INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
Online ISSN 1841-9844, ISSN-L 1841-9836, Volume: 21, Issue: 1, Month: February, Year: 2026
Article Number: 6873, https://doi.org/10.15837/ijccc.2026.1.6873

CCC Publications

A Hybrid Approach to Fault Diagnosis and Lifespan Prediction in Complex Computer Systems

Shigan Yu, Xiaoling Ru

Shigan Yu
Fuyang Normal University
Fuyang, Anhui 236041, China
yushigan@163.com

Xiaoling Ru*
Fuyang Normal University
Fuyang, Anhui 236041, China
*Corresponding author: ruxiaoling@163.com

Abstract

The increasing integration of software and hardware in modern computer systems has introduced significant reliability challenges, necessitating advanced fault diagnosis and lifespan prediction techniques. This study proposes a hybrid approach leveraging an improved Bat Algorithm (BA) and Hidden Markov Model (HMM) to enhance fault detection and system longevity assessment. By incorporating a genetic competition mechanism, the enhanced BA improves fault classification accuracy, while the HMM-based lifespan prediction model provides superior forecasting precision. Experimental results demonstrate that the proposed fault diagnosis model achieves a 95.68% accuracy and an area under the curve (AUC) of 96.83%, significantly outperforming conventional methods. Moreover, the lifespan prediction model surpasses backpropagation neural networks (BPNN), achieving a lower root mean square error (RMSE) of 0.021, indicating higher predictive reliability. These findings contribute to improving the stability and security of complex computing environments while reducing maintenance costs. Future work will focus on optimizing computational efficiency and extending real-time applications in large-scale systems.

Keywords: complex computer systems, bat algorithm, hidden markov model, reliability, remaining lifespan, fault diagnosis.

1 Introduction

Computer System Reliability Analysis (CSRA) refers to the process of evaluating the system's capacity to carry out specified functions under specified states and within a given time. This process aims to identify faults and risks in the system, analyze their causes, and develop corresponding solutions to improve the system's reliability. Through fault diagnosis and life prediction, potential problems can be identified in advance and preventive measures can be taken, thereby improving the reliability of computer systems [1, 2]. Computer system fault diagnosis refers to the detection of faults that occur in the system. The Fault Diagnosis Model (FDM) analyzes the operational data of the system, identifies abnormal patterns and trends, and promptly identifies potential faults in the system [3]. Through fault diagnosis, the risk of system crashes and data loss can be prevented and reduced, ensuring business continuity and data integrity. In addition, fault diagnosis also helps optimize system maintenance and management, improve system reliability and service life. Computer system life prediction refers to predicting the possible failure time or replacement time of a computer system or component in the future through certain methods and tools. The Life Prediction Model (LPM) focuses on the entire life cycle of the system, and predicts the remaining service life of the system by analyzing its operational data and historical information. LPM can help prepare maintenance and replacement plans in advance, avoiding business interruptions and data loss caused by system failures [4]. Hidden Markov Models (HMM) are statistical models used to describe Markov processes containing hidden unknown parameters. By using these parameters, the degradation status of the equipment can be deeply analyzed, and the rest of equipment life can be predicted. Software and hardware integration refers to the organic combination of computer hardware and software to ensure that they can work together to achieve specific functions or services. At present, this technology is widely used in various fields to improve the performance, stability, and availability of computer systems, but the reliability analysis process of such complex computer systems is also more complex [5]. In complex systems that integrate software and hardware, the interaction between software and hardware is complex, and existing fault diagnosis and life prediction methods are difficult to fully consider the collaborative work of software and hardware, resulting in limited accuracy of diagnosis and prediction. Therefore, this study proposes FDM based on improved Bat Algorithm (BA) and LPM based on HMM. The research aims to propose an effective computer fault diagnosis and life prediction model to detect potential faults in complex computer systems in advance and improve their reliability. The innovation of the research lies in using Genetic Algorithm (GA) to improve BA and enhance its population diversity.

2 Literature review

CSRA can effectively improve the reliability of computer systems, reduce the possibility of failures, and ensure that the system can operate stably under specified conditions and within specified time. Chi Y et al. reviewed the latest research progress on knowledge-based fault diagnosis knowledge bases in response to the high complexity of reliability detection in Industrial Internet of Things (IIoT) systems caused by the increasing level of connectivity between devices. This study contributed to fault detection and isolation of specific components in IIoT systems [6]. John Y M et al. proposed a multi-hardware software fault interaction reliability model to address the reliability issues of computer system hardware and software. This model established and solved differential difference equations to obtain formulas for mean time to failure, profit, and steady-state availability, which helps to improve the lifespan and operational efficiency of computer systems [7]. Verma A et al. proposed a reliability assessment model that considers the operation of hardware software systems in different modes, addressing the issue of reliability assessment models not taking into account the time, compatibility, and dependencies between hardware and software. This model has been applied to time critical stepper motor systems and has certain effectiveness [8]. De Sio et al. proposed a neural network resilience evaluation method based on programmable hardware to address the issue of the rapid increase in complexity of modern neural networks, which leads to an increased demand for computing power in the required network architecture. This method could involve the hardware execution of neural networks in reliability analysis and solve specific fault models [9]. Haque et al. stated that the software reliability growth model explains system reliability by analyzing the fault dataset throughout the entire testing process, but it has certain limitations. Therefore, they introduced the mathematical foundations of five popular software reliability models, which helps promote the application of software reliability models in predicting natural health reliability [10]. Zhu M et al. established a generalized multi-environmental factor software reliability growth model to address the impact of environmental factors on software reliability, which can help promote the development of large-scale software development. Tests on real-world datasets have shown that the model has good software reliability [11]. Shahin M et al. proposed a fault detection model based on deep learning and gradient enhancement algorithms, which achieved an average accuracy of over 90% in the synthetic predictive maintenance dataset, indicating that deep learning can demonstrate satisfactory performance in system reliability analysis [12]. Yuan Z et al. proposed a system fault diagnosis model based on velocity adaptive graph convolutional networks. The results indicate that the proposed model has an average accuracy of 98.83% and good robustness [13].

