b]

[a]School of Accounting, Henan Finance University, Zhengzhou, P.R. China
[b]School of Management and Economics, North China University of Water Resources and Electric Power, Zhengzhou, P.R. China

ABSTRACT

Ideally, the solution to job-shop scheduling problem (JSP) should effectively reduce the cost of manpower and materials, thereby enhancing the core competitiveness of the manufacturer. Deep learning (DL) neural networks have certain advantages in handling complex dynamic JSPs with a massive amount of historical data. Therefore, this paper proposes a dynamic job-shop scheduling model based on DL. Firstly, a data prediction model was established for dynamic job-shop scheduling, with long short-term memory network (LSTM) as the basis; the Dropout technology and adaptive moment estimation (ADAM) were introduced to enhance the generalization ability and prediction effect of the model. Next, the dynamic JSP was described in details, and three objective functions, namely, maximum makespan, total device load, and key device load, were chosen for optimization. Finally, the multi-objective problem of dynamic JSP scheduling was solved by the improved multi-objective genetic algorithm (MOGA). The effectiveness of the algorithm was proved experimentally.

ARTICLE INFO

Keywords:
Long short-term memory (LSTM);
Dynamic job-shop scheduling;
Multi-objective genetic algorithm (MOGA);
Adaptive moment estimation (ADAM)

**Corresponding author:*
51980572@qq.com
(Tian, W.)

Article history:
Received 24 February 2021
Revised 4 March 2021
Accepted 8 March 2021

1. Introduction

With the advent of the new industrial era, the old production model has gradually been replaced by the flexible, highly integrated, and automated production model of modern intelligent manufacturing. In modern intelligent manufacturing, the production process focuses on the coordination and balance of various links, such as production, supply, sales, transportation, and storage, and attaches equal importance to economic benefits as well as the efficiency, quality, and safety of production [1-4].

The rational allocation of job-shop resources can effectively reduce manpower and material costs of manufacturers. To enhance its core competitiveness, every manufacturer needs to look for an effective way to solve the job-shop scheduling problem (JSP) [5-8]. In actual production, the processing performance is greatly affected by various uncertainties and instabilities. Therefore, it is urgent to solve the job-shop scheduling model under disturbance factors.

The JSP is mostly solved by evolutionary algorithms, swarm optimization algorithms, and artificial intelligence algorithms. The optimal solution is usually obtained through iterations and group search, with the aim to minimize the makespan and energy consumption [9-13]. Shokouhi

[14] classified the dynamic events in actual production process into four categories according to processing strategies and processes, namely, addition events, device occupation events, delayable events, and exchangeable events, provided the mathematical model and constraints for two typical dynamic times (i.e., device failure and early delivery), and solved the dynamic JSP with the adaptive genetic algorithm. Somashekhara *et al*. [15] introduced the local search algorithm to speed up the discrete PSO, and designed a task-driven rolling window scheduling strategy to cope with the dynamic events like the shutdowns caused by device failure, order placement, and order cancelation. Shahrabi *et al*. [16] analyzed the properties of three typical dynamic JSPs, including the shutdown caused by staged arrival of single workpieces, that caused by staged arrival of workpiece batches, and that caused by device failure, designed a scheduling simulation test with the scheduling rules as independent variables, and verified that the genetic algorithm with adaptive variable neighborhood search is feasible and effective in solving these dynamic JSPs. Gondran *et al*. [17] explored the JSP under the background of multi-agent technology application: the multi-agent collaboration contract network protocol was improved to control the excessively high traffic and increase the cooperation efficiency; then, a job-shop scheduling model was constructed for multiple agents and objectives and specific constraints; finally, the performance of the adaptive PSO in solving the JSP was tested.

The traditional genetic algorithm and its improvements face several problems in optimizing job-shop scheduling plans: slow convergence, lack of population diversity, and proneness to local optimum trap [18-20]. From the perspectives of parallel structure and multiple swarms, Marzouki *et al*. [21] constructed a modularized dynamic job-shop scheduling model, which contains a dynamic database module, a genetic algorithm module, and a re-scheduling program application module; their model integrates the advantages of the elite population protection strategy in the hierarchical evaluation system in improving crossover and mutation probabilities. Keddari *et al*. [22] adopted bee colony optimization to improve a solving algorithm for the multi-objective JSP in actual production. Reddy *et al*. [23] adopted the improved weed algorithm to deal with the multi-objective JSP, with device load, unit device load, and scheduling cycle as optimization goals.

Many have conducted valuable research into the dynamic JSP in actual production process [24-26]. Danielsson *et al*. [27] combined demand prediction with scheduling plan to build a neural network-based prediction model for manhour realization rate, and applied intelligent group optimization algorithm to optimize the multiple objectives in job-shop scheduling and stabilize the manhour realization rate. Teymourifar *et al*. [28] integrated second-order oscillation with optimal network-bandwidth allocation (ONBA) algorithm to obtain the scheduling data of dynamic job-shop model, and greatly shortened the prediction time by improving the learning rate of deep learning (DL) network. Facing the optimization of multiple objectives, resource scheduling in cloud manufacturing extends the time to solve cloud manufacturing tasks. To solve the problem, Teymourifar *et al*. [28] improved the ONBA algorithm by differential evolution algorithm to obtain the scheduling data of the cloud manufacturing model. The data were applied to train the improved deep belief network (IDBN). The learning rate of DL was improved to realize rapid prediction of the scheduling results of cloud manufacturing model. Experimental results show that their method can accurately predict the scheduling results, and shorten the scheduling time, opening a new path to multi-objective group optimization.

In summary, the existing algorithms for dynamic JSP boast strong applicability, and good optimization effect. However, they often converge to the local minimum, and behave stochastically. Compared with traditional shallow neural networks, DL neural networks are excellent in handling complex dynamic JSPs with a huge amount of historical data. Therefore, the authors developed a dynamic job-shop scheduling model based on DL.