HMM is a statistical model used to describe Markov processes with hidden unknown parameters. Singh A et al. summarized the performance matrix and evaluation criteria used to evaluate fraud detection systems for automatic detection of credit card fraud, and used HMM models for probability statistics, which will provide useful directions for research in the field of credit card fraud detection [14]. Moudoud H et al. proposed an HMM-based IoT device attack detection and prediction module to address the vulnerability of IoT systems to false data injection attacks. The proposed model had high attack detection accuracy and efficiency, ensuring the reliable operation of IoT devices [15]. Liberali G et al. proposed using HMM to evaluate the real-time response performance of website visitors in response to the poor real-time response caused by the rich design elements in the website. It combined the multi-armed bandit model to understand the effectiveness of hidden states and web design, which helped improve the visitor experience [16]. Sidrow et al. suggested that datasets composed of high-frequency time sampled curve sequences may exhibit complex dependency structures, making modeling difficult. To this end, a hierarchical method was proposed that treats curves as observations of HMMs and uses Fourier analysis for data transformation. The proposed method could generate interpretable state estimates and accurate parameter estimates [17]. Salehian M et al. used HMM for fault detection on a test bench to address the issue of adjusting controller parameters, to improve the engine design process and meet fuel consumption and driving comfort demands. The practicality of the proposed method was verified through an industrial engine test bench [18]. Che J et al. investigated the active fault-tolerant control problem of a class of discrete-time Markov jump LPV systems. It utilized time-varying HMM to characterize the data transmission between fault diagnosis and controller reconstruction mechanisms, and verified the applicability of the proposed method through numerical examples [19]. Li J et al. proposed a time series prediction fuzzy model based on fuzzy rules and HMM model. The experimental results show that the proposed model exhibits better performance than models based on fuzzy rules [20]. Shao W et al. proposed a semi supervised Bayesian HMM dynamic soft sensing method to address the issue of industrial data complexity hindering the development of high-precision soft sensors. The results indicate that the proposed method can effectively address the issue of missing values in quality variables [21].

In summary, although current CSRA has been affirmed the effectiveness of HMM in data analysis, there is still rare research on CSRA related to HMM. Therefore, this study is based on HMM to analyze the reliability of computer systems, aiming to ensure their stable operation.

3 Research methodology

3.1 FDM construction based on improved BA

With the increasing complexity of computer systems and the increasing needs for system reliability, it is necessary to quickly locate and diagnose faults in complex computer systems. The Malek model is a model used for system fault diagnosis, mainly for diagnosing faults that occur during the communication process of various nodes in multi machine systems. Assuming there are n nodes in a processor system, the topology of the system is $G(U, E, W)$. $U = \{u_1, u_2, ..., u_n\}$ is the node set, $E = \{e_1, e_2, ..., e_n\}$ is the edge set between nodes, and $W = \{w_1, w_2, ..., w_n\}$ is the weight of the edges. To determine the fault node, a constraint equation is designed as shown in formula (1).

$$f(u_i, u_j) = \begin{cases} u_i + u_j = 0, e_{ij} = 0 \\ (u_i + u_j - 2)u_i u_j = 0, e_{ij} = 1 \end{cases} \tag{1}$$

In formula (1), u_i and u_j denote the states of nodes i and j. e_{ij} means the comparison result between adjacent nodes. To improve the efficiency of Malek model fault diagnosis, the study introduces BA to achieve more efficient parameter optimization and problem solving. BA is a heuristic search algorithm built on swarm intelligence, proposed by Professor Xin She Yang in 2010. BA simulates the bat's behaviors in nature using sonar to detect prey and avoid obstacles [22]. The basic principle of BA is to map bat individuals into feasible ways in a multidimensional issue space, and search for the global optimal solution through iterative optimization process, which owns the advantages of simple implementation and few parameters [23, 24]. Fig. 1 shows the acoustic localization of bats.

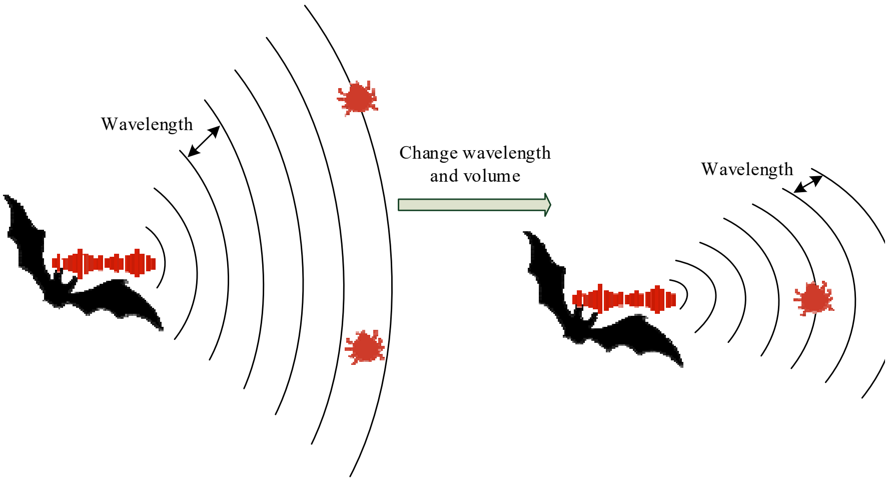

Figure 1: Schematic diagram of bat sound wave localization

In Fig. 1, bats locate prey and obstacles by emitting ultrasound waves and receiving reflected echoes, searching for prey with a fixed frequency, variable wavelength, and volume. Pulse frequency, sound intensity, and pulse emission frequency have significant impacts on the adaptive iteration of BA. In BA, the calculation of the search pulse frequency fi emitted by bat i is shown in formula (2).

$$fi = f\min + (f\max - f\min)\,\varepsilon \tag{2}$$

In formula (2), ε is a random number with values within [0,1], and $f\max$ and $f\min$ are the max and min of fi. During the random flight search process, the speed and position update rules for the t-th iteration of bat i are shown in formula (3).

$$\begin{cases} v_i^t = v_i^{t-1} + \left(x_i^{t-1} - x^*\right) fi \\ x_i^t = x_i^{t-1} + v_i^t \end{cases} \tag{3}$$

In formula (3), v_i^t and x_i^t are the velocity and location of bat i's t-th iteration. v_i^{t-1} and x_i^{t-1} are the velocity and location of bat i's $t-1$-th iteration. x^* is the present local optima. The local search process of BA is to generate a novel solution around the optima. Once a solution is taken from the current optima, the random walk method is utilized to produce a new local solution nearby. If the bat is at the global optima, the bat individual updates its position based on its loudness, as shown in formula (4).

$$xnew = xold + \gamma A^k \tag{4}$$

In formula (4), $xold$ and $xnew$ are the positions before and after the update. γ is a random number with a value range of [-1,1]. A^k is the average loudness of all bats at time k. During the search for prey phase, the algorithm's parameters are updated as shown in formula (5).

$$\begin{cases} A_i^{k+1} = aA_i^k \\ r_i^{k+1} = r_i^0\left[1 - \exp(-\delta k)\right] \end{cases} \tag{5}$$

In formula (5), A_i^{k+1} and r_i^{k+1} are the pulse loudness and pulse emission frequency of bat i at time $k+1$. a and δ represent constants greater than zero. r_i^0 is the initial pulse emission frequency of bat i. As the iterations increase, the loudness will gradually approach 0, and ri will gradually approach r_i^0

until the global optimal solution is found. However, BA also faces issues of poor population diversity and early maturity. Therefore, this study introduces GA with good global search performance to enrich the population diversity of BA. GA is a random search algorithm that iteratively updates the solution to optimization problems by simulating natural selection and genetic principles. Its core idea originates from the laws of evolution in nature [25, 26]. The main steps of GA include population initialization, selection operation, crossover operation, and mutation operation. Among them, crossover operation means the process of generating novel individuals by exchanging partial genes of two parent individuals. It can add the population's diversity, assist the algorithm in exploring new solutions in the search space, and thus improving the global search capacity. The two-point intersection used is shown in Fig. 2.

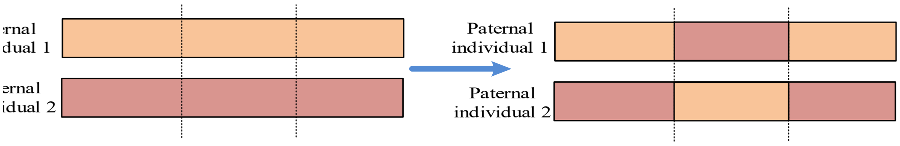

Figure 2: Schematic diagram of two-point intersection

In Fig. 2, two-point crossover refers to gene exchange between two randomly selected intervals of two parent individuals, thereby generating two new offspring individuals and ensuring that the exchanged gene segments are complete. The two-point crossover simulates the phenomenon of gene recombination in the process of biological evolution, which helps the algorithm explore new solutions in the search space. The selection operation aims to optimize the search by selecting individuals with higher fitness to construct the next generation population. In the selection operation, this study adopts the roulette wheel selection method, and the core principle lies in the possibility of an individual being chose is proportional to its fitness function value. This method first calculates the fitness value of each individual in the population, and then calculates their probability of being selected based on their fitness value. The calculation of the probability P of an individual being selected is shown in formula (6).

$$P = p/(\sum_{i=1}^{N} p) \tag{6}$$

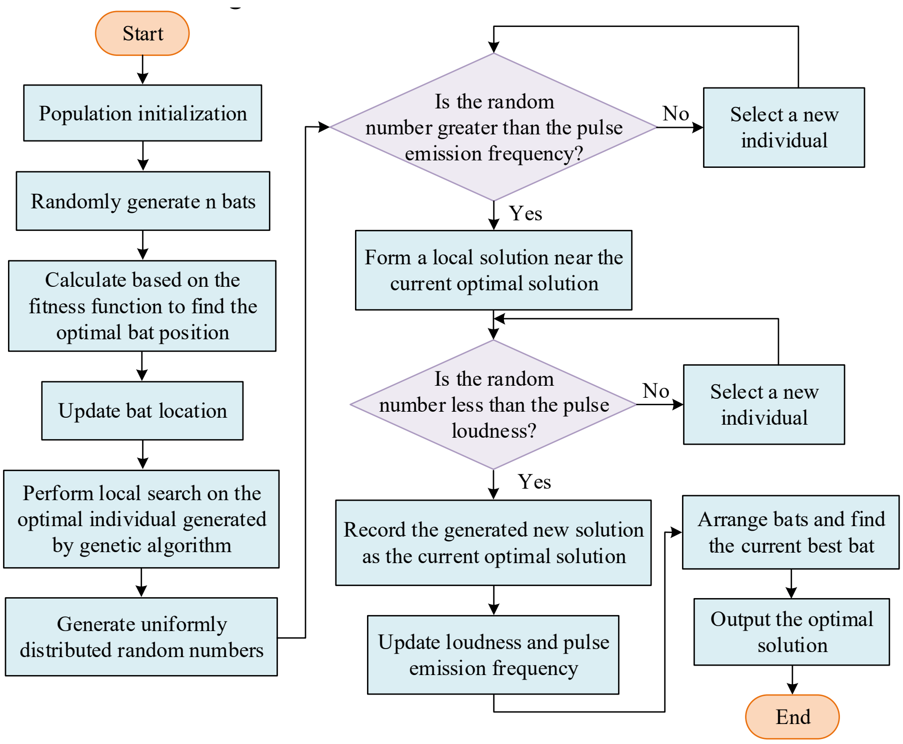

Figure 3: Flowchart of fault diagnosis model based on improved BA

In formula (6), p is the individual's fitness value. N is the population size. After calculating the probability P, random numbers within [0,1] are selected for comparison. If the random number is greater than P, the individual is not selected, otherwise it is selected. Mutation operation refers to changing the gene values on certain loci of individual strings in a population, aiming to introduce new genes, increase population diversity, and help avoid algorithms getting stuck in local optima. To improve the robustness and adaptability, this paper adopts a simple single point mutation method. Single point mutation involves randomly selecting a point as the mutation point, and then modifying the genes of the chromosome at that point to introduce diversity and prevent the algorithm from getting stuck in local optima. In summary, the flowchart of FDM based on improved BA is shown in Fig. 3.