In the following parts of the paper: Section 2 establishes a data prediction model for dynamic job-shop scheduling based on long short-term memory network (LSTM), and improves the generalization and prediction performance of the model by Dropout technology and adaptive moment estimation (ADAM). Section 3 describes the dynamic JSP in details, chooses three objective functions, namely, maximum makespan, total device load, and key device load, and constructs the corresponding mathematical model. Based on the predicted data on dynamic job-shop

scheduling obtained in Section 2, Section 4 improves the multi-objective genetic algorithm (MOGA) to handle the multi-objective dynamic JSP, and describes the improved MOGA. Section 5 verifies the effectiveness of the improved MOGA in solving dynamic JSP. Section 6 summarizes the findings of this research.

2. Data prediction model

As a multi-layer representation learning algorithm, DL has a deeper network structure, solves more complex problems, and achieves better prediction accuracy, than ordinary neural networks. The LSTM can effectively overcome the vanishing or exploding gradients of traditional DL neural networks. Fig. 1 illustrates the structure of the LSTM. The typical feature of the LSTM is that the hidden layer nodes are replaced with a storage unit, and four modules, i.e., input door, control door, forget door, and output door, are added to the conventional recurrent neural network.

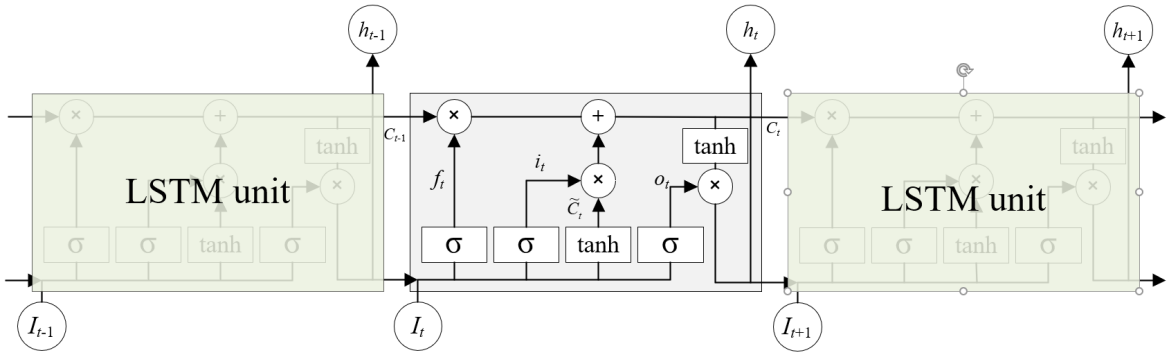

Fig. 1 Structure of the LSTM

Let C_{t-1} be the output of the control door at time $t-1$, which reflects whether to discard the current data; W_i, W_f, W_C, and W_o be the coefficient matrices of input door, forget door, control door, and output door, respectively; σ be the nonlinear activation function sigmoid. Then, the output of the forget door can be calculated by:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, I_t] + \varepsilon_f\right) \tag{1}$$

The input door determines which data to be updated by the LSTM cell. The output of the input door can be calculated by:

$$i_t = \sigma(W_i \cdot [h_{t-1}, I_t] + \varepsilon_i) \tag{2}$$

The new candidate value vector can be created by the tanh function:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, I_t] + \varepsilon_C) \tag{3}$$

The control door determines which data to be updated by the LSTM cell. The value of C_t can be calculated by:

$$C_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{C}_t \tag{4}$$

The output door completes the state update of the LSTM cell, and its output can be calculated by:

$$o_t = \sigma(W_o \cdot [h_{t-1}, I_t] + \varepsilon_o) \tag{5}$$

The input of LSTM cell at time t can be expressed as:

$$h_t = o_t \cdot \tanh(C_t) \tag{6}$$

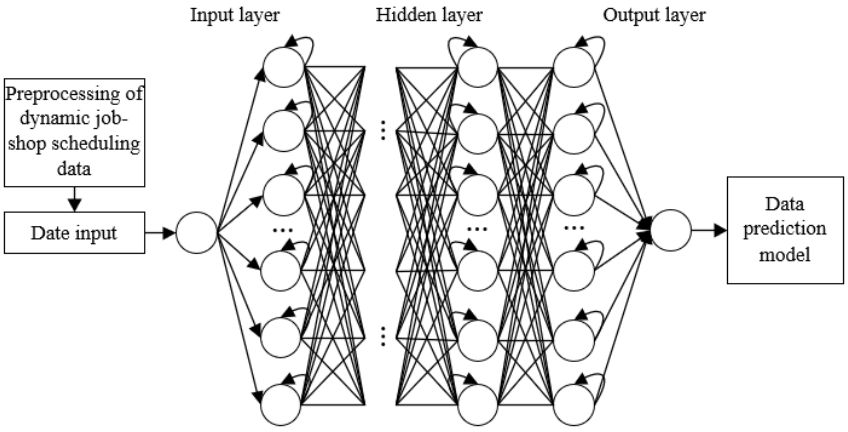

Fig. 2 LSTM-based data prediction model for dynamic JSP

In a DL network model, the parameters will increase exponentially with the increase of network layers and that of layer nodes. The exponential increase of parameters will weaken the generalization ability and prediction performance of the model, increasing the probability of overfitting.

This problem was solved by the Dropout technology. Firstly, half of the hidden layer nodes in the network were randomly chosen, and their inputs and outputs were set to zero. Then, the input I was propagated forward through the new network, and the resulting loss was propagated backward through the network. After some training samples completed the above process, the weights and errors of nodes in the hidden layer, whose inputs and outputs had not been set to zero, were updated by stochastic gradient descent. Then, the nodes whose inputs and outputs had been set to zero were restored. After that, half of the hidden layer nodes in the new network were randomly chosen, and their inputs and outputs were set to zero. This process was repeated again and again. Fig. 3 shows the network structure improved by the Dropout technology.