In Fig. 3, FDM based on improved BA utilizes the genetic competition mechanism introduced in GA to enhance the diversity of the population and improve the optimization performance of BA. To investigate the impact of parameter settings on the performance of a fault diagnosis model based on an improved BA algorithm, a single parameter method was used to observe the changes in the output results of the model by changing the input parameters one by one. Specifically, following the parameter settings of Deb K et al [27]. The study first set the mutation probability to 0.3 and the crossover probability to 0.5, exploring the changes in CPU runtime under different selection probabilities. Then, set the selection probability to 0.3 and the crossover probability to 0.5, and change the mutation probability. The probability of crossing is the same.

3.2 LPM based on HMM

FDM can only provide fault information for computer systems and cannot provide predictive information about equipment lifespan, lacking comprehensive reliability assessment. Therefore, to further analyze the reliability of complex computer system equipment that integrates software and hardware, this study will establish LPM. HMM is vital for studying the state space of discrete event dynamic systems. Assuming that the set of hardware performance degradation indicators in a computer system is $I = \{i_1, i_2, ..., i_n\}$, and the set of all possible operating states of m software is $S = \{s_1, s_2, ..., s_m\}$. The observed operating state sequence is $R = \{r_1, r_2, ..., r_T\}$, and the corresponding degradation indicator observation sequence is $H = \{h_1, h_2, ..., h_T\}$. T represents the sequence length. HMM is defined by the initial state probability matrix, the hidden state transition probability matrix $B = [a_{ij}]_{m \times n}$, and the transition probability matrix $C = [b_j(k)]_{m \times n}$ of the observed state of the hardware performance degradation index. a_{ij} is the probability of s_i first and then s_j. $b_j(k)$ is the probability of generating observation i_k in state s_j. The degradation state evolution process based on HMM is shown in Fig. 4.

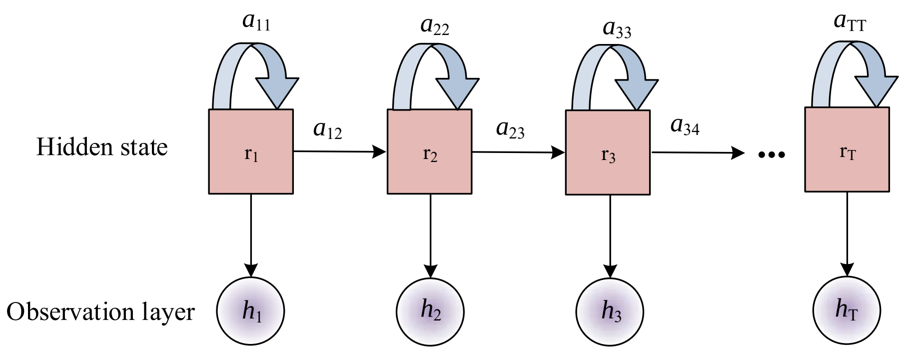

Figure 4: Degradation state evolution process based on HMM

Before establishing HMM, Step 1 is to determine the amount of hidden states in the model, and the study uses Bayesian Information Criterion (BIC) to determine it. BIC is a criterion used for model selection, aimed at balancing the fit goodness and model complexity to select the best model [28]. Assuming $x_1, x_2, ..., x_b$ is independent and identically distributed, the posterior probability $f(g\,|\theta)$ is shown in formula (7).

$$f(g\,|\theta)\,|\theta = (\gamma_1, \gamma_2, ..., \gamma_k), \theta \in \vartheta \tag{7}$$

In formula (7), θ is the model, γ is the observed data, k is the quantity of hidden states, and ϑ is the model space. The definition of BIC is shown in formula (8).

$$B(f) = k \ln b - 2 \ln L_{\hat{\theta}_k}(X) \tag{8}$$

In formula (8), $\ln L_{\hat{\theta}_k}(X)$ is the maximum likelihood estimate of $f(g\,|\theta)$. This study uses the Baum-Welch Algorithm (BWA) to estimate the parameters of HMM. The BWA is an iterative optimization algorithm used to train HMMs, which maximizes the likelihood function of HMMs through two main steps: iterative expectation and maximization [29]. The BWA first initializes the HMM parameters, then calculates the forward and backward probabilities for each time step, and calculates the posterior probabilities for each state at each time step based on these probabilities. Then, based on the posterior probability, the algorithm re-estimates the model parameters by maximizing the logarithmic likelihood function, repeating until the model parameters converge. In addition, this study also uses the Viterbi algorithm to obtain the optimal hidden sequence estimation. The Viterbi is a dynamic programming algorithm utilized to look for the most likely Hidden State Sequence (HSS) to generate an observed event sequence [30]. In HMM, the observation sequence is a sequence composed of a series of random events, and the HSS is the underlying cause of these events. The Viterbi calculates the possibility of each state and selects the path with the highest possibility to find the most likely HSS. The performance degradation model of software and hardware systems is shown in formula (9).

$$D(n) = Y(n) - Y(n-1) = \begin{cases} \mu\left(Z(n-1)\right)h, Z(n) = Z(n-1) \\ J\left(Z(n-1), Z(n)\right), Z(n) \neq Z(n-1) \end{cases} \tag{9}$$