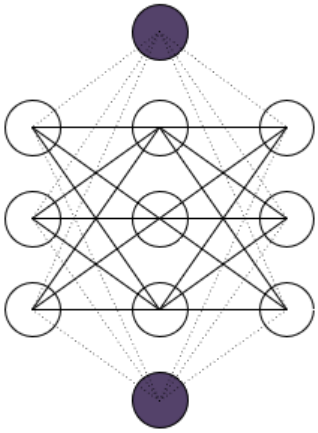

Fig. 3 Network structure improved by the Dropout technology

The learning rate α greatly affects the training efficiency of the network. To boost the efficiency, the key is to improve the adaptivity of the learning rate to the network. In this paper, the ADAM is selected to iteratively compute the gradient of the loss function, and to further update network parameters. Let θ_0 be the initial parameter vector. If parameter θ_t does not converge at time t, then make $t = t + 1$, and obtain the gradient for the new iteration by:

$$g_t = \nabla_\theta f_t(\theta_{t-1}) \tag{7}$$

The first moment vector can be updated by:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{8}$$

where, $\beta_1$is the exponential decay rate of the first moment estimates. The second moment vector can be updated by:

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{9}$$

where, β_2 is the exponential decay rate of the second moment estimates. The calculation deviations for the first and second moment vectors can be updated by:

$$\begin{cases} \hat{m}_t = \dfrac{m_t}{1-\beta_1^t} \\ \hat{v}_t = \dfrac{v_t}{1-\beta_2^t} \end{cases} \tag{10}$$

The network parameters can be updated by:

$$\theta_t = \theta_{t-1} - \frac{\alpha \cdot \hat{m}_t}{\sqrt{\hat{v}_t + b}} \tag{11}$$

where, b is a small constant that prevents the denominator from being zero.

Fig. 4 presents our data prediction model of dynamic job-shop scheduling. The data prediction model is constructed through the following steps based on the deep LSTM:

Step 1: Design the node structure on each layer, and randomly initialize network parameters.
Step 2: Initialize the data on dynamic job-shop scheduling, and divide the vectorized data samples into training and test sets at a certain ratio. During the training, the network loss is calculated through forward propagation as that in backpropagation (BP) neural network, and the network parameters are updated iteratively through ADAM.
Step 3: Import the test samples into the trained model, which outputs the predicted data on dynamic job-shop scheduling.

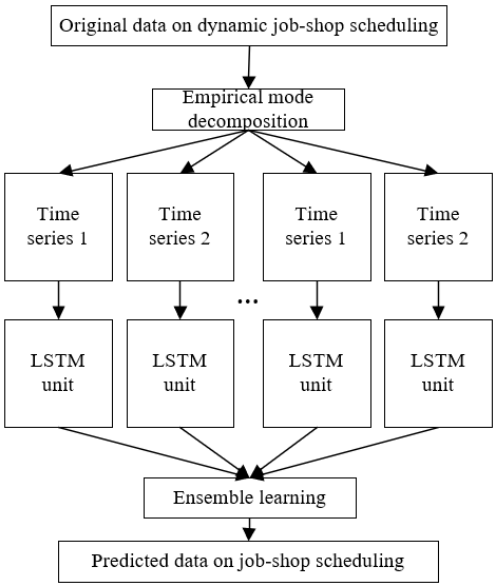

Fig. 4 Structure of the data prediction model

3. Problem description and mathematical modeling

Fig. 5 shows the distribution of processes in the scheduling data on a dynamic car-making job-shop. It can be seen that each workpiece in the dynamic job-shop contains one or more preset processes that can be implemented on different devices. The scheduling performance of the entire production system can be optimized by assigning the ideal device to each process, and ensuring that the processes on each device is sorted in the best possible manner.

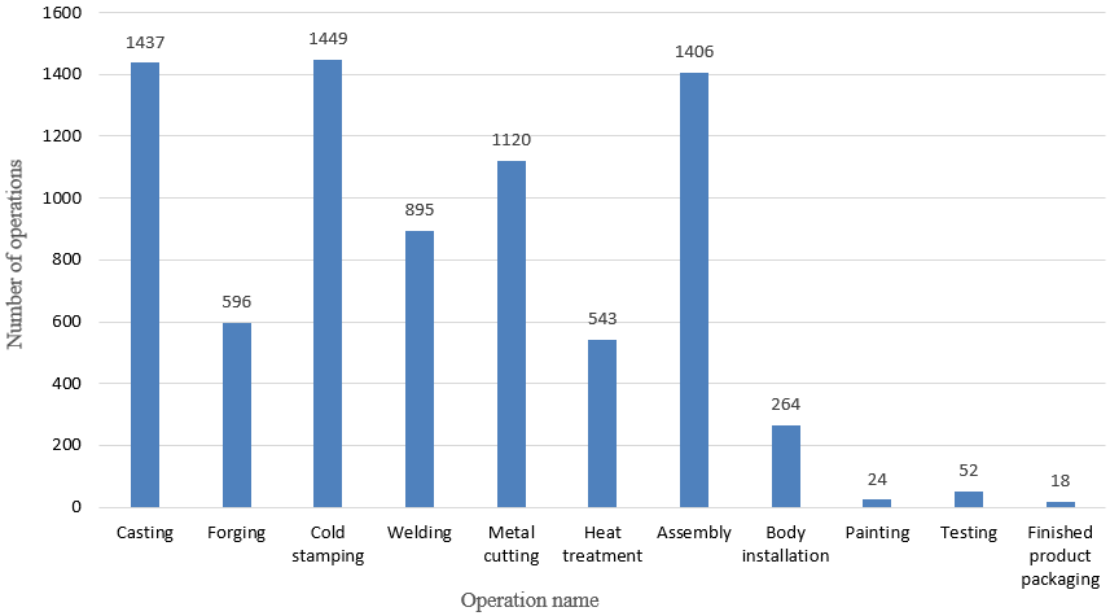

Fig. 5 Distribution of processes in the scheduling data on a dynamic car-making job-shop

Suppose the job-shop has N workpieces A_i to be processed, $i = 1, 2, \ldots, N$, where the i-th workpiece A_i involves N_i processes P_k, $k = 1, 2, \ldots, N_i$, and M devices E_j, $j = 1, 2, \ldots, M$. Let E_{ij} be the set of devices suitable for implementing the k-th process P_{ik} of workpiece A_i, and T_{ijk} be the time required to implement process P_{ik} on the i-th device E_j. Then, the sum of processes for all N workpieces can be expressed as:

$$P_{TOTAL} = \sum_{i=1}^{N} N_i \tag{12}$$

The production cycle of workpieces is greatly affected by the maximum makespan, and the overall utilization of devices in the dynamic job-shop depends largely on device load. Under these constraints, this paper decides to optimize three objective functions: maximum makespan, total device load, and key device load. Among them, the maximum makespan characterizes the production efficiency of the job-shop. The minimization of the maximum makespan needs to satisfy:

$$\min T_{\max} = \max_{1 \le i \le N} \sum_{j=1}^{N} \sum_{k=1}^{N_i} T_{ijk} \tag{13}$$

The total device load characterizes the working time of all devices. The minimization of the total device load needs to satisfy:

$$\min L_{TOTAL} = \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{k=1}^{N_i} T_{ijk} \varphi_{ijk} \tag{14}$$

where, φ_{ijk} is a binary function about whether process P_{ik} is implemented on the j-th device E_j (if $\varphi_{ijk} = 1$, the process is implemented on that device; if $\varphi_{ijk} = 0$, the process is not implemented on that device).

The key device load determines which device is assigned to each process in the scheduling plan. The minimization of the key device load needs to satisfy:

$$\min L_j = \max_{1\le i\le N} \sum_{i=1}^{N} \sum_{k=1}^{N_i} T_{ijk}\varphi_{ijk} \tag{15}$$

The dynamic job-shop scheduling is mainly subject to seven constraints: (1) the spatiotemporal exclusivity of devices, (2) the spatiotemporal exclusivity of processes, (3) the time continuity of processing, (4) the time limit of processing, (5) the sequence of processes, (6) the fairness of the priority of processes, (7) the start-up consistency of devices.

Let ST_{ik} be the start time of the k-th process P_{ik} of A_i. Since each device needs to execute more than one process, the time limit of processing can be defined as:

$$\begin{cases} ST_{ik} + \varphi_{ijk} \cdot T_{ijk} \le \sum_{j=1}^{M} \sum_{k=1}^{N_i} T_{ijk} \\ \sum_{j=1}^{M} \sum_{k=1}^{N_i} T_{ijk} \le T_{max} \end{cases} \tag{16}$$

The time continuity of processing can be defined as:

$$\sum_{j=1}^{M} \sum_{k=1}^{N_i} T_{ijk} \le ST_{i(k+1)} \tag{17}$$

The spatiotemporal exclusivity of processes can be defined as:

$$ST_{ik} + T_{ijk} \le ST_{i(k+1)} \tag{18}$$

Since each process of a workpiece must be assigned a device, the following constraint must be satisfied to find a device in the device set E_j for process P_{ik}:

$$\sum_{j=1}^{M} \varphi_{ijk} = 1 \tag{19}$$

4. Problem solving

For the multi-objective dynamic job-shop scheduling, the MOGA enjoys good optimization performance and model robustness, but faces several defects: the limited population size and the difficulty in saving the optimal solution. It is necessary to improve the algorithm in two aspects: reducing computing complexity and increasing the preservation probability of the optimal individuals.

To ensure the diversity of solutions, this paper replaces the calculation and selection of the crowded distance in the traditional MOGA with the selection of predefined reference points. Whereas three objectives have been chosen for the multi-objective dynamic job-shop scheduling, the reference points were defined on a three-dimensional hyperplane, and each objective was split into Q parts. Then, the number of reference points can be calculated by:

$$N_R = \binom{3 + Q - 1}{Q} \tag{20}$$

The coordinates of each point in the corresponding two-dimensional plane can be expressed as:

$$c_l = \left\{0, \frac{1}{Q}, \frac{2}{Q}, \ldots, 1\right\} \tag{21}$$

The solution diversity of the target problem was maintained by the correlation between each reference point and each solution. The different scales of each objective function can be reflected

by the reference points on the planes in the three-dimensional hyperplane. Therefore, the algorithm's preference for the scale of each objective function should be lowered through adaptive normalization.

Step 1: Regard the minimum of population K_t as the optimal point of the population. Set up the set of the optimal points $K = (K_{1-\min}, K_{2-\min}, \ldots, K_{D-\min})$ to transform the objective functions Eqs. 13-15 by:

$$F_i^*(x) = F_i(x) - K_{i-\min} \tag{22}$$

Step 2: Determine the extra points on the hyperplane by:

$$EP(x,\sigma) = \max_{i=1}^{3} \frac{F_i^*(x)}{\sigma_i}, \qquad x \in K_z \tag{23}$$

The i-th optimal point can be expressed as:

$$K_{i-\min} = c: \operatorname{argmin}_{c \in K_z} EP\left(x, \sigma^i\right), \sigma^i = (10^{-6}, \ldots, 10^{-6}), \qquad \sigma_j^i = 1 \tag{24}$$

Calculate the distance between the hyperplane and each coordinate axis, and perform adaptive normalization on each objective function by:

$$g_i^n = \frac{F_i^*(x)}{x_i - K_{i-\min}} = \frac{F_i(x) - K_{i-\min}}{x_i - K_{i-\min}}, \qquad i = 1,2,3 \tag{25}$$

Step 3: Solve the connected reference line of the first V non-dominated optimal layers. Specifically, calculate the vertical distance on the hyperplane for the reference line ς between the origin and reference point corresponding to each individual K by:

$$D^{\perp}(K,\varsigma) = \|(K - \varsigma^T K\varsigma)/\|\varsigma\|^2\| \tag{26}$$

Then, define the reference point corresponding to the shortest distance as the connected reference point, and the corresponding connected reference line can represent the correlation between the connected reference point and each solution.

Fig. 6 explains the workflow of the proposed algorithm for solving dynamic JSP.

Step 1: Initialize the objective functions, coding method, and parameters.