In formula (9), $D(n)$ is the incremental degradation of system performance. $Y(n)$ is the sensor indicator value predicted at time nh. $Y(n-1)$ is the sensor indicator value predicted at time $(n-1)h$. $J\left(Z(n-1), Z(n)\right)$ is the degradation of system performance when the system transitions from state $Z(n-1)$ to state $Z(n)$. $\mu\left(Z(n-1)\right)$ is the degradation rate. Assuming $Z(n-1) = i$, $Z(n) = j$, then $D(n)$ has a probability density function $G(d|\, i, j, \varpi)$. ϖ is a parameter vector. The expected degradation $E\left[Y(n)\right]$ of computer system performance under different software operating states is shown in formula (10).

$$E\left[Y(n)\right] = Y(n_c) + \sum_{n=n_c+1}^{N} E\left[G(d\,|Z(n-1) = i, Z(n) = j, \varpi\right] \tag{10}$$

In formula (10), $Y(n_c)$ is the performance indicator degradation observed at time $t_{n_c} = n_c h$. n_c is the current number of cycles. Fig. 5 shows the process of system level remaining life prediction.

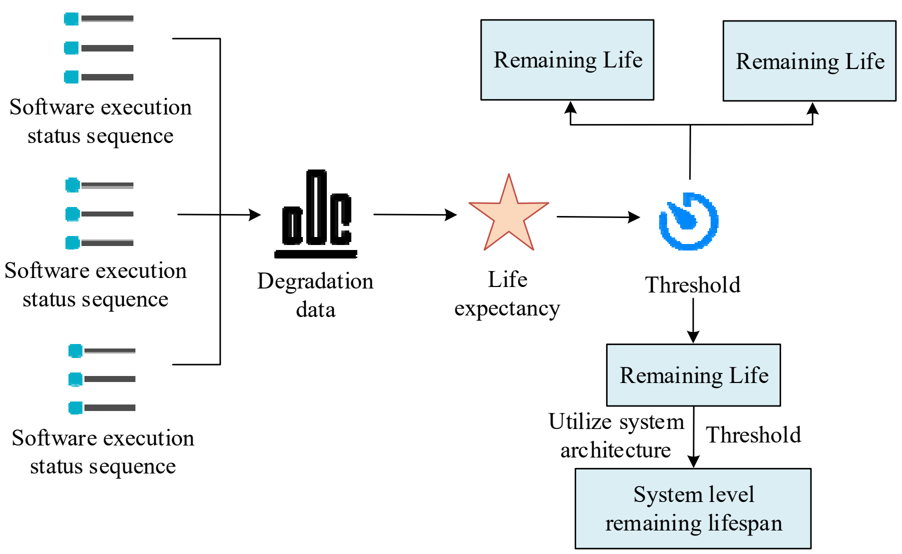

Figure 5: Flowchart for system level remaining life prediction

In Fig. 5, firstly, m software execution transfer path is generated through simulation, and the corresponding performance indicator degradation amount is generated. Secondly, by calculating the expected degradation value based on the degradation path data, the remaining lifespan of the system can be obtained. The next step is to perform weighted averaging on the estimated values. The final step is to estimate the remaining lifespan of different subsystems separately and use the system structure to estimate the system level remaining lifespan.

4 Results and discussion

4.1 Analysis of the effect of FDM

To investigate the impact of different GA parameters on the performance of FDM based on improved BA, this study sets the population size to 30 and the iteration count to 500. This study will conduct experiments on the Windows 10 system using Intel (R) Pentium 3558U@1.70GHz Processor with 16GB of memory. Fig. 6 shows the CPU runtime of the model under different parameters. In Figs. 6 (a), (b), and (c), as the values of selection probability, crossover probability, and mutation probability change, the running time of the CPU also fluctuates continuously. This indicates that the proposed fault diagnosis model based on the improved BA algorithm is sensitive to parameter settings for selection probability, crossover probability, and mutation probability. Even small parameter adjustments may lead to significant increases or decreases in CPU running time. When these three probability values are set to 0.3, 0.5, and 0.35, the CPU runtime is the shortest and the efficiency of the model is higher.