Step 2: Generate a fixed number of predefined reference points, according to the number of optimization objectives of the multi-objective dynamic job-shop scheduling, as well as the number of equal divisions of each objective.

Step 3: Make the initial number of iterations zero, generate a population $K_t$of the size D that meets the seven constraints of dynamic job-shop scheduling, and perform fitness calculation

Step 4: Generate new individuals through crossover, and obtain a new population K_t' through mutation.

Step 5: Perform fitness calculation for the new population K_t, merge the new and old populations, and conduct rapid dominated and non-dominated sorting of individuals, and obtaining the non-dominated optimal layers. By assigning a shared virtual fitness to all non-dominated individuals, obtain the non-dominated optimal layers, such as rank1, rank, rank3, etc.

Step 6: Perform adaptive normalization of individuals in the next-generation population, compare the vertical distances of all reference lines of all individuals on the hyperplane, obtain the set of reference points connected with all individuals with the shortest distance, and compute the niche of each reference point. The preservation of the niche can be realized through the following steps:

- Let $\Omega_{min} = \{z: argmin_z \upsilon_z\}$ be the set of reference points for the minimum niche.

- Randomly choose reference points from $\Omega_{\min}$, which contains multiple reference points, and assemble them into the set Ψ_z of connected individuals on the V-th layer.

  If Ψ_z is empty, there is no individual on the V-th layer connected with the selected reference point; thus, remove this reference point.

  If Ψ_z is not empty, and if υ_z equals zero, take the individual with the shortest vertical distance as the optimal individual, pass it down to the next generation, and make $\upsilon_z = \upsilon_z + 1$; if υ_z is greater than 1, randomly choose an individual from the V-th layer, pass it down to the next generation, and make $\upsilon_z = \upsilon_z + 1$.

Step 7: After niche preservation, judge if the size of the next-generation population reaches the preset value. If yes, it means the iterations have completed. Then, judge if the number of iterations meets the preset maximum value. If yes, terminate the iteration, and output the result; otherwise, make $t = t + 1$, and continue with the iterative process.

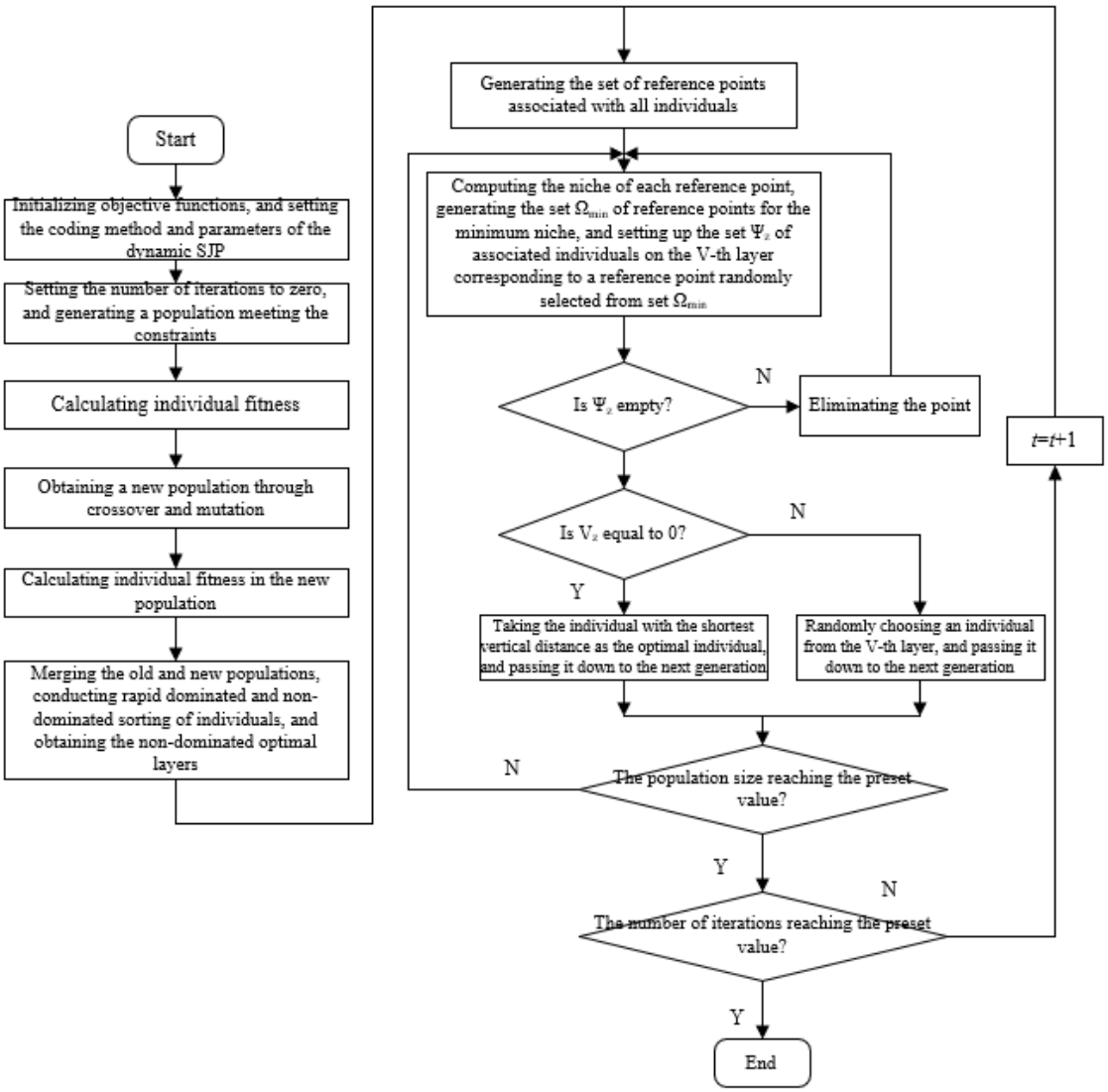

Fig. 6 Workflow of solving dynamic JSP based on MOGA

5. Experiments and results analysis

The hyperparameter setting directly determines the accuracy of the proposed LSTM-based data prediction model for dynamic job-shop scheduling. In this paper, tanh is selected as the activation function of the network, mean squared error (MSE) is taken as the objective function, ADAM is chosen to improve the parameters, and the batch size is set to 30.

The degree of feature extraction from dynamic job-shop scheduling data depends on the hidden layer(s). Table 1 shows the prediction results at different number of hidden layers. Obviously, the MSE, MAE, and MAPE of our model were minimized, when there were five hidden layers. Table 2 presents the prediction results at different number of nodes in the hidden layer. It can be seen that, when there were 25 hidden layer nodes, the MAE and MAPE were minimized, and the MSE was relatively small. Since a small loss means highly accurate prediction, the number of hidden layers and the number of hidden layer nodes were set to 5 and 25, respectively.

Table 1 Prediction results at different number of hidden layers

Number of layers	1	2	3	4	5	6	7	8	9
MSE	23.69	22.44	21.61	20.91	19.18	20.93	21.33	22.64	23.11
MAE	16.71	15.28	14.57	13.99	13.18	13.37	14.29	15.23	16.50
MAPE	8.93	8.54	8.57	7.28	6.84	7.03	7.53	8.27	8.49

Table 2 Prediction results at different number of hidden layer nodes

Number of nodes	5	10	15	20	25	30	35	40
MSE	22.45	21.19	20.64	20.84	20.97	21.09	21.91	22.10
MAE	16.74	15.36	15.49	14.73	14.26	14.58	15.53	15.67
MAPE	7.86	7.37	7.49	7.26	6.81	7.15	7.36	7.89

To verify its effectiveness, the proposed deep LSTM for data prediction of dynamic job-shop scheduling was compared with support vector machine (SVM) through experiments. A total of 7,500 pieces of data were selected to train the model, and 2,500 sets of data to test the model. Tables 3-5 present the test results on any five sets of test data.

Table 3 Test data of objective functions

Serial number	Input data			
	Maximum makespan	Processing cost	Key device load	Total device load
1	51	15.217	77	184
2	62	11.463	89	169
3	55	13.503	93	215
4	64	15.227	84	210
5	59	9.803	76	175

Table 4 Test data of SVM

Serial number	Output data			
	Maximum makespan	Processing cost	Key device load	Total device load
1	48	14.863	83	210
2	68	11.980	80	184
3	53	14.254	91	201
4	60	14.739	98	231
5	64	11.004	84	168

Table 5 Test data of LSTM

Serial number	Output data			
	Maximum makespan	Processing cost	Key device load	Total device load
1	50	14.291	74	192
2	59	10.649	95	173
3	54	15.210	92	206
4	62	15.972	79	199
5	60	9.599	82	167

Based on the test results, this paper selects five indices to measure the performance of each prediction model: accuracy, precision, recall, comprehensive evaluation index (CEI), and area under the curve (ACU). The SVM is a supervised learning model. It is often adopted to analyze data, predict classes, and regress data. Table 6 compares the prediction performance of SVM and LSTM. It can be seen that, when only the processes in Fig. 5 were considered (e.g., casting, forging, cold stamping, welding, metal cutting, heat treatment, assembly, body installation, painting, testing, finished product packaging), the proposed model outperformed the SVM, as it achieved the higher values of all metrics.

Fig. 7 shows that the ROC curve of SVM remained below that of LSTM. This further confirms that our data prediction model for dynamic job-shop scheduling is better than the other models, and is highly feasible.

Normally, multi-objective planning problems can be modeled into the search for Pareto optimal solutions. Compared with traditional genetic algorithm, adaptive genetic algorithm boasts the ability to solve multiple objectives, strong robustness, and excellence in optimization. The adaptive genetic algorithm II is improved from NSGA, by optimizing the elite strategy for retaining the optimal solutions. On this basis, the adaptive genetic algorithm III is derived by replacing the crowded distance. The two algorithms are widely used in many fields. To verify the superiority of our algorithm, this paper sets up a dynamic job-shop scheduling model with three objective functions: minimal maximum makespan, minimal total device load, and minimal key device load. Then, the model was solved by the proposed algorithm, adaptive genetic algorithm II, and adaptive genetic algorithm III. Table 7 lists the maximum, mean, and minimum of the objective functions obtained by the three algorithms. Fig. 8 displays the solution sets of the three algorithms in a three-dimensional space. The mean Euclidean distance from each reference point to the nearest solution was computed to measure the performance of the above three methods. The algorithm with the smallest inverted generational distance has the best performance. It can be seen that our algorithm achieved comparable results as adaptive genetic algorithms II and III in key device load, while significantly outshined the two algorithms in maximum makespan and total device load.

Table 6 Prediction performance of SVM and LSTM

Prediction model	Accuracy	Precision	Recall	CEI	ACU
SVM	92,16	90.62	95.72	96.33	0.875
LSTM	94.37	95.11	96.21	96.78	0.898

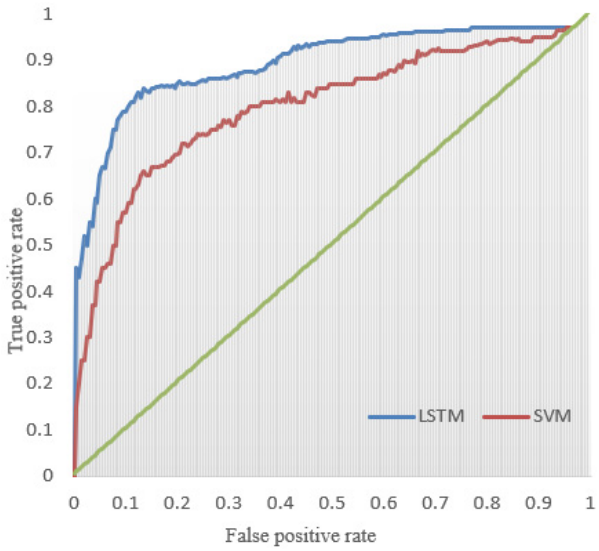

Fig. 7 ROC curves of SVM and LSTM

Table 7 Solving results of different algorithms

Objectives		Adaptive genetic algorithm II	Adaptive genetic algorithm III	Our algorithm
Maximum makespan	Max	64.34	62.15	60.75
	Mean	53.18	54.11	53.77
	Min	49.79	50.17	52.15
Key device load	Max	98.13	92,75	90.22
	Mean	87.84	86.14	86.05
	Min	69.47	71.94	78.25
Total device load	Max	247.49	231.85	210.05
	Mean	181.64	184.23	183.55
	Min	167.21	170.51	170.26