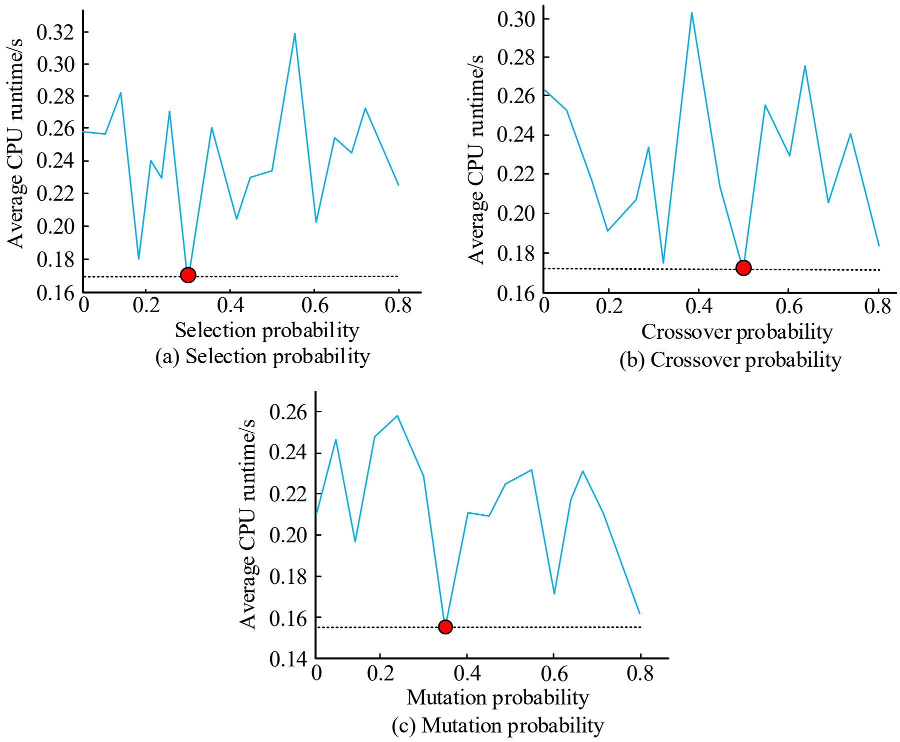

Figure 6: The impact of different parameters on model performance

In Fig. 6, the selection probability is 0.3, the crossover probability is 0.5, and the mutation probability is 0.35. To further verify the superiority of the improved BA, it is compared with the currently advanced FDM that combines Convolutional Neural Networks Gated Recurrent Unit (CNN-GRU), CNN and Bidirectional Long Short-Term Memory (CNN-BiLSTM), and PSO-GA, as shown in Fig. 7. In Fig. 7 (a), compared to the other three models, the proposed FDM has fewer iterations, followed by the PSO-GA, and the CNN-BiLSTM has the highest number of iterations. In Fig. 7 (b), the average CPU runtime of the research model is the shortest, not exceeding 20 seconds. The CNN-BiLSTM has the longest average CPU runtime. This indicates that the improved FDM of BA has faster operational efficiency and demonstrates certain superiority.

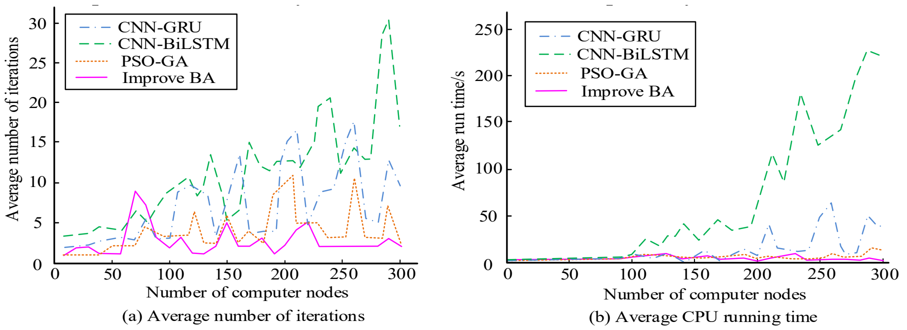

Figure 7: Comparison of iteration times and runtime of four algorithms

4.2 Performance analysis of LPM

To verify the feasibility of HMM based LPM, this study detects and preprocesses the vibration signals of various devices in a complex system that integrates software and hardware. The two hardware datasets obtained are named Dataset 1 and Dataset 2, respectively. Fig. 8 shows the waveform of two datasets.

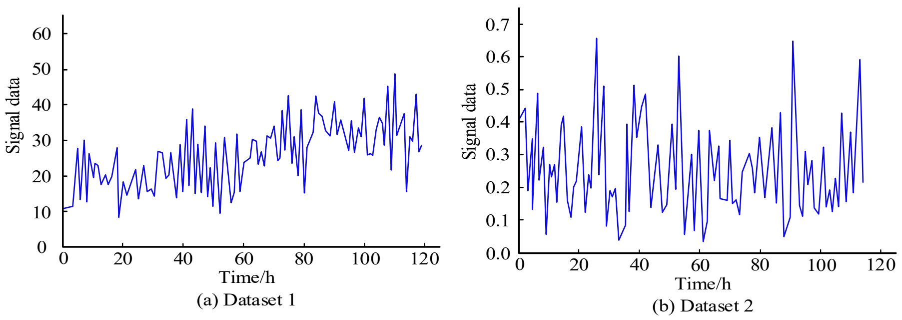

Figure 8: Waveform diagram of the dataset

Set the error ranges to [-2,2] and [-0.05,0.05] in two datasets, respectively. Fig. 9 shows the remaining life prediction results comparing the proposed model with BPNN. In Figs. 9(a) and (b), in both datasets, the predicted curve of the research model is next to the true curve, and the prediction error is smaller than BPNN. Therefore, HMM-based LPM has good residual life prediction performance and high prediction accuracy.

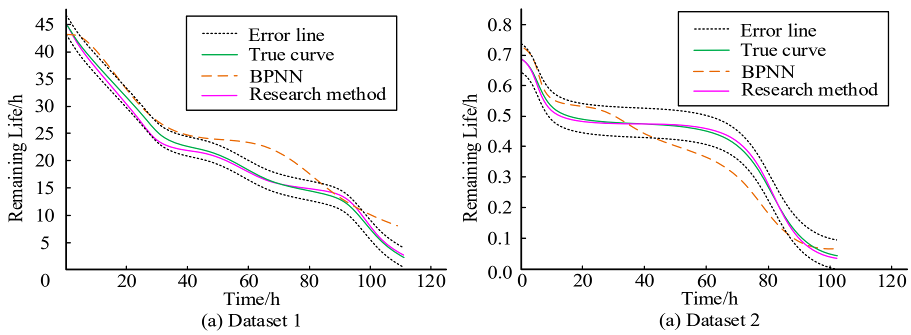

Figure 9: Comparison of remaining life prediction results

To further validate the superiority of the proposed model, accuracy, recall, and root mean square error were used as evaluation metrics with a confidence interval of 95%. Compare the proposed model with the Transformer model, the Sparrow Search Algorithm LSTM (SSA-LSTM), and the PSO-LSTM, as shown in Table 1. Among the four models, the accuracy (92.47%) and recall (91.73%) of the research model are the highest, while the RMSE (0.021) is the lowest. Therefore, HMM-based LPM has good prediction accuracy and certain superiority.

Table 1: Comparison results of indicators for four models

Index	Models			
	Transformer	SSA-LSTM	PSO-LSTM	Research method
Accuracy/%	78.65	85.47	88.73	92.47#&*
Recall/%	77.21	84.31	86.54	91.73#&*
RMSE	0.437	0.216	0.227	0.021#

Note: # indicates a significant difference compared to the Transformer model (P<0.05). & indicates a significant difference compared to the SSA-LSTM model (P<0.05). * indicates a significant difference compared to the PSO-LSTM model (P<0.05).