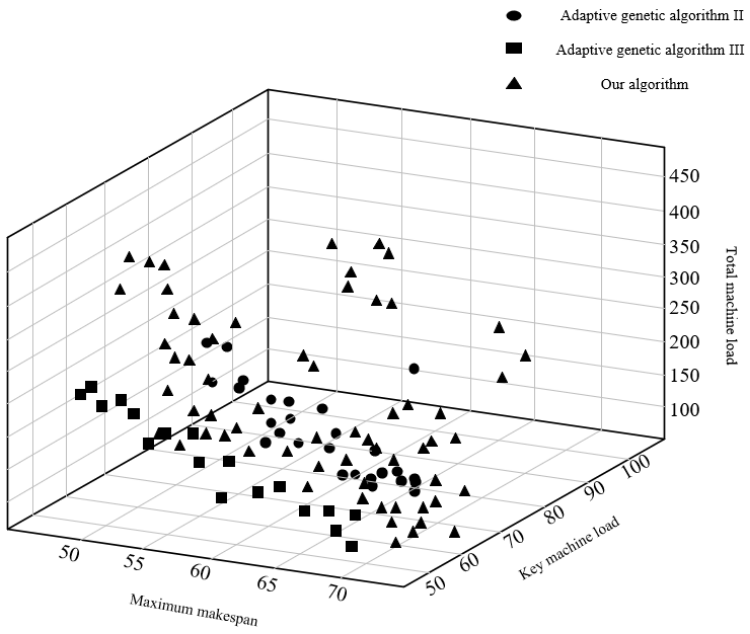

Fig. 8 Solving results of different algorithms for dynamic JSP

6. Conclusion

This paper mainly proposes a dynamic job-shop scheduling model based on DL. Firstly, a data prediction model was innovatively constructed based on LSTM. Then, the Dropout technology and ADAM were introduced to enhance the generalization ability and prediction effect. To evaluate the prediction performance of the model, five metrics were selected, including accuracy, precision, recall, CEI, and ACU. The experimental results show that our model outperformed the SVM. Next, the three objective functions, namely, maximum makespan, total device load, and key device load, were optimized, completing the description of dynamic JSP. Finally, the MOGA was improved innovatively to solve the multi-objective dynamic JSP. From the experimental results, it is inferred that our algorithm achieved comparable results as adaptive genetic algorithms II and III in key device load, while significantly outshined the two algorithms in maximum makespan and total device load.

In future, the research contents will be deepened in the following aspects: the strategy for maintaining population diversity will be further improved, shedding light on combinatory optimization. Better evaluation indices will be found to measure the convergence of the algorithm. The test set will be expanded to reflect the actual production situation of flexible WSP.

Acknowledgement

This paper is supported by Postgraduate Education Reform Project of Henan Province, China (2019SJGLX045Y), Postgraduate Education Reform and Quality Improvement Project of Henan Province, China (HNYJS2018KC02).

References

[1] Ahmadian, M.M., Salehipour, A., Cheng, T.C.E. (2021). A meta-heuristic to solve the just-in-time job-shop scheduling problem, *European Journal of Operational Research*, Vol. 288, No. 1, 14-29, doi: 10.1016/j.ejor.2020.04.017.

[2] Liang, Q. (2020). Production logistics management of industrial enterprises based on wavelet neural network, *Journal Européen des Systèmes Automatisés,* Vol. 53, No. 4, 581-588, doi: 10.18280/jesa.530418.

[3] Vrecko, I., Kovac, J., Rupnik, B., Gajsek, B. (2019). Using queuing simulation model in production process innovations, *International Journal of Simulation Modelling*, Vol. 18, No 1, 47-58, doi: 10.2507/IJSIMM18(1)458.

[4] Zhang, G., Zhang, L., Song, X., Wang, Y., Zhou, C. (2019). A variable neighborhood search based genetic algorithm for flexible job shop scheduling problem, *Cluster Computing*, Vol. 22, No. 5, 11561-11572, doi: 10.1007/s10586-017-1420-4.

[5] Amjad, M.K., Butt, S.I., Anjum, N., Chaudhry, I.A., Faping, Z., Khan, M. (2020). A layered genetic algorithm with iterative diversification for optimization of flexible job shop scheduling problems, *Advances in Production Engineering & Management*, Vol. 15, No. 4, 377-389, doi: 10.14743/apem2020.4.372.

[6] Ojstersek, R., Lalic, D., Buchmeister, B. (2019). A new method for mathematical and simulation modelling interactivity: A case study in flexible job shop scheduling, *Advances in Production Engineering & Management*, Vol. 14, No. 4, 435-448, doi: 10.14743/apem2019.4.339.

[7] Meolic, R., Brezočnik, Z. (2018). Flexible job shop scheduling using zero-suppressed binary decision diagrams, *Advances in Production Engineering & Management*, Vol. 13, No. 4, 373-388, doi: 10.14743/apem2018.4.297.

[8] Zhang, H., Liu, S., Moraca, S., Ojstersek, R. (2017). An effective use of hybrid metaheuristics algorithm for job shop scheduling problem, *International Journal of Simulation Modelling*, Vol. 16, No. 4, 644-657, doi: 10.2507/IJSIMM16(4)7.400.

[9] Koblasa, F., Kralikova, R., Votrubec, R. (2020). Influence of EA control parameters to optimization process of FJSSP problem, *International Journal of Simulation Modelling*, Vol. 19, No. 3, 387-398, doi: 10.2507/IJSIMM19-3-519.