To further investigate the robustness of the proposed model, the Maryland battery dataset and the Mechanical Fault Prevention Technology Society MFPT dataset were used for testing, covering various operating conditions and fault modes. The accuracy comparison results of the above four models are shown in Figure 10. From Figures 10 (a) and (b), it can be seen that the accuracy of the proposed model is above 90% in both datasets, which is higher than the other three models. The results show that the proposed model exhibits good performance and robustness in different types of datasets.

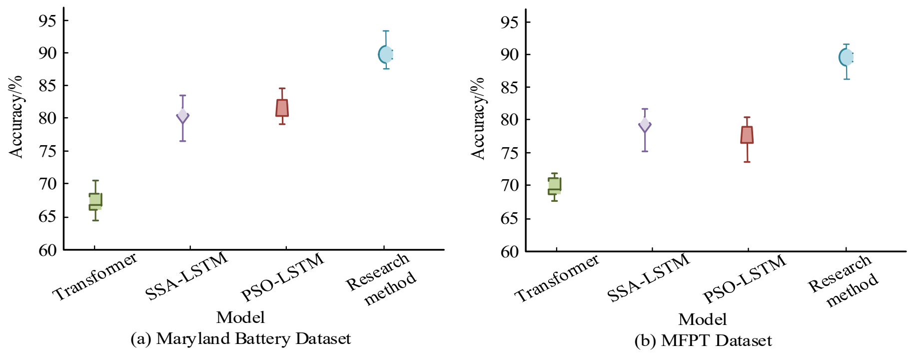

Figure 10: Comparison of Accuracy among Four Models

5 Conclusion

For the stability analysis of complex computer systems, this study developed FDM based on improved BA and LPM based on HMM. In the experiment, when the selection, crossover, and mutation probabilities were 0.3, 0.5, and 0.35, the CPU running time was the shortest and the efficiency of the model was higher. In SPTF, compared to BA, GA, and PSO, the improved BA had the lowest fitness value and a faster iteration speed. It tended to converge after about 400 iterations. In MPTF, the fitness value of the proposed model was still the lowest and began to converge after about 230 iterations. The diagnostic accuracy of the research model was relatively high, at 95.68%, with an AUC of 96.83%. The research model had the least number of iterations, followed by PSO-GA, and CNN-BiLSTM had the most iterations. The average running time of FDM's CPU was the shortest, not exceeding 20s. CNN-BiLSTM had the longest average CPU running time. The predicted curve of the proposed model was relatively close to the true curve, with a prediction error smaller than BPNN. Its accuracy and recall were the highest, at 92.47% and 91.73%, respectively, and its RMSE was the lowest, at 0.021. However, the residual life prediction model proposed by the research institute, while using GA algorithm to improve BA algorithm for model performance, also brings an increase in model complexity. The increase in model complexity usually means that more computing resources are needed for training and prediction, and often more parameters need to be adjusted, which may limit the actual deployment of the model in environments with limited computing resources. Therefore, in future research, more concise residual life prediction models should be further explored, such as reducing the number of layers or parameters in the model and lowering computational complexity. And develop automated parameter adjustment methods using hyperparameter optimization algorithms such as grid search and random search to improve the applicability of the model in different systems.

Acknowledge

Talent research launch start-up project of Fuyang Normal University under grant 2019kyqd0018 and Anhui Province university key research project under grant 2022AH052820.

References

[1] Kim K, Lee J H, Lim H K, Oh S W, Han Y H. Deep RNN-based network traffic classification scheme in edge computing system. Computer Science and Information Systems, 2022, 19(1): 165-184, DOI: 10.2298/CSIS200424038K.

[2] Zeng Y, Sun Y, Xu T, Su S. A reliability evaluation method for complex systems based on the editable GSPN and adaptive Monte Carlo simulation. Systems Engineering, 2024, 27(3): 520-531, DOI: 10.1002/sys.21736.

[3] Pang J L. Adaptive Fault Prediction and Maintenance in Production Lines Using Deep Learning. International Journal of Simulation Modelling (IJSIMM), 2023, 22(4):734-745, DOI: 10.2507/IJSIMM22-4-CO20.

[4] Jianling T A N, ZHANG X, Dan L I, et al. Research on Damage Detection of Civil Structures Based on Machine Learning of Multiple Vegetation Index Time Series. Technical Gazette/Tehnički Vjesnik, 2024, 31(3): 906-914, DOI: 10.17559/TV-20240104001243.

[5] Doinea M, Trandafir I, Toma C V, Popa M, Zamfiroiu A. IoT Embedded Smart Monitoring System with Edge Machine Learning for Beehive Management. International Journal of Computers Communications & Control, 2024, 19(4), DOI: 10.15837/ijccc.2024.4.6632.

[6] Chi Y, Dong Y, Wang Z J, Yu F R, Leung V C. Knowledge-based fault diagnosis in industrial internet of things: a survey. IEEE Internet of Things Journal, 2022, 9(15): 12886-12900, DOI: 10.1109/JIOT.2022.3163606.

[7] John Y M, Sanusi A, Yusuf I, Modibbo U M. Reliability analysis of multi-hardware–software system with failure interaction. Journal of Computational and Cognitive Engineering, 2023, 2(1): 38-46, DOI: 10.47852/bonviewJCCE2202216.

[8] Verma A, Gayen T. Reliability Assessment of Combined Hardware–Software Non-repairable Time-Critical Systems. The Computer Journal, 2023, 66(7): 1644-1663, DOI: 10.1093/comjnl/bxac032.

[9] De Sio C, Azimi S, Sterpone L. FireNN: Neural networks reliability evaluation on hybrid platforms. IEEE Transactions on Emerging Topics in Computing, 2022, 10(2): 549-563, DOI: 10.1109/TETC.2022.3152668.