[10] Asadzadeh, L. (2016). A parallel artificial bee colony algorithm for the job shop scheduling problem with a dynamic migration strategy, *Computers & Industrial Engineering*, Vol. 102, 359-367, doi: 10.1016/j.cie.2016.06.025.

[11] Kuhpfahl, J., Bierwirth, C. (2016). A study on local search neighborhoods for the job shop scheduling problem with total weighted tardiness objective, *Computers & Operations Research*, Vol. 66, 44-57, doi: 10.1016/j.cor.2015.07.011.

[12] Lei, D., Guo, X. (2016). A shuffled frog-leaping algorithm for job shop scheduling with outsourcing options, *International Journal of Production Research*, Vol. 54, No. 16, 4793-4804, doi: 10.1080/00207543.2015.1088970.

[13] Wang, C., Zeng, L. (2019). Optimization of multi-objective job-shop scheduling under uncertain environment, *Journal Européen des Systèmes Automatisés*, Vol. 52, No. 2, 179-183, doi: 10.18280/jesa.520210.

[14] Shokouhi, E. (2018). Integrated multi-objective process planning and flexible job shop scheduling considering precedence constraints, *Production & Manufacturing Research*, Vol. 6, No. 1, 61-89, doi: 10.1080/21693277.2017.1415173.

[15] Somashekhara, S.C.H., Setty, A.K.Y., Sridharmurthy, S.M., Adiga, P., Mahabaleshwar, U.S., Lorenzini, G. (2019). Makespan reduction using dynamic job sequencing combined with buffer optimization applying genetic algorithm in a manufacturing system, *Mathematical Modelling of Engineering Problems,* Vol. 6, No. 1, 29-37, doi: 10.18280/mmep.060104.

[16] Shahrabi, J., Adibi, M.A., Mahootchi, M. (2017). A reinforcement learning approach to parameter estimation in dynamic job shop scheduling, *Computers & Industrial Engineering*, Vol. 110, 75-82, doi: 10.1016/j.cie.2017.05.026.

[17] Gondran, M., Huguet, M.-J., Lacomme, P., Tchernev, N. (2019). Comparison between two approaches to solve the job shop scheduling problem with routing, *IFAC-PapersOnLine*, Vol. 52, No. 13, 2513-2518, doi: 10.1016/j.ifacol.2019.11.584.

[18] Zhong, Y., Yang, F., Liu, F. (2019). Solving multi-objective fuzzy flexible job shop scheduling problem using MABC algorithm, *Journal of Intelligent & Fuzzy Systems*, Vol. 36, No. 2, 1455-1473, doi: 10.3233/JIFS-181152.

[19] Phanden, R.K. (2016). Multi agents approach for job shop scheduling problem using genetic algorithm and variable neighborhood search method, In: *Proceedings of the 20th World Multi-Conference on Systemics, Cybernetics and Informatics,* Orlando, Florida, USA, 275-278.

[20] Bożek, A., Werner, F. (2018). Flexible job shop scheduling with lot streaming and sublot size optimisation, *International Journal of Production Research*, Vol. 56, No. 19, 6391-6411, doi: 10.1080/00207543.2017.1346322.

[21] Marzouki, B., Driss, O.B., Ghédira, K. (2018). Solving distributed and flexible job shop scheduling problem using a chemical reaction optimization metaheuristic, *Procedia Computer Science*, Vol. 126, 1424-1433, doi: 10.1016/j.procs.2018.08.114.

[22] Keddari, N., Mebarki, N., Shahzad, A., Sari, Z. (2018). Solving an integration process planning and scheduling in a flexible job shop using a hybrid approach, In: Amine, A., Mouhoub, M., Ait Mohamed, O., Djebbar, B. (eds), *Computational Intelligence and Its Applications, CIIA 2018, IFIP Advances in Information and Communication Technology,* Vol. 522, Springer, Cham, Switzerland, 387-398, doi: 10.1007/978-3-319-89743-1_34.

[23] Sreekara Reddy, M.B.S., Ratnam, C., Rajyalakshmi, G., Manupati, V.K. (2018). An effective hybrid multi objective evolutionary algorithm for solving real time event in flexible job shop scheduling problem, *Measurement*, Vol. 114, 78-90, doi: 10.1016/j.measurement.2017.09.022.

[24] Dabah, A., Bendjoudi, A., AitZai, A. (2016). Efficient parallel B&B method for the blocking job shop scheduling problem, In: *Proceedings of 2016 International Conference on High Performance Computing & Simulation (HPCS),* Innsbruck, Austria, 784-791, doi: 10.1109/HPCSim.2016.7568414.
[25] Maharana, D., Kotecha, P. (2019). Optimization of job shop scheduling problem with grey wolf optimizer and JAYA algorithm, In: Panigrahi, B., Trivedi, M., Mishra, K., Tiwari, S., Singh, P. (eds), *Smart Innovations in Communication and Computational Sciences*, *Advances in Intelligent Systems and Computing*, Vol. 669, Springer, Singapore, 47-58, doi: 10.1007/978-981-10-8968-8_5.
[26] Jahan, M.V., Dashtaki, M., Dashtaki, M. (2015). Water cycle algorithm improvement for solving job shop scheduling problem, In: *Proceedings of 2015 International Congress on Technology, Communication and Knowledge (ICTCK),* Mashhad, Iran, 576-581, doi: 10.1109/ICTCK.2015.7582732.
[27] Danielsson, F., Svensson, B., Reddy, D. (2015). A genetic algorithm with shuffle for job shop scheduling problems, In: *Proceedings of the Modelling and simulation 2015: The European simulation and modelling conference,* Leicester, United Kingdom, 363-367.
[28] Teymourifar, A., Ozturk, G., Ozturk, Z.K., Bahadir, O. (2020). Extracting new dispatching rules for multi-objective dynamic flexible job shop scheduling with limited buffer spaces, *Cognitive Computation*, Vol. 12, No. 1, 195-205, doi: 10.1007/s12559-018-9595-4.