[10] Haque M A, Ahmad N. Key issues in software reliability growth models. Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science), 2022, 15(5): 741-747, DOI: 10.2174/2666255813999201012182821.

[11] Zhu M, Pham H. A generalized multiple environmental factors software reliability model with stochastic fault detection process. Annals of Operations Research, 2022, 311(1): 525-546, DOI: 10.1007/s10479-020-03732-3.

[12] Shahin M, Chen F F, Hosseinzadeh A, Zand N. Using machine learning and deep learning algorithms for downtime minimization in manufacturing systems: An early failure detection diagnostic service. The International Journal of Advanced Manufacturing Technology, 2023, 128(10): 3857-3883, DOI: 10.1007/s00170-023-12020-w.

[13] Yuan Z, Ma Z, Li X, Cui Y. Speed adaptive graph convolutional network for wheelset-bearing system fault diagnosis under time-varying rotation speed conditions. Journal of Vibration Engineering & Technologies, 2024, 12(1): 247-258, DOI: 10.1007/s42417-022-00841-0.

[14] Singh A, Jain A. An empirical study of AML approach for credit card fraud detection—financial transactions. International Journal of Computers Communications & Control, 2019, 14(6): 670-690, DOI: 10.15837/ijccc.2019.6.3498.

[15] Moudoud H, Mlika Z, Khoukhi L, Cherkaoui S. Detection and prediction of fdi attacks in iot systems via hidden markov model. IEEE Transactions on Network Science and Engineering, 2022, 9(5): 2978-2990, DOI: 10.1109/TNSE.2022.3161479.

[16] Liberali G, Ferecatu A. Morphing for consumer dynamics: Bandits meet hidden Markov models. Marketing Science, 2022, 41(4): 769-794, DOI: 10.1287/mksc.2021.1346.

[17] Sidrow E, Heckman N, Fortune S M E, Trites A W, Murphy I, Auger-Méthé M. Modelling multi-scale, state-switching functional data with hidden Markov models. Canadian Journal of Statistics, 2022, 50(1): 327-356, DOI: 10.1002/cjs.11673.

[18] Salehian M, Haghani A, Jeinsch T. Application of Hidden Markov Models for Fault Detection in Automotive Engines. IFAC-PapersOnLine, 2022, 55(6): 767-771, DOI: 10.1016/j.ifacol.2022.07.219.

[19] Che J, Zhu Y, Basin M V, Zhou D. Active fault-tolerant control for discrete-time Markov jump LPV systems via time-varying hidden Markov model approach. International Journal of Control, Automation and Systems, 2022, 20(6): 1785-1799, DOI: 10.1007/s12555-021-0109-x.

[20] Li J, Pedrycz W, Wang X, Liu P. A Hidden Markov Model-based fuzzy modeling of multivariate time series. Soft Computing, 2023, 27(2): 837-854, DOI: 10.1007/s00500-022-07623-6.

[21] Shao W, Xiao C, Wang J, Zhao D, Song Z. Real-time estimation of quality-related variable for dynamic and non-Gaussian process based on semisupervised Bayesian HMM. Journal of Process Control, 2022, 111(1):59-74, DOI: 10.1016/j.jprocont.2022.01.007.

[22] Worasan K, Sethanan K, Moonsri K, Golinska-Dawson P. The multi-product vehicle routing problem with cross-docking: a novel strategy hybrid bat algorithm for Industry 3.5 in Thailand's food industry. International Journal of Logistics Research and Applications, 2024, 27(2): 284-308, DOI:10.1080/13675567.2023.2286352 .

[23] Umar S U, Rashid T A, Ahmed A M, Hassan B A, Baker M R. Modified Bat Algorithm: a newly proposed approach for solving complex and real-world problems. Soft Computing, 2024, 28(13): 7983-7998, DOI: 10.1007/s00500-024-09761-5.

[24] Yang B, Li H, Xing Y, Zeng F, Qian C, Shen Y, Wang J. Directed search based on improved whale optimization algorithm for test case prioritization. International Journal of Computers Communications & Control, 2023, 18(2), DOI: 10.15837/ijccc.2023.2.5049.

[25] Obeidat A, Al-Shalabi M. An efficient approach towards network routing using genetic algorithm. International Journal of Computers Communications & Control, 2022, 17(5), DOI: 10.15837/ijccc.2022.5.4815.

[26] Ganapriya K, Poobalan A, Gopinath S, Vedha Vinodha D. An Enhanced Trust Scheduling Algorithm for Medical Applications in a Heterogeneous Cloud Computing Environment. Technical Gazette, 2024, 31(3): 945-950, DOI: 10.17559/TV-20230913000935.

[27] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 2002, 6(2), 182-197, DOI: 10.1109/4235.996017.

[28] Cuc L D, Rad D, Săplăcan S, Șendroiu C, Bâtcă-Dumitru G C, Wysocki D, Duțu A, Manolescu A A. A hierarchical clustering analysis of the management accounting practices perceptions in Romania. International Journal of Computers Communications & Control, 2024, 19(6), DOI: 10.15837/ijccc.2024.6.6864.

[29] Zhang S, Yang L T, Zhang Y, Lu Z, Yu J, Cui Z. Tensor-Based Baum–Welch Algorithms in Coupled Hidden Markov Model for Responsible Activity Prediction. IEEE Transactions on Computational Social Systems, 2023, 10(6): 2924-2937, DOI: 10.1109/TCSS.2022.3227458.

[30] Lu D, Guo F. Application of wearable motion sensor in business English teaching. Computer Science and Information Systems, 2022, 19(3): 1481-1498, DOI: 10.2298/CSIS210320020L.

Cite this paper as:

Yu, S.; Ru, X. (2026). A Hybrid Approach to Fault Diagnosis and Lifespan Prediction in Complex Computer Systems, *International Journal of Computers Communications* & *Control*, 21(1), 6873, 2026.
https://doi.org/10.15837/ijccc.2026.1.6